

WebSphere Application Server V8.5 Administration and Configuration Guide for the Full Profile

Learn about Websphere Application
Server V8.5.5

Configure and administer a
WebSphere system

Deploy applications in a
WebSphere environment



Fabio Albertoni
Tanja Baumann
Yogesh Bhatia
Eduardo Monich Fronza
Marcio da Ros Gomes
Sebastian Kapciak
Catalin Mierlea
Sergio Pinto
Anoop Ramachandra
Liang Rui
Miguel Troncoso



International Technical Support Organization

**WebSphere Application Server V8.5 Administration
and Configuration Guide for the Full Profile**

July 2013

Note: Before using this information and the product it supports, read the information in “Notices” on page xix.

Second Edition (July 2013)

This edition applies to WebSphere Application Server V8.5, including the features in V8.5.5.

Contents

| | |
|---|------|
| Notices | xix |
| Trademarks | xx |
| Preface | xxi |
| Authors | xxi |
| Now you can become a published author, too! | xxiv |
| Comments welcome | xxiv |
| Stay connected to IBM Redbooks | xxiv |

Part 1. Installation and profile management 1

| | |
|--|----|
| Chapter 1. System management: Technical overview | 3 |
| 1.1 WebSphere Application Server profiles | 4 |
| 1.2 System management overview | 4 |
| 1.2.1 Terminology | 5 |
| 1.2.2 Directory conventions | 5 |
| 1.2.3 Core concepts of system management | 5 |
| 1.2.4 System management tools | 6 |
| 1.3 New features for administrators | 7 |
| 1.4 Java Management Extensions | 10 |
| 1.4.1 JMX architecture | 10 |
| 1.4.2 JMX MBeans | 11 |
| 1.5 System management in a stand-alone server environment | 11 |
| 1.6 System management in a distributed server environment | 12 |
| 1.6.1 Centralized changes to configuration and application data | 14 |
| 1.6.2 Rules for process startup | 18 |
| 1.6.3 Distributed process discovery | 18 |
| 1.6.4 File synchronization in distributed server environments | 20 |
| 1.7 Advanced system management of multiple stand-alone servers | 25 |
| 1.8 Advanced management of distributed and stand-alone servers | 28 |
| Chapter 2. Installing WebSphere Application Server on distributed systems | 31 |
| 2.1 IBM Installation Manager overview | 32 |
| 2.1.1 Terminology | 32 |
| 2.1.2 Capabilities | 33 |
| 2.2 Installation Manager installation | 34 |
| 2.2.1 Using the GUI installer to install Installation Manager | 34 |
| 2.2.2 Using console mode to install Installation Manager | 35 |
| 2.2.3 Using the command line to install Installation Manager | 36 |
| 2.2.4 Using the silent installer to install Installation Manager | 37 |
| 2.2.5 Uninstalling Installation Manager | 37 |
| 2.3 Using Installation Manager | 38 |
| 2.3.1 Wizard mode | 38 |
| 2.3.2 Command-line mode | 39 |
| 2.3.3 Console mode | 39 |
| 2.3.4 Silent mode | 40 |
| 2.4 Customizing Installation Manager | 41 |
| 2.4.1 Installation Manager preferences | 41 |
| 2.4.2 Repositories overview | 43 |

| | | |
|--------|---|------------|
| 2.4.3 | Repository configuration | 43 |
| 2.4.4 | Updating Installation Manager | 45 |
| 2.4.5 | Managing packages | 45 |
| 2.4.6 | Examining log files | 45 |
| 2.5 | Installing WebSphere Application Server | 46 |
| 2.5.1 | Prerequisites | 47 |
| 2.5.2 | Using GUI mode | 47 |
| 2.5.3 | Using silent mode | 50 |
| 2.6 | Installing additional software | 52 |
| 2.6.1 | WebSphere Customization Toolbox | 52 |
| 2.6.2 | Application Client for WebSphere Application Server V8.5. | 56 |
| | Chapter 3. Working with profiles on distributed systems | 59 |
| 3.1 | Types of profiles | 60 |
| 3.1.1 | Application server profile. | 60 |
| 3.1.2 | Deployment manager profile. | 60 |
| 3.1.3 | Custom profile. | 61 |
| 3.1.4 | Cell profile | 61 |
| 3.1.5 | Administrative agent profile. | 61 |
| 3.1.6 | Job manager profile | 62 |
| 3.2 | Planning for profiles | 63 |
| 3.3 | Building systems with profiles | 64 |
| 3.3.1 | Starting the WebSphere Customization Toolbox Profile Management Tool. | 64 |
| 3.3.2 | Common steps for all profiles | 65 |
| 3.3.3 | Creating an application server profile | 74 |
| 3.3.4 | Creating a deployment manager profile | 78 |
| 3.3.5 | Creating a cell profile | 80 |
| 3.3.6 | Creating a custom profile | 81 |
| 3.3.7 | Federating nodes to a cell. | 83 |
| 3.3.8 | Creating a job manager profile | 88 |
| 3.3.9 | Creating an administrative agent profile | 89 |
| 3.3.10 | Registering nodes to an administrative agent. | 90 |
| 3.3.11 | Deregistering a node from the administrative agent | 92 |
| 3.3.12 | Registering administrative nodes with a job manager. | 92 |
| 3.4 | Managing profiles with the command line | 95 |
| 3.4.1 | Listing profiles | 95 |
| 3.4.2 | Creating profiles from templates | 95 |
| 3.4.3 | Creating profiles with non-default ports. | 96 |
| 3.4.4 | Deleting profiles | 97 |
| 3.4.5 | Using the manageprofiles interactive utility. | 98 |
| | Chapter 4. Installing WebSphere Application Server on z/OS systems | 101 |
| 4.1 | IBM Installation Manager overview | 102 |
| 4.2 | Installing Installation Manager. | 103 |
| 4.2.1 | Checking prerequisites | 104 |
| 4.2.2 | Obtaining an Installation Manager installation kit | 104 |
| 4.2.3 | Installing Installation Manager on the system | 105 |
| 4.3 | Working with Installation Manager | 107 |
| 4.3.1 | Installation Manager preferences | 107 |
| 4.3.2 | Repository overview | 107 |
| 4.3.3 | Updating Installation Manager | 108 |
| 4.3.4 | Installing the WebSphere Application Server initial repository | 108 |
| 4.4 | Using Installation Manager | 108 |

| | | |
|-------------------|--|------------|
| 4.4.1 | Key features of Installation Manager | 109 |
| 4.4.2 | Uninstalling Installation Manager | 111 |
| 4.5 | Installing WebSphere Application Server | 112 |
| 4.5.1 | Installing using the command line | 112 |
| 4.5.2 | Installing additional packages | 113 |
| 4.5.3 | Creating response files | 114 |
| 4.5.4 | Installing silently | 115 |
| 4.5.5 | The post-installer | 116 |
| 4.5.6 | Service information | 116 |
| 4.5.7 | Uninstalling packages | 117 |
| 4.5.8 | Preparing the base z/OS operating system | 118 |
| 4.6 | WebSphere Customization Toolbox | 118 |
| 4.7 | Troubleshooting | 118 |
| 4.7.1 | Error message overview | 119 |
| 4.7.2 | Collecting Installation Manager information | 119 |
| Chapter 5. | Working with profiles on z/OS systems | 121 |
| 5.1 | Creating WebSphere environments | 122 |
| 5.1.1 | WebSphere Application Server for z/OS | 124 |
| 5.1.2 | WebSphere DMZ secure proxy server for z/OS | 124 |
| 5.2 | Getting started with the Profile Management tool | 124 |
| 5.3 | Creating a sample z/OS Network Deployment cell | 127 |
| 5.3.1 | Creating a deployment manager definition | 128 |
| 5.3.2 | Creating the base application server definition | 148 |
| 5.3.3 | Uploading jobs and associated instructions | 161 |
| 5.3.4 | Federating an application server | 162 |
| 5.3.5 | Uploading jobs and associated instructions | 166 |
| 5.4 | Creating a job manager profile | 166 |
| 5.4.1 | Creating the customization definition | 167 |
| 5.4.2 | Uploading the jobs and the associated instructions | 174 |
| 5.5 | Creating an administrative agent profile | 174 |
| 5.5.1 | Creating the customization definition | 175 |
| 5.5.2 | Uploading jobs and the associated instructions | 182 |
| Chapter 6. | Administration consoles and commands | 183 |
| 6.1 | Introducing the WebSphere administrative consoles | 184 |
| 6.1.1 | Starting and accessing the consoles | 185 |
| 6.1.2 | Logging into an administrative console | 187 |
| 6.1.3 | Changing the administrative console session timeout | 192 |
| 6.1.4 | The graphical interface | 192 |
| 6.1.5 | Administrative console resources scopes | 198 |
| 6.1.6 | Updating existing items | 203 |
| 6.1.7 | Adding new items | 204 |
| 6.1.8 | Removing items | 204 |
| 6.1.9 | Starting and stopping items | 205 |
| 6.1.10 | Using variables | 205 |
| 6.1.11 | Saving work | 207 |
| 6.1.12 | Getting help | 207 |
| 6.1.13 | New options in version 8.5 deployment manager administrative console | 208 |
| 6.2 | Securing the administrative console | 211 |
| 6.2.1 | Enabling security after profile creation | 211 |
| 6.2.2 | Administrative security roles | 215 |
| 6.3 | Job manager console | 216 |

| | | |
|-------------------|--|------------|
| 6.3.1 | Submitting a job with the job manager | 218 |
| 6.3.2 | Distributing files using the job manager | 225 |
| 6.4 | Using command-line tools | 227 |
| 6.4.1 | Command location | 227 |
| 6.4.2 | Key usage parameters | 228 |
| 6.4.3 | Entering commands | 228 |
| Part 2. | Administration and configuration techniques | 231 |
| Chapter 7. | Administration of WebSphere processes | 233 |
| 7.1 | Working with deployment manager | 234 |
| 7.1.1 | Deployment manager configuration settings | 234 |
| 7.1.2 | Starting and stopping the deployment manager | 238 |
| 7.1.3 | The high-availability deployment manager function | 240 |
| 7.2 | Working with the administrative agent | 241 |
| 7.2.1 | Starting and stopping the administrative agent | 241 |
| 7.2.2 | Administrative agent configuration settings | 242 |
| 7.3 | Working with the job manager | 244 |
| 7.3.1 | Starting and stopping the job manager | 244 |
| 7.3.2 | Job manager configuration settings | 244 |
| 7.4 | Working with application servers | 248 |
| 7.4.1 | Creating an application server | 248 |
| 7.4.2 | Viewing the status of an application server | 258 |
| 7.4.3 | Starting an application server | 261 |
| 7.4.4 | Stopping an application server | 266 |
| 7.4.5 | Viewing runtime attributes of an application server | 268 |
| 7.4.6 | Customizing application servers | 270 |
| 7.4.7 | Repository checkpoints service | 279 |
| 7.5 | Working with nodes in a Network Deployment environment | 282 |
| 7.5.1 | Starting and stopping nodes | 282 |
| 7.5.2 | Node agent synchronization | 285 |
| 7.5.3 | Removing a node from a cell | 287 |
| 7.5.4 | Renaming a node | 289 |
| 7.5.5 | Recovering an existing node | 289 |
| 7.5.6 | Node groups | 290 |
| 7.6 | Working with clusters | 292 |
| 7.6.1 | Creating application server clusters | 292 |
| 7.6.2 | Viewing the cluster topology | 301 |
| 7.6.3 | Managing clusters | 302 |
| 7.7 | Working with virtual hosts | 303 |
| 7.8 | Creating and updating virtual hosts | 304 |
| 7.9 | Managing applications | 305 |
| 7.9.1 | Managing enterprise applications: Administrative console | 306 |
| 7.9.2 | Preventing an enterprise application from starting on a server | 307 |
| 7.9.3 | Viewing application details | 307 |
| 7.9.4 | Finding a URL for a servlet or JSP | 309 |
| 7.10 | Enabling process restart on failure | 313 |
| 7.10.1 | Windows | 315 |
| 7.10.2 | UNIX and Linux | 316 |
| 7.10.3 | z/OS | 317 |
| Chapter 8. | Administration with scripting | 319 |
| 8.1 | Overview of WebSphere scripting | 320 |
| 8.2 | Launching wsadmin | 321 |

| | |
|--|------------|
| 8.2.1 Scripting environment properties file | 322 |
| 8.2.2 Script profile file | 323 |
| 8.2.3 Connected versus local mode | 323 |
| 8.3 Command and script invocation | 324 |
| 8.3.1 Invoking a single command (-c) | 324 |
| 8.3.2 Running script files (-f) | 324 |
| 8.3.3 Invoking commands interactively | 325 |
| 8.4 The wsadmin tool management objects | 325 |
| 8.4.1 Help | 326 |
| 8.4.2 AdminControl | 327 |
| 8.4.3 AdminConfig | 327 |
| 8.4.4 AdminApp | 327 |
| 8.4.5 AdminTask | 327 |
| 8.5 Properties file based configuration | 328 |
| 8.6 Managing WebSphere using script libraries | 329 |
| 8.6.1 Invoking script libraries | 330 |
| 8.6.2 Displaying help for script libraries | 331 |
| 8.6.3 Application script library | 332 |
| 8.6.4 Resource script library | 334 |
| 8.6.5 Security script library | 335 |
| 8.6.6 Server script library | 336 |
| 8.6.7 System management script library | 338 |
| 8.6.8 Applying performance tuning | 339 |
| 8.7 Assistance with scripting | 339 |
| 8.7.1 Enabling command assistance | 339 |
| 8.7.2 Building script files using command assist | 341 |
| 8.8 Example: Using scripts with the job manager | 343 |
| 8.8.1 Introduction | 343 |
| 8.8.2 Creating the customized script | 345 |
| 8.8.3 Submitting the job | 348 |
| 8.8.4 Verifying the results | 350 |
| 8.9 Online resources | 350 |
| Chapter 9. Accessing relational databases from WebSphere. | 351 |
| 9.1 JDBC resources | 352 |
| 9.1.1 JDBC providers and data sources | 352 |
| 9.1.2 WebSphere support for data sources | 353 |
| 9.2 Steps to define access to a database | 355 |
| 9.3 Creating an authentication alias | 356 |
| 9.4 Connecting to an IBM DB2 database | 357 |
| 9.4.1 Creating the JDBC provider | 357 |
| 9.4.2 Creating the data source | 360 |
| 9.5 Connecting to an Oracle database | 363 |
| 9.5.1 Creating the JDBC provider | 363 |
| 9.5.2 Creating the data source | 365 |
| 9.6 Connecting to an SQL Server database | 367 |
| 9.6.1 Creating the JDBC provider | 368 |
| 9.6.2 Creating the data source | 370 |
| 9.7 Configuring connection pooling properties | 373 |
| 9.8 WebSphere Application Server data source properties | 376 |
| 9.9 Shared and unshared connections | 378 |
| 9.9.1 Factors that determine sharing | 378 |
| 9.9.2 Configuring Shared and Unshared Connections | 379 |

| | | |
|--|--|------------|
| 9.10 | Troubleshooting database access problems | 379 |
| 9.10.1 | Enabling JDBC tracing for database problems | 380 |
| 9.10.2 | Enabling ConnLeakLogic | 380 |
| 9.10.3 | Dumping connection pool information using wsadmin | 381 |
| 9.10.4 | Tool to debug Database Access problems | 381 |
| Chapter 10. Accessing EIS applications from WebSphere | | 383 |
| 10.1 | JCA resource adapters | 384 |
| 10.2 | WebSphere Application ServerJCA support | 385 |
| 10.2.1 | Resource adapters | 386 |
| 10.2.2 | Connection factories | 386 |
| 10.3 | Installing and configuring resource adapters | 387 |
| 10.4 | Configuring J2C connection factories | 391 |
| 10.5 | Resource authentication | 393 |
| 10.5.1 | Container-managed authentication | 394 |
| 10.5.2 | Component-managed authentication | 394 |
| Chapter 11. Configuring messaging providers | | 397 |
| 11.1 | Messaging providers introduction | 398 |
| 11.2 | Configuring resources for the default messaging provider | 398 |
| 11.2.1 | Configuring JMS connection factories. | 398 |
| 11.2.2 | Configuring JMS destinations | 400 |
| 11.2.3 | Configuring JMS queues. | 401 |
| 11.2.4 | Configuring JMS activation specifications. | 402 |
| 11.3 | Configuring resources for the WebSphere MQ messaging provider. | 403 |
| 11.3.1 | Configuring WebSphere MQ messaging provider connection factories | 404 |
| 11.3.2 | Configuring WebSphere MQ messaging provider destinations | 406 |
| 11.3.3 | Configuring WebSphere MQ messaging provider activation specifications | 409 |
| 11.4 | Configuring resources for third-party messaging providers. | 412 |
| 11.4.1 | Configuring JMS messaging providers | 413 |
| 11.4.2 | Configuring JMS connection factories. | 413 |
| 11.4.3 | Configuring JMS destinations | 414 |
| Chapter 12. Configuring and managing web servers | | 417 |
| 12.1 | Web server overview and basic concepts. | 418 |
| 12.1.1 | Request routing using the plug-in | 419 |
| 12.1.2 | Web server and plug-in management. | 420 |
| 12.2 | Installations | 424 |
| 12.3 | Web server configuration using the WebSphere Customization Toolbox | 425 |
| 12.3.1 | Configuration files | 426 |
| 12.3.2 | Stand-alone server environment | 426 |
| 12.3.3 | Distributed server environment | 428 |
| 12.3.4 | Configuring a remote web server in a distributed environment. | 431 |
| 12.4 | Working with web servers and plug-ins. | 439 |
| 12.4.1 | Manually defining nodes and web servers | 439 |
| 12.4.2 | Viewing the status of a web server | 443 |
| 12.4.3 | Starting and stopping a web server | 444 |
| 12.4.4 | IBM HTTP Server remote administration | 445 |
| 12.4.5 | Mapping modules to servers | 449 |
| 12.5 | Working with the plug-in configuration file. | 450 |
| 12.5.1 | Regenerating the plug-in configuration file | 452 |
| 12.5.2 | Propagating the plug-in configuration file | 457 |
| 12.5.3 | Modifying the plug-in request routing options | 458 |
| 12.6 | IBM HTTP Server and Web Server Plug-ins for IBM WebSphere Application Server for | |

| | |
|--|------------|
| z/OS | 461 |
| 12.6.1 IBM HTTP Server | 461 |
| 12.6.2 Web Server Plug-ins for IBM WebSphere Application Server for z/OS | 461 |
| 12.7 Troubleshooting some common errors | 466 |
| 12.7.1 Troubleshooting Error 404 | 466 |
| 12.7.2 Troubleshooting Error 500 | 467 |
| Chapter 13. Intelligent management | 469 |
| 13.1 Introduction to Intelligent Management | 470 |
| 13.2 Configuring dynamic operations | 472 |
| 13.2.1 Creating the ODR | 473 |
| 13.2.2 Service policies | 475 |
| 13.2.3 Creating service policies | 478 |
| 13.2.4 Associating service policies with an application | 481 |
| 13.3 Configuring health management | 484 |
| 13.3.1 Health conditions | 484 |
| 13.3.2 Enabling and disabling health management | 485 |
| 13.3.3 Health policy actions | 486 |
| 13.3.4 Reaction mode | 487 |
| 13.3.5 Creating health policies | 487 |
| Part 3. Managing distributed systems | 491 |
| Chapter 14. Performance tuning on distributed environments | 493 |
| 14.1 Performance tuning overview | 494 |
| 14.2 Using the queue analogy to tune WebSphere resource pools | 494 |
| 14.2.1 Upstream queuing | 496 |
| 14.2.2 Data source tuning | 497 |
| 14.2.3 EJB container | 499 |
| 14.2.4 Web container tuning | 500 |
| 14.2.5 Web server tuning | 501 |
| 14.2.6 Estimating web container and ORB thread pool initial sizes | 504 |
| 14.2.7 WebSphere Plug-in tuning | 504 |
| 14.3 JVM tuning | 506 |
| 14.3.1 Garbage collection | 507 |
| 14.3.2 Sizing the JVM heap | 509 |
| 14.3.3 Sizing the nursery and tenured space when using the gencon policy | 510 |
| 14.3.4 Using compressed references | 511 |
| 14.4 Other tuning considerations | 512 |
| 14.4.1 Dynamic caching | 512 |
| 14.4.2 The pass by reference parameter | 512 |
| 14.4.3 Large page support | 513 |
| 14.4.4 Application tuning | 513 |
| 14.5 Tools | 514 |
| 14.5.1 Tivoli Performance Viewer | 514 |
| 14.5.2 Collecting Java dumps and core files using the administrative console | 514 |
| 14.5.3 IBM Pattern Modelling and Analysis Tool for Java Garbage Collector | 514 |
| 14.5.4 IBM Monitoring and Diagnostic tools for Java | 515 |
| 14.5.5 IBM HTTP server status monitoring page | 515 |
| 14.5.6 WebSphere performance advisors | 516 |
| 14.6 Case Study | 517 |
| Chapter 15. Clustering, workload management, and high availability | 519 |
| 15.1 Clustering | 520 |

| | | |
|--------------------|---|------------|
| 15.1.1 | Clustering for scalability and failover. | 520 |
| 15.1.2 | Intelligent Management. | 521 |
| 15.1.3 | Dynamic cluster | 522 |
| 15.1.4 | Static cluster versus dynamic cluster | 523 |
| 15.1.5 | Creating a static application server cluster | 524 |
| 15.1.6 | Creating a dynamic application server cluster | 527 |
| 15.1.7 | Setting the operational mode for dynamic clusters | 532 |
| 15.2 | Workload management | 532 |
| 15.2.1 | Dynamic workload management. | 533 |
| 15.2.2 | Components that can be workload managed | 533 |
| 15.2.3 | Workload management benefits | 538 |
| 15.3 | High availability and failover | 538 |
| 15.3.1 | Overview | 538 |
| 15.3.2 | WebSphere Application Server high availability and failover | 539 |
| 15.3.3 | How high availability features work. | 545 |
| 15.4 | ODR server considerations. | 549 |
| 15.4.1 | Web server plug-in when using the ODR server. | 551 |
| 15.4.2 | Configuring the ODR proxy plug-in configuration policy | 551 |
| Chapter 16. | Monitoring distributed systems | 553 |
| 16.1 | Overview | 554 |
| 16.2 | Enabling monitoring infrastructures. | 555 |
| 16.2.1 | PMI defaults and monitoring settings | 555 |
| 16.2.2 | Enabling request metrics | 562 |
| 16.3 | Viewing the monitoring data | 567 |
| 16.3.1 | Starting TPV monitoring and configuring settings. | 567 |
| 16.3.2 | Exploring Tivoli Performance Viewer data views | 571 |
| 16.4 | Monitoring examples | 575 |
| 16.4.1 | JVM memory and CPU usage. | 576 |
| 16.4.2 | Threading resources | 578 |
| 16.4.3 | Database interactions | 580 |
| 16.4.4 | Request level details. | 581 |
| 16.5 | Monitoring operations | 584 |
| 16.5.1 | Runtime operations overview | 585 |
| 16.5.2 | Creating and managing reports | 586 |
| 16.5.3 | Configuring the visualization data service. | 588 |
| 16.5.4 | Task management | 589 |
| 16.5.5 | Managing runtime tasks | 589 |
| 16.6 | IBM Tivoli Composite Application Manager for WebSphere Application Server | 591 |
| 16.6.1 | Installing the data collector | 591 |
| 16.6.2 | Configuring Tivoli Composite Application Manager for WebSphere metrics. | 591 |
| 16.6.3 | Viewing IBM Tivoli Composite Application Manager for WebSphere data | 593 |
| 16.7 | Additional resources for monitoring. | 595 |
| 16.7.1 | Java dump and core files | 595 |
| 16.7.2 | Basic logging. | 596 |
| 16.7.3 | Advanced logging | 596 |
| 16.7.4 | Operating system monitoring | 598 |
| 16.7.5 | Summary of monitoring tips | 598 |

Part 4. Managing z/OS systems 599

| | | |
|--------------------|--|------------|
| Chapter 17. | Performance tuning | 601 |
| 17.1 | Introduction to WebSphere Application Server for z/OS V8.5 performance | 602 |
| 17.2 | External factors and z/OS specifics | 602 |

| | | |
|--------------------|---|------------|
| 17.2.1 | Getting the most benefit from collocation | 603 |
| 17.2.2 | Addressing hardware configuration. | 603 |
| 17.2.3 | z/OS tuning tips. | 603 |
| 17.3 | Performance tuning templates | 605 |
| 17.4 | 64-bit considerations | 607 |
| 17.4.1 | Enabling 64-bit mode | 607 |
| 17.4.2 | Effects of switching to 64-bit mode | 608 |
| 17.5 | JVM tuning | 613 |
| 17.5.1 | Default garbage collection | 613 |
| 17.5.2 | General JVM suggestions. | 613 |
| 17.6 | Connection pool tuning | 618 |
| 17.7 | Runtime provisioning. | 618 |
| 17.8 | Pass by reference | 619 |
| 17.9 | Logging and tracing. | 620 |
| 17.9.1 | High Performance Extensible Logging overview. | 620 |
| 17.9.2 | Enabling HPEL mode | 620 |
| 17.9.3 | z/OS logging and tracing tips | 620 |
| 17.10 | Tuning workload management on z/OS systems | 624 |
| 17.10.1 | The concept of workload management on z/OS systems. | 624 |
| 17.10.2 | Classification rules | 625 |
| 17.10.3 | Classification XML | 626 |
| 17.10.4 | Commands and tools | 627 |
| 17.11 | Fast response cache accelerator and caching | 628 |
| 17.11.1 | FRCA overview | 629 |
| 17.11.2 | Enabling FRCA in WebSphere Application Server | 629 |
| 17.11.3 | Cache specification XML | 636 |
| 17.11.4 | FRCA and RACF integration. | 637 |
| 17.11.5 | Caching enhancements in WebSphere Application Server V8.5 | 637 |
| 17.11.6 | Using IBM Extended Dynamic Cache Monitor to supervise caching | 637 |
| 17.12 | Using WebSphere for z/OS Optimized Local Adapters. | 638 |
| 17.12.1 | Introduction to Optimized Local Adapters. | 638 |
| 17.12.2 | Enabling WebSphere for z/OS Optimized Local Adapters | 640 |
| 17.13 | IBM HTTP Server Status monitoring page | 643 |
| 17.14 | Tools | 643 |
| Chapter 18. | Clustering and high availability. | 645 |
| 18.1 | Clustering on z/OS systems | 646 |
| 18.1.1 | Clustering for scalability and failover. | 646 |
| 18.1.2 | Creating a cluster on a z/OS system. | 646 |
| 18.2 | High availability | 650 |
| 18.2.1 | High availability manager | 650 |
| 18.2.2 | Core groups | 652 |
| 18.2.3 | High-availability policies and groups. | 671 |
| 18.3 | Failover and failback | 674 |
| 18.3.1 | High availability and failover of singletons | 674 |
| 18.3.2 | Data replication domains | 685 |
| 18.3.3 | Session management replication | 687 |
| 18.3.4 | EJB stateful session bean replication | 688 |
| 18.3.5 | Cache replication | 692 |
| 18.3.6 | Resource workload routing | 693 |
| 18.3.7 | High-availability application update rollout | 697 |
| 18.4 | Enabling multiple servants | 700 |
| 18.4.1 | Balancing workload with WLM | 702 |

| | | |
|--------------------|---|------------|
| 18.4.2 | Balancing workload without WLM | 702 |
| 18.5 | Additional resources | 703 |
| Chapter 19. | Monitoring z/OS systems | 705 |
| 19.1 | Overview | 706 |
| 19.2 | Monitoring from the administrative console | 707 |
| 19.2.1 | PMI Monitoring | 707 |
| 19.2.2 | Monitoring Dynamic Caching | 708 |
| 19.2.3 | Monitoring web services through PMI | 708 |
| 19.3 | IBM Tivoli Composite Application Manager for WebSphere Application Server | 709 |
| 19.3.1 | Installing the data collector | 710 |
| 19.3.2 | Configuring Tivoli Composite Application Manager for WebSphere metrics | 710 |
| 19.3.3 | Viewing IBM Tivoli Composite Application Manager for WebSphere data | 720 |
| 19.4 | Additional resources for monitoring | 720 |
| 19.4.1 | IBM Support Assistant | 720 |
| 19.4.2 | Verbose garbage collection | 720 |
| 19.4.3 | Java dump and core files | 723 |
| 19.4.4 | Basic logging | 724 |
| 19.4.5 | Advanced logging | 725 |
| 19.4.6 | z/OS monitoring | 731 |
| 19.4.7 | Summary of monitoring tips | 735 |
| Part 5. | Working with applications | 737 |
| Chapter 20. | Features for application development and deployment | 739 |
| 20.1 | Java Enterprise Edition 6 support | 740 |
| 20.2 | Integrated standards-base programming models and extensions | 741 |
| 20.2.1 | Session Initiation Protocol applications | 741 |
| 20.2.2 | WebSphere Batch programming model | 742 |
| 20.2.3 | OSGi applications programming model | 744 |
| 20.2.4 | Communications enabled applications | 745 |
| 20.2.5 | Service Component Architecture programming model | 746 |
| 20.2.6 | Extensible Markup Language programming model | 747 |
| 20.2.7 | Integrated Web Services support | 747 |
| 20.2.8 | Support for integrated IBM WebSphere Application Server Web 2.0 and Mobile Toolkit | 747 |
| 20.2.9 | Simplified development of server-side REST applications using Java API for RESTful Web Services | 748 |
| 20.2.10 | IBM WebSphere SDK Java Technology Edition Version 7.0 | 748 |
| 20.3 | Monitored directory support | 748 |
| 20.4 | Development and deployment tools | 748 |
| 20.4.1 | IBM Assembly and Deploy Tools for WebSphere Administration | 748 |
| 20.4.2 | WebSphere Application Server Developer Tools for Eclipse | 749 |
| 20.4.3 | Rational Application Developer for WebSphere Software | 749 |
| Chapter 21. | WebSphere Batch | 751 |
| 21.1 | Overview of WebSphere Batch | 752 |
| 21.1.1 | Batch jobs | 752 |
| 21.1.2 | Batch applications | 752 |
| 21.1.3 | Elements of the batch environment | 753 |
| 21.2 | Batch programming models | 755 |
| 21.2.1 | Transactional batch programming model | 755 |
| 21.2.2 | Compute-intensive programming model | 761 |
| 21.2.3 | Parallel batch | 762 |

| | | |
|---|--|------------|
| 21.2.4 | COBOL support | 763 |
| 21.2.5 | Batch toolkit | 764 |
| 21.3 | Configuring the batch environment | 765 |
| 21.3.1 | Configuring the job scheduler | 765 |
| 21.3.2 | Securing the job scheduler | 766 |
| 21.3.3 | Job scheduler integration with external schedulers | 767 |
| 21.3.4 | Configuring grid endpoints | 770 |
| 21.3.5 | Configuring the job scheduler and job management console | 770 |
| 21.3.6 | Command-line interface for batch jobs | 771 |
| 21.3.7 | Job logs | 772 |
| 21.3.8 | Job classes | 773 |
| 21.4 | Example: Working with batch applications | 774 |
| 21.4.1 | Enabling the job scheduler | 774 |
| 21.4.2 | Verifying the job scheduler installation | 775 |
| 21.4.3 | Installing the sample batch application | 776 |
| 21.4.4 | Securing the job scheduler using Job groups | 777 |
| 21.4.5 | Using the job management console | 780 |
| 21.4.6 | Using the command-line interface for batch jobs | 784 |
| 21.4.7 | Checking the batch job logs | 785 |
| Chapter 22. Understanding class loaders | | 789 |
| 22.1 | JVM class loaders | 790 |
| 22.2 | WebSphere Application Server and Java EE application class loaders | 791 |
| 22.2.1 | WebSphere extensions class loader | 792 |
| 22.2.2 | Application and web module class loaders | 793 |
| 22.2.3 | Handling Java Native Interface code | 794 |
| 22.3 | Configuring class loaders for Java EE applications | 795 |
| 22.3.1 | Application server class loader policies | 795 |
| 22.3.2 | Class loading and delegation mode | 797 |
| 22.3.3 | Shared libraries | 798 |
| 22.3.4 | Class loader viewer | 799 |
| 22.4 | Learning class loaders for Java EE by example | 800 |
| 22.4.1 | Example 1: Simple web module packaging | 800 |
| 22.4.2 | Example 2: Adding an EJB module and utility jar | 803 |
| 22.4.3 | Example 3: Changing the WAR class loader delegation mode | 804 |
| 22.4.4 | Example 4: Sharing utility JAR files using shared libraries | 805 |
| 22.5 | OSGi class loaders | 810 |
| Chapter 23. Packaging and deploying Java EE applications | | 813 |
| 23.1 | Java EE applications | 814 |
| 23.1.1 | Java EE 6 EAR files | 814 |
| 23.1.2 | Development tools | 816 |
| 23.1.3 | Packaging enterprise applications | 817 |
| 23.1.4 | Packaging EJB 3.1 modules | 820 |
| 23.1.5 | Packaging JPA persistence units | 823 |
| 23.1.6 | JPA access intent | 823 |
| 23.1.7 | Packaging resource adapters | 824 |
| 23.1.8 | Packaging Web modules | 824 |
| 23.1.9 | Packaging EJB 3.1 content in Web modules | 829 |
| 23.2 | Preparing to use the sample application | 830 |
| 23.2.1 | Downloading the application | 830 |
| 23.2.2 | Importing the application to the development tool | 830 |
| 23.2.3 | Customizing the sample application | 831 |

| | | |
|--------------------|---|------------|
| 23.2.4 | Creating the ITSO Bank DB2 database | 832 |
| 23.2.5 | Configuring web module extensions | 833 |
| 23.3 | Packaging recommendations | 834 |
| 23.4 | Creating WebSphere-enhanced EAR files | 835 |
| 23.4.1 | Configuring a WebSphere enhanced EAR | 835 |
| 23.4.2 | Configuring application options | 836 |
| 23.4.3 | Configuring the JDBC provider and data source for DB2 | 837 |
| 23.4.4 | Configuring substitution variables | 843 |
| 23.4.5 | Configuring a virtual host | 843 |
| 23.4.6 | Setting the default virtual host for web modules | 844 |
| 23.4.7 | Examining the WebSphere-enhanced EAR file | 844 |
| 23.5 | Exporting an application project to an EAR file | 845 |
| 23.6 | Preparing the runtime environment for the application | 846 |
| 23.6.1 | Creating an environment variable for the application file directory | 847 |
| 23.6.2 | Creating the ITSO Bank application server | 847 |
| 23.6.3 | Defining the ITSO Bank virtual host | 851 |
| 23.6.4 | Creating the virtual host for the IBM HTTP Server | 852 |
| 23.6.5 | Creating a DB2 JDBC provider and data source | 853 |
| 23.7 | Deploying the application | 855 |
| 23.7.1 | Deploying using the administrative console | 855 |
| 23.7.2 | Deploying using the monitored directory support feature | 860 |
| 23.7.3 | Deploying applications using the job manager | 866 |
| 23.8 | Deploying business-level applications | 868 |
| 23.8.1 | Creating a business-level application | 871 |
| 23.9 | Deploying application clients | 874 |
| 23.9.1 | Installing application client environments | 875 |
| 23.9.2 | Preparing the sample application | 875 |
| 23.9.3 | Launching the J2EE client | 876 |
| Chapter 24. | Updating Java EE applications | 879 |
| 24.1 | Working with applications | 880 |
| 24.2 | Replacing an entire application EAR file | 880 |
| 24.3 | Replacing or adding an application module | 882 |
| 24.3.1 | Replacing or adding single files in an application or module | 883 |
| 24.3.2 | Removing application content | 883 |
| 24.3.3 | Performing multiple updates to an application or module | 884 |
| 24.3.4 | Rolling out application updates to a cluster | 886 |
| 24.4 | Application edition management and rollout | 889 |
| 24.4.1 | Installing an application edition | 889 |
| 24.4.2 | Activating an edition | 890 |
| 24.4.3 | Creating routing policies for application editions | 891 |
| 24.4.4 | Validating an edition | 892 |
| 24.4.5 | Rolling out an edition | 894 |
| 24.4.6 | Rolling back an edition | 898 |
| 24.5 | Hot deployment and dynamic reloading | 899 |
| Chapter 25. | Working with SCA applications | 901 |
| 25.1 | SCA application introduction | 902 |
| 25.1.1 | SCA component | 903 |
| 25.1.2 | SCA composite | 903 |
| 25.1.3 | SCA contribution | 905 |
| 25.2 | Preparing to use the sample application | 906 |
| 25.2.1 | Downloading the application | 906 |

| | | |
|---|--|------------|
| 25.2.2 | Importing the application to the development tool. | 907 |
| 25.2.3 | Completing the service definition | 907 |
| 25.3 | Packaging an SCA application for deployment. | 908 |
| 25.3.1 | Creating the contribution. | 909 |
| 25.3.2 | Exporting the SCA application for deployment | 911 |
| 25.4 | Deploying an SCA application. | 912 |
| 25.4.1 | Importing the SCA archive file as an asset. | 912 |
| 25.4.2 | Creating a new business-level application | 915 |
| 25.4.3 | Adding the new asset to the business-level application | 916 |
| 25.4.4 | Starting and verifying the business-level application | 919 |
| 25.5 | Additional resources for learning. | 919 |
| Chapter 26. Working with OSGi applications | | 921 |
| 26.1 | OSGi overview | 922 |
| 26.1.1 | OSGi application model | 922 |
| 26.1.2 | OSGi bundle lifecycle | 926 |
| 26.1.3 | OSGi Service | 928 |
| 26.2 | Enterprise OSGi | 928 |
| 26.3 | Using the sample application | 929 |
| 26.3.1 | Downloading the application | 929 |
| 26.3.2 | Importing the application to the development tool. | 930 |
| 26.4 | Packaging OSGi applications | 932 |
| 26.4.1 | Common OSGi patterns | 933 |
| 26.4.2 | Sample application packaging | 933 |
| 26.4.3 | Exporting OSGi applications | 935 |
| 26.5 | Deploying OSGi applications | 936 |
| 26.5.1 | Importing the enterprise bundle archive file as an asset. | 936 |
| 26.5.2 | Adding the enterprise bundle archive asset to the business-level application | 937 |
| 26.6 | Administrating OSGi applications | 940 |
| 26.6.1 | Updating OSGi applications | 940 |
| 26.6.2 | Securing OSGi applications | 944 |
| Chapter 27. Working with Service Mapping | | 945 |
| 27.1 | Service mapping overview | 946 |
| 27.1.1 | Service maps | 947 |
| 27.2 | Local mapping service | 950 |
| 27.2.1 | Creating a local mapping service | 950 |
| 27.2.2 | Starting and stopping a local mapping service | 952 |
| 27.3 | Administration for target service clients | 952 |
| 27.3.1 | Policy sets and bindings | 955 |
| 27.3.2 | Override target service URLs | 955 |
| 27.4 | Event emissions | 957 |
| 27.4.1 | Schema explanation | 958 |
| 27.5 | Securing a service map. | 959 |
| Chapter 28. Session management. | | 961 |
| 28.1 | Session overview | 962 |
| 28.1.1 | Session identifiers. | 962 |
| 28.1.2 | Session invalidation | 964 |
| 28.1.3 | Session listeners. | 964 |
| 28.1.4 | Session security | 965 |
| 28.2 | Session management configuration | 966 |
| 28.2.1 | Session management properties | 966 |
| 28.2.2 | Accessing session management properties | 967 |

| | | |
|--------------------|---|-------------|
| 28.2.3 | Selecting session tracking options | 969 |
| 28.2.4 | Scheduled invalidation configuration | 969 |
| 28.2.5 | Cookie setting | 970 |
| 28.3 | Storing session information. | 972 |
| 28.3.1 | Local sessions | 972 |
| 28.3.2 | Persistent sessions management | 973 |
| 28.3.3 | Enabling database persistence | 974 |
| 28.3.4 | Memory-to-memory replication | 976 |
| 28.4 | Session affinity | 983 |
| 28.4.1 | What is the session affinity | 983 |
| 28.4.2 | Session affinity and failover | 984 |
| 28.5 | Session management tuning. | 986 |
| 28.5.1 | Session performance considerations | 986 |
| 28.5.2 | Session management tuning | 987 |
| 28.5.3 | Session database tuning. | 994 |
| 28.6 | Stateful session bean failover | 995 |
| 28.6.1 | Enabling stateful session bean failover. | 995 |
| 28.6.2 | Stateful session bean failover consideration. | 998 |
| Part 6. | Maintenance | 999 |
| Chapter 29. | Managing an environment with the centralized installation manager. | 1001 |
| 29.1 | The centralized installation manager prerequisites | 1002 |
| 29.1.1 | Linux and UNIX target requirements. | 1002 |
| 29.1.2 | Windows target requirements | 1003 |
| 29.1.3 | IBM i targets | 1003 |
| 29.1.4 | Additional requirements | 1003 |
| 29.2 | Planning considerations | 1004 |
| 29.2.1 | WebSphere Application Server V8 releases. | 1004 |
| 29.2.2 | WebSphere Application Server V6.1 and V7 | 1005 |
| 29.3 | Using centralized installation manager with V8 releases | 1005 |
| 29.3.1 | Installation Manager | 1005 |
| 29.3.2 | Accessing the centralized installation manager | 1007 |
| 29.4 | Using centralized installation manager with prior releases | 1007 |
| 29.4.1 | IBM Update Installer | 1008 |
| 29.4.2 | The centralized installation manager repository structure. | 1008 |
| 29.4.3 | Package types | 1009 |
| 29.4.4 | Accessing the central installation manager. | 1010 |
| 29.5 | Managing V8 release environments with the centralized installation manager. | 1012 |
| 29.5.1 | Adding new targets | 1012 |
| 29.5.2 | Installing Installation Manager on remote targets | 1016 |
| 29.5.3 | Installing a Secure Shell (SSH) public key | 1021 |
| 29.5.4 | Installing WebSphere Application Server binaries on remote hosts | 1022 |
| 29.5.5 | Creating a WebSphere Application Server profile on a remote target | 1024 |
| 29.5.6 | Registering and unregistering the profile with the Job Manager. | 1027 |
| 29.5.7 | Working with remote targets | 1029 |
| 29.5.8 | Installing maintenance to remote targets | 1034 |
| 29.5.9 | Using the centralized installation manager with a command line | 1035 |
| 29.6 | Managing V6.1 and V7 with the centralized installation manager. | 1036 |
| 29.6.1 | Installing the IBM Installation Factory | 1037 |
| 29.6.2 | Creating the centralized installation manager repository | 1037 |
| 29.6.3 | Adding packages when deployment manager is connected to the Internet | 1038 |
| 29.6.4 | Adding packages when the deployment manager does not have access to the | |

| | |
|---|-------------|
| Internet | 1043 |
| 29.6.5 Adding and removing additional installation targets | 1045 |
| 29.6.6 Installing a Secure Shell public key. | 1046 |
| 29.6.7 Installing packages to the target systems. | 1048 |
| 29.6.8 Installing a software package | 1049 |
| 29.6.9 Installing maintenance to a target system. | 1051 |
| 29.6.10 Uninstalling packages. | 1053 |
| 29.6.11 The centralized installation manager AdminTask commands. | 1053 |
| Chapter 30. System recovery | 1055 |
| 30.1 Overview | 1056 |
| 30.2 Configuring for backup and restore | 1056 |
| 30.2.1 Backing up a profile configuration. | 1056 |
| 30.2.2 Restoring a profile configuration | 1057 |
| 30.2.3 Exporting and importing a configuration archive | 1059 |
| 30.3 Configuring checkpoints service | 1061 |
| 30.3.1 Creating repository checkpoints | 1062 |
| 30.3.2 Archiving or deleting checkpoints | 1063 |
| 30.3.3 Restoring checkpoints. | 1064 |
| 30.3.4 Configuring change audit | 1064 |
| 30.4 Restoring transactions | 1064 |
| 30.4.1 Restarting an application server in recovery mode. | 1065 |
| 30.4.2 Administering the transaction service | 1065 |
| 30.4.3 Transactional high availability | 1066 |
| 30.5 Recovery node with addNode -asExistingNode command | 1067 |
| 30.5.1 Considerations when using the -asExistingNode command | 1067 |
| 30.5.2 Recovering a failed managed node of deployment manager | 1067 |
| 30.5.3 Moving a node to a different system | 1069 |
| 30.5.4 Recreating a cell from a template | 1073 |
| Chapter 31. Troubleshooting | 1075 |
| 31.1 Overview | 1076 |
| 31.2 WebSphere Application Server logs | 1076 |
| 31.2.1 Server log files | 1077 |
| 31.2.2 JVM log interpretation. | 1079 |
| 31.2.3 Logging modes | 1082 |
| 31.2.4 High Performance Extensible Logging | 1083 |
| 31.2.5 Cross Component Trace. | 1091 |
| 31.2.6 Sensitive log and trace guard | 1096 |
| 31.2.7 Javacores and Heapdumps | 1096 |
| 31.2.8 HTTP Plug-in Log | 1097 |
| 31.3 Tools for collecting and analyzing diagnostic data | 1097 |
| 31.3.1 Hang detection policy | 1097 |
| 31.3.2 Memory leak detection policy | 1099 |
| 31.3.3 MustGather for troubleshooting | 1100 |
| 31.3.4 IBM Support Assistant | 1101 |
| 31.4 Troubleshooting scenarios | 1103 |
| 31.4.1 Hung threads | 1103 |
| 31.4.2 High CPU | 1106 |
| 31.4.3 Out of Memory exceptions in WebSphere Application Server | 1109 |
| Appendix A. Additional material | 1115 |
| Locating the web material | 1115 |
| Using the web material. | 1115 |

| | |
|---|------|
| Downloading and extracting the Web material | 1116 |
| Related publications | 1117 |
| IBM Redbooks | 1117 |
| Online resources | 1117 |
| Help from IBM | 1120 |

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.


COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

| | | |
|-----------------------------|---|-----------------|
| AIX® | Language Environment® | RMF™ |
| CICS® | Lotus® | System z10® |
| ClearCase® | MVS™ | System z9® |
| DataPower® | OS/400® | System z® |
| DB2® | Parallel Sysplex® | SystemPac® |
| developerWorks® | Passport Advantage® | Tivoli® |
| Domino® | POWER® | Velocity™ |
| eServer™ | RACF® | VTAM® |
| Global Technology Services® | Rational Team Concert™ | WebSphere® |
| GPFS™ | Rational® | z/Architecture® |
| i5/OS™ | Redbooks® | z/OS® |
| IBM® | Redpapers™ | z10™ |
| IMS™ | Redbooks (logo)  ® | z9® |
| Informix® | Resource Measurement Facility™ | zSeries® |

The following terms are trademarks of other companies:

ITIL is a registered trademark, and a registered community trademark of The Minister for the Cabinet Office, and is registered in the U.S. Patent and Trademark Office.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication provides system administrators and developers with the knowledge to configure an IBM WebSphere® Application Server Version 8.5 runtime environment, to package and deploy applications, and to perform ongoing management of the WebSphere environment. As one in a series of IBM Redbooks publications and IBM Redpapers™ publications for V8.5, the entire series is designed to give you in-depth information about key WebSphere Application Server features.

WebSphere Application Server V8.5 provides two runtime profiles. Every WebSphere Application Server package includes both profile types. The run time traditionally available with the WebSphere Application Server packages is referred to as the *full profile*. The application serving run time provided with this profile is composed of a wide spectrum of runtime components that are available when the server is started. The full profile provides support for Java Platform Enterprise Edition 6 (Java EE 6) and Enterprise OSGi technologies.

The *Liberty profile* provides a simplified stand-alone run time for web applications, supporting a subset of the programming model available with the full profile. Any application that runs on the Liberty profile will also run on the full profile.

In this book, we provide a detailed exploration of the WebSphere Application Server V8.5 runtime administration process for the *full profile*. This book includes configuration and administration information for WebSphere Application Server V8.5 and WebSphere Application Server Network Deployment V8.5 on distributed platforms and WebSphere Application Server for IBM z/OS® V8.5.

This book has been updated with information about the new features in WebSphere Application Server V8.5.5. The Liberty profile administration and configuration information has been moved into a separate book.

The following publications are prerequisites for this book:

- ▶ *WebSphere Application Server: New Features in V8.5.5*, REDP-4870
- ▶ *WebSphere Application Server V8.5.5 Technical Overview*, REDP-4855
- ▶ *IBM WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022

The following publications are companion books, covering the Liberty profile of WebSphere Application Server:

- ▶ *WebSphere Application Server Liberty Profile Guide for Developers*, SG24-8076
- ▶ *WebSphere Application Server V8.5 Administration Guide for the Liberty Profile*, SG24-8170

Authors

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Fabio Albertoni is a Senior IT Specialist working in Integrated Technology Delivery SSO in Hortolandia, Brazil. He has sixteen years of experience in the IT and banking industries. He

has spent the last twelve years developing and implementing integrated solutions using WebSphere Application Server and MQ Series. He holds a degree in Data Processing from FATEC University of Ourinhos and a Master's degree in Computer Engineering from Instituto de Pesquisas Tecnológicas of Sao Paulo, Brazil.

Tanja Baumann is a co-operative student in Germany. She has worked for IBM for two years. She is currently working on a B.Sc. in Computer Science at the University of Corporate Education, Mannheim. Tanja participated in this IBM Redbooks publication residency during an internship.

Yogesh Bhatia is a subject matter expert on middleware technologies and handling middleware operations for all IBM India domestic accounts. He has rich experience in all IBM middleware products and leading middleware competency in the Central Data Center of major telecoms in India, which is one of the biggest Data Centers in Asia and the most dynamic and growing environment. He is responsible for managing the entire WebSphere and IBM Tivoli® family of products. He is an IBM certified WebSphere professional with numerous other vendor certifications on EAI, UML, and Object-Oriented Concepts. He has deep production environment experience, which includes expert-level troubleshooting on different products, architecture finalization, deployment, performance tuning, and providing end-to-end support for middleware.

Eduardo Monich Fronza is a Level 2 Certified IT Specialist working for IBM Global Technology Services® in Brazil. He has eight years of experience in supporting IBM WebSphere Application Server and IBM WebSphere Portal Server. His main areas of expertise are automation for WebSphere administration, infrastructure design, implementation, and maintenance and problem determination of the WebSphere environment. Eduardo is also an IBM Certified System Administrator for WebSphere Application Server, WebSphere Portal Server, and Service-Oriented Architecture (SOA) Solutions. He holds a Bachelor's degree in Computer Science from Universidade do Estado de Santa Catarina.

Marcio da Ros Gomes is an IBM Certified IT Specialist at IBM Brazil. Marcio holds a Bachelor's degree in Computer Science from Universidade Federal do Espírito Santo. He has more than 10 years of IT experience in middleware, network security, open source tools, UNIX operating systems, and web hosting environments. He has designed, implemented, and supported various middleware infrastructure and web hosting environments in large public and private organizations. Mr. Gomes is certified for Linux Professional Institute Certified Level 1, Oracle BEA WebLogic Application Server 9, IBM Certified System Administrator for WebSphere Application Server ND 6.1, IBM Certified SOA Associate, and ITIL Foundations Certified.

Sebastian Kapciak is an Advisory IT Specialist working for IBM Global Technology Services in Warsaw, Poland. Sebastian joined IBM in 2007 and has over eight years of experience in software architecture and development. His areas of expertise are system integration and JEE technologies. He specializes in the WebSphere Application Server (for which he actively contributes to IBM Redbooks publication projects), IBM DataPower® appliances, and IBM Tivoli Access Manager. Sebastian holds a Master's degree in Information Technology from the University of Technology of Warsaw.

Catalin Mierlea is a Middleware Software Specialist in IBM Romania. Catalin joined IBM in March 2012 and has 10 years of experience in IT. His areas of expertise include WebSphere products, SOA, and software architecture. He specializes in the WebSphere Application Server, WebSphere Portal Server, WebSphere Process Server, and WebSphere Business Process Manager. Catalin has a Bachelor of Science degree in Automation Control and Computers, a Master of Science degree in Integrated Informatics Systems, and certifications in the IBM WebSphere products and in several Microsoft and Oracle complementary

technologies. He has extensive industry knowledge and hands-on project experience in the banking sector.

Sergio Pinto is an IT Specialist working for Integrated Technology Delivery, Server Systems Operations, in Brazil. He has worked for IBM for 17 years. His areas of expertise include support in WebSphere Application Server for z/OS and support in WebSphere MQ and WebSphere Message Broker on distributed and z/OS environments. He holds a Bachelor's degree in Business Administration and Accounting from Instituto Católico de Minas Gerais, Brazil.

Anoop Ramachandra is a Senior Staff Software Engineer in the IBM India Software Labs. He has over eight years of experience working with WebSphere Application Server products as a Technical Lead, Developer, and Level 3 Support Engineer. His major areas of expertise in WebSphere Application Server are system management, Java EE Connector Architecture, Virtual Member Manager, Scheduler, and Asynchronous beans. He is an IBM Certified System Administrator for WebSphere Application Server.

Liang Rui is a Staff Software Engineer working for the China Development Lab in Beijing, China. Ray has six years of experience as a developer and tester in IT and worked four years with IBM WebSphere Enterprise Service Bus, IBM WebSphere Process Server, and IBM Business Process Management (BPM) product development and support. Ray's focus areas are configuration, integration capability, and cloud provision for BPM and WebSphere Application Server. Ray is an IBM Certified Administrator for WebSphere Portal Server V6.1 and WebSphere Application Server V6.1. He holds a Master's degree of Communication and Information Systems from Beijing University of Post and Telecommunication, China.

Miguel Troncoso is a WebSphere Solutions Architect in Software Group in IBM Mexico. He has 10 years of experience with WebSphere Application Server. Previously, Miguel worked as a Java EE Developer, mainly for the banking industry. He holds a Bachelor's degree in Computer Engineering from the National Autonomous University of Mexico (UNAM) and completed the studies for the Master's degree in Computer Engineering in the same university. Miguel is certified in WebSphere Application Server ND administration for V6.1 and V7.

Thanks to the following people for their contributions to this project:

Margaret Ticknor, Carla Sadler, Deana Coble, Tamikia Lee, Linda Robinson, Shawn Tooley, KaTrina Love
International Technical Support Organization, Raleigh Center

Mehrdad Ashrafian, Soloman Barghouthi, Dave Cohen, David Follis, Alex Mulholland, Sajan Sankaran, Keith B. Smith, Erin Schnabel, Hendriz Tavaréz
IBM US

Alasdair Nottingham, James Mullineux
IBM UK

Felix Wong
IBM Canada

Thanks to the authors of the previous editions of this book:

- Authors of the first editions of this book, *WebSphere Application Server V6.1: System Management and Configuration*, SG24-7304, published in November 2006:
Carla Sadler, Fabio Albertoni, Bernardo Fagalde, Thiago Kleinubing, Henrik Sjostrand, Ken Worland, Lars Bek Laursen, Martin Phillips, Martin Smithson, and Kwan-Ming Wan.

- ▶ Authors of the second edition, *WebSphere Application Server V7: Administration and Configuration Guide*, SG24-7615, published in March 2010:
Fabio Albertoni, Leonard Blunt, Michael Connolly, Stefan Kwiatkowski, Carla Sadtler, Thayaparan Shanmugaratnam, Henrik Sjostrand, Saori Tanikawa, Margaret Ticknor, and Joerg-Ulrich Veser
- ▶ Authors of the third edition, *WebSphere application Server V8: Administration and Configuration Guide*, SG24-7971, published in November 2011:
Martin Bentancour, Libor Cada, Marcio d’Amico, Ural Emekci, Sebastian Kapiak, Jennifer Ricciuti, Margaret Ticknor

Now you can become a published author, too!

Here’s an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:

ibm.com/redbooks

- ▶ Send your comments in an email to:

redbooks@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- ▶ Find us on Facebook:
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on Twitter:
<http://twitter.com/ibmredbooks>

- ▶ Look for us on LinkedIn:
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:
<http://www.redbooks.ibm.com/rss.html>



Part 1

Installation and profile management



System management: Technical overview

This chapter provides a technical overview of the system management functionality of WebSphere Application Server *full profile*. It explains the many system management capabilities and tools that make WebSphere Application Server so useful, including new features that are available in WebSphere Application Server V8.5 and V8.5.5.

In this chapter, we cover the following topics:

- ▶ System management overview
- ▶ New features for administrators
- ▶ Java Management Extensions
- ▶ System management in a stand-alone server environment
- ▶ System management in a distributed server environment
- ▶ Advanced system management of multiple stand-alone servers
- ▶ Advanced management of distributed and stand-alone servers

1.1 WebSphere Application Server profiles

WebSphere Application Server V8.5 provides two runtime profiles. Every WebSphere Application Server package includes both profile types.

The run time traditionally available with the WebSphere Application Server packages is referred to as the *full profile*. The application serving run time (*application server*) provided with this profile is composed of a wide spectrum of runtime components that are always available when the server is started. The full profile provides support for Java Platform Enterprise Edition 6 (Java EE 6) and Enterprise OSGi technologies. In addition to the application servers, the full profile supports the creation of generic server definitions to configure other servers or processes that are necessary to support the application server environment. Additional capabilities, such as clustering application servers for load balancing and high availability, vary depending on the WebSphere Application Server package.

In addition to the full profile, a new *Liberty profile* is included with each package and **(New in V8.5.5)** the Liberty profile is also available as a stand-alone offering, called *WebSphere Application Server Liberty Core*. The Liberty profile provides a simplified stand-alone run time for web applications, supporting a subset of the programming model available with the full profile. It is a good option for developers who are building web applications that do not require the full Java EE environment of traditional enterprise application server profiles. Any application that runs on the Liberty profile will also run on the full profile. The application-serving environment is configured with the correct level of capabilities that are required for the individual applications. You can use the Liberty profile to specify only those features that are required for the applications that are deployed, reducing the memory footprint and increasing performance. The Liberty profile has a simplified installation and uses an easy-to-configure XML configuration file format. The Liberty profile is optimized for use in both development and production environments. Within the development environment, the Liberty profile supports the same platforms as the base application server and Mac OS. Enterprise qualities of service (QoS), such as security and transaction integrity, are enabled as required.

Many new enhancements have been made to the Liberty profile with WebSphere Application Server V8.5.5. These enhancements include additional programming model support, new security and troubleshooting features, and new options for managing servers, including the option to cluster servers for availability and scalability.

This book focuses on the full profile. For more information about the Liberty profile, see these books:

- ▶ *WebSphere Application Server Liberty Profile Guide for Developers*, SG24-8076
- ▶ *WebSphere Application Server V8.5 Administration Guide for the Liberty Profile*, SG24-8170

1.2 System management overview

WebSphere Application Server V8.5 provides easy-to-use administration tools and powerful features to make system management simple to understand and operate. The system management functionality of WebSphere Application Server is based on the use of Java Management Extensions (JMX).

1.2.1 Terminology

There are differences in how WebSphere Application Server handles administration, depending on the environment that you have set up. As you go through this Redbooks publication, you will see the following terms used:

- ▶ *Stand-alone server environment* refers to a single server that is not managed as part of a cell. (The server was not federated to the cell.) With the Base and Express offerings of WebSphere Application Server, this is your only option. The Network Deployment offering allows you to create either a distributed or a stand-alone server environment.
- ▶ *Distributed server environment* refers to multiple servers managed from a single deployment manager in the cell. These servers are also called *managed servers*. Distributed server environments are only possible with the Network Deployment offering.
- ▶ *Application server* refers to the process that provides the functions that are required to support and host user applications. An application server can be a stand-alone application server, a distributed application server that is managed by a deployment manager, or a stand-alone application server that is managed using an administrative agent.
- ▶ *Managed processes* refer to the deployment manager, nodes (node agents), and application servers.
- ▶ *Flexible management* refers to asynchronous job management through a job manager (available in Network Deployment only) and managing multiple unfederated application servers using an administrative agent (available in all offerings).

1.2.2 Directory conventions

Throughout this book, we use the following notations when indicating the location of files and commands:

- ▶ *install_root* is used to denote the installation directory for a product. The default installation directory locations are at the following website:
http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=rins_dircon
- ▶ *profile_root* denotes the home profile for a directory. This is equivalent to:
install_root/profiles/profile_name
Special instances of *profile_root* are used to denote the profile home for the following processes:
 - Deployment manager:
dmgr_profile_root
 - Administrative agent:
adminAgnt_profile_root
 - Job manager:
jmgr_profile_root

1.2.3 Core concepts of system management

The core concepts of system management are:

Profiles To create different types of WebSphere Application Server runtime environments, you must install the WebSphere Application Server

core product files and then create a set of configuration files called *profiles*.

| | |
|-----------------------------|--|
| Application server | The application server is the platform on which Java language-based applications run. |
| Node | A node is an administrative grouping of application servers for configuration and operational management within one operating system instance. |
| Deployment manager | The deployment manager is the central administration point of a cell that consists of multiple nodes and node groups in a distributed server configuration. |
| Node agent | In distributed server configurations, each node has a node agent that works with the deployment manager to manage administration processes. |
| Cell | A cell is a group of nodes in a single administrative domain. |
| Administrative agent | An administrative agent is a component that provides enhanced management capabilities for multiple stand-alone application servers. |
| Job Manager | The job manager is a component that provides management capabilities for multiple stand-alone application servers, administrative agents, and deployment managers. |

For a more detailed look at WebSphere Application Server concepts, refer to *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022.

1.2.4 System management tools

WebSphere Application Server provides a variety of administrative tools to configure and manage your runtime environment, including:

- ▶ WebSphere Customization Toolbox (WCT)

WebSphere Customization Toolbox includes tools for customizing your WebSphere Application Server environment. Among the tools are:

- Web Server Plug-in Configuration Tool
- Profile Management Tool
- z/OS Migration Tool

- ▶ Integrated Solutions Console, also called the *administrative console*

The administrative console is a browser-based client that uses a web application running in the web container to administer WebSphere Application Server. It can provide remote administration access.

- ▶ WebSphere scripting client (wsadmin)

The wsadmin client is a non-graphical scripting interface that administers WebSphere Application Server from a command-line prompt. It uses the Bean Scripting Framework (BSF), which supports a variety of scripting languages and can provide remote administration access.

- ▶ Another Neat Tool (ANT)

ANT is used for task automation. You can use it to create build scripts that compile, package, install, and test applications on WebSphere Application Server.

- ▶ Administrative applications

You can develop custom Java applications that use Java Management Extensions based on the WebSphere application programming interface (API).

► **Command-line utilities**

These administrative utilities help you manage your WebSphere Application Server environment and include the following features:

- Called from a command line
- Can be used to perform common administrative tasks, such as starting and stopping the application server and backing up the configuration

Command-line utilities work on local application servers, nodes, and the deployment manager only.

The set of administrative tools that you employ ultimately depends on the size and complexity of your runtime environment. The next sections of this publication address the multiple levels of administration in WebSphere Application Server.

1.3 New features for administrators

WebSphere Application Server Network Deployment V8.5 and V8.5.5 provide the following enhanced capabilities to extend application development and deployment:

► **Support for Java 7**

Java 6 is installed with the product and used by default. WebSphere Application Server V8.5 provides optional support for the IBM WebSphere SDK Java Technology Edition Version 7.0. This IBM software development kit (SDK) provides a full-function SDK for Java that is compliant with Java SE 7 application programming interfaces. To use Java 7, install IBM WebSphere SDK7.0 using IBM Installation Manager and then use the `managesdk` tool to enable it.

► **Comprehensive programming model support**

A wide variety of programming models are supported, providing greater flexibility and improving developer productivity. The following programming models are enhanced in WebSphere Application Server V8.5:

- Service Component Architecture (SCA): Support for several OASIS specifications
- Open Services Gateway Initiative (OSGI): Support for Enterprise JavaBean (EJB) assets in reusable bundles, plus a blueprint security update
- Web 2.0 and Mobile Toolkit support

► **WebSphere Batch**

Use the enhanced features of WebSphere Batch to build robust batch applications for performing lengthy bulk transaction processing and computation-intensive work.

► **Monitored directory**

You can speed the process of installing, debugging, updating, and uninstalling applications by dragging applications into the monitored directory. The following application file types are supported:

- Enterprise archive (EAR)
- Web archive (WAR)
- Java archive (JAR)
- Session Initiation Protocol (SIP) archive (SAR)

► **(New in V8.5.5)** Service mapping

The new service mapping feature is designed to shield applications from minor changes in the services they use. This feature gives administrators the ability to define a mapping service that can intercept service client invocations bound for a particular service. The mapping service can determine to which service location the message should be routed, which operation on the service provider should be invoked, and how the fields in the client and server messages should be mapped to each other. Administrators can control to which service interactions the service mapping applies. The mapping is created by using a graphical interface, simplifying the task.

WebSphere Application Server provides consolidated workload management, operational scaling efficiency, and high resiliency. The latest version adds the following enhanced features to help reduce operational costs and minimize the likelihood of lost business opportunities due to failure:

► Intelligent management

Intelligent management uses the on demand router (ODR) with configurable operation policies to govern the resource, performance, and health of your application server. Key features of intelligent management are:

- Application edition management
- Application server health management based on policies
- Intelligent routing
- Dynamic clustering

In WebSphere Application Server V8.5.5, a new option exists that allows administrators to implement the ODR functionality in the web server tier by enabling the WebSphere web server plug-in for Intelligent Management.

► Improved high availability

The high-availability manager provides features that allow other product components to keep operating consistently, including the following items:

- A framework that allows singleton services to remain highly available
- A mechanism that allows servers to easily exchange state data
- A specialized framework for high speed, reliable messaging between processes

► Messaging infrastructure resiliency

Message resilience is improved in the following areas:

- Recovery of messaging engine errors in high-availability environments
- Prevention of long-running database locks
- Performance of messaging bus at start-up

► Enhanced memory leak detection and protection

WebSphere Application Server V8.5 provides comprehensive, pattern-based memory leak detection, prevention, and response by watching for suspect patterns in application code at run time. This includes the following improved features:

- Ability to mitigate a memory leak when stopping applications
- Ability to prevent leaks, receive warnings, and get system dumps
- Managed beans (MBeans) to list the applications that have memory leaks

► **(New in V8.5.5)** WebSphere Application Server (base edition), WebSphere Application Server Network Deployment, and WebSphere Application Server for z/OS now include WebSphere eXtreme Scale in the package and entitlements to its use. Both the Liberty profile and the full profile can take advantage of the advanced caching abilities of WebSphere eXtreme Scale.

► Enhanced Edge capabilities

The Load Balancer for IPV4 and IPV6 provides horizontal scalability. It dispatches HTTP requests among several web server or application server nodes that support various dispatching options and algorithms to assure high availability in high volume environments. Using the Load Balancer for IPV4 and IPV6 can reduce web server congestion, increase content availability, and provide scaling ability for the web server.

(New in V8.5.5) The Load Balancer for IPV4 and IPV6 has been enhanced in V8.5.5 to improve flexibility in configuration and to improve workload balancing. The following features are new:

- The load balancer can now be run on the same machine as the servers it is balancing. This feature is supported on Linux and IBM AIX® only.
- The Content Based Routing (CBR) component has been added to enable load balancing based on the content of client requests, for example, the URI.
- The Site Selector component has been added to enable load balancing by using a domain name service (DNS) round-robin or using a user-provided algorithm.
- Network Address Translation (NAT) has been added, removing the limitation that back-end servers are on the same locally attached network.

The Edge Components also include a Load Balancer for IPV4, which is being deprecated. The primary capabilities of this load balancer are being migrated to the Load Balancer for IPV4 and IPV6.

► **(New in V8.5.5)** Serviceability and troubleshooting enhancements to Session Initiation Protocol (SIP) support enable more resilient processing of SIP sessions:

- New PMI counters at the SIP container and proxy have been added to monitor and trigger on key performance indicators (KPIs):
 - New counters for the SIP container allow you to monitor for thread and message congestion issues, the number of replicated and non-replicated SIP sessions, the number of rejected requests, and SIP timers.
 - New counters for the SIP proxy allow you to monitor queue statistics, the health of the SIP container and load balancer, and invalid SIP messages received.
- The following new troubleshooting features have been included:
 - The SIP context is now added to binary log entries for the SIP container and SIP proxy. The new information allows you to trace the flow of a SIP call through all the SIP components.
 - A new utility is provided to dump SIP application sessions and their session IDs for improved debugging of SIP container sessions. This utility can be particularly useful in production environments when fine-grained tracing cannot be enabled.
 - SIP proxy call logging now provides complete message logging, as well as logging of rejected messages.
- Application composition performance improvements have been added to allow multiple independent applications installed at a single Java virtual machine (JVM) to independently process either a request or response. The number of composed applications that can be deployed is increased through avoidance of serialization and de-serialization of the request headers.
- A new API has been included that provides callback when a message is not matched to an existing dialog. The API receives incoming SIP request or response messages that cannot be processed by the SIP container.

WebSphere Application Server provides numerous features to help administrators work productively so they have more time to focus on critical tasks and problem determination. These features include the following items:

- ▶ **Centralized Installation Manager (CIM)**
CIM can be used to manage version 8.5 and previous versions of WebSphere Application Server. You can use CIM to install or uninstall WebSphere Application Server on remote machines and perform maintenance from the administrative console.
- ▶ **Cross Component Trace (XCT)**
XCT enables administrators to follow the flow of requests from end-to-end. Requests can be tracked as they traverse thread or process boundaries and travel between stack products and WebSphere Application Server.
- ▶ **Configuration repository checkpoint and audit report**
A checkpoint can be configured to back up copies of files from the master configuration repository. You can use a checkpoint to restore the configuration repository back to a prior state.
- ▶ **High performance extensible logging (HPEL)**
HPEL provides a convenient mechanism for storing and accessing logging information produced by the application server or your applications.
- ▶ **IBM Support Assistant**
IBM Support Assistant is a tool provided by IBM at no charge to troubleshoot a WebSphere Application Server environment. It is composed of the following components:
 - IBM Support Assistant Workbench
 - IBM Support Assistant Agent Manager
 - IBM Support Assistant Agent

1.4 Java Management Extensions

Important: Extensive knowledge of JMX is not required to administer WebSphere Application Server. However, familiarity with some basic concepts, such as MBeans, can be useful when you are writing scripts for wsadmin.

JMX is a framework that provides a standard way of exposing Java resources. The system management functionality of WebSphere Application Server is based on the use of JMX. All operations on managed resources go through JMX functions.

The following WebSphere Application Server administration tools use JMX:

- ▶ WebSphere administrative console
- ▶ wsadmin scripting client
- ▶ Administration client Java API

1.4.1 JMX architecture

The JMX architecture is structured in three layers:

Instrumentation layer

Dictates how resources can be wrapped within special Java beans called managed beans (MBeans).

| | |
|-------------------------|---|
| Agent layer | Consists of the MBean server and agents, which provide a management infrastructure. The services that are implemented include monitoring, event notification, and timers. |
| Management layer | Defines how external management applications can interact with the underlying layers in terms of protocols, APIs, and so on. |

The use of JMX opens the door for third parties to provide management tools to administer WebSphere Application Server, for example:

- ▶ Programs written to control the WebSphere Application Server runtime and its resources by programmatically accessing the JMX API
- ▶ Applications that include custom JMX MBeans as part of their deployed code, allowing their components and resources to be managed through the JMX API

For more information about JMX, refer to the following websites:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=cxml_javamanagementx

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=txml_programming

1.4.2 JMX MBeans

Resources are managed by JMX MBeans. Each MBean wraps a certain runtime resource, for example, to expose an application server as a manageable resource, WebSphere Application Server provides an application server MBean.

External applications can interact with the MBeans through JMX connectors and protocol adapters. Connectors are used to connect an agent with a remote JMX-enabled management application. This form of communication involves a connector in the JMX agent and a connector client in the management application.

Each JMX-enabled Java virtual machine (JVM) contains an MBean server that registers all of the MBeans in the system. It is the MBean server that provides access to all of its registered MBeans. There is only one MBean server per JVM.

WebSphere Application Server provides a number of MBeans, each of which can have different functions and operations available, for example:

- ▶ An application server MBean can expose operations, such as start and stop.
- ▶ An application MBean can expose operations, such as install and uninstall.

1.5 System management in a stand-alone server environment

There are multiple levels of administration for different WebSphere Application Server environment types. In this section, and the next one, we introduce the common system management types and methods.

A stand-alone application server provides the necessary capabilities to run J2EE-compliant applications. A stand-alone application server is a good starting point for development and test teams. It can also be used for proof-of-concept or lightweight applications that do not require intensive system resources.

To create a stand-alone application server, you must create a WebSphere Application Server profile on a single (physical) machine or logical partition (LPAR) with one application server only. The profile defines the application server, node, and cell.

You can manage the application server using the administrative console, wsadmin, and command-line utilities. All of the configuration data for the application server, including the installed applications, is stored in a configuration repository created when the profile is created.

Figure 1-1 shows the system management components of a stand-alone application server environment.

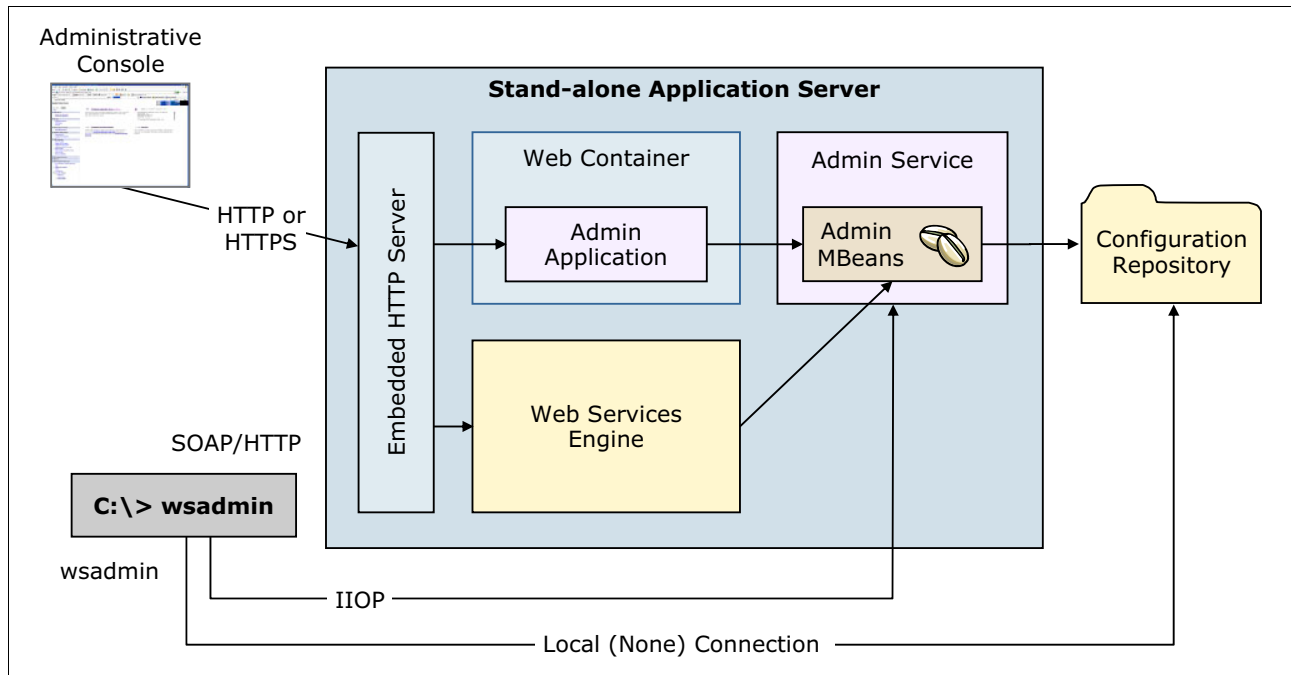


Figure 1-1 Stand-alone application server system management environment

1.6 System management in a distributed server environment

A single stand-alone server does not provide load balancing, scaling, or high-availability capability; however, a distributed server environment can help you meet these challenges by creating clusters of application servers. Clustered servers provide work load balancing, session data replication, and failover.

At a high level, building a distributed server environment involves these steps:

1. You start by creating a deployment manager profile. The deployment manager is responsible for administering the entire cell. A deployment manager administers one cell only.
2. After the deployment manager is created, the next step is to create a custom profile, which creates a second cell (defaultCell), a node, and a node agent. At this point, you do not have a functioning application server environment, just the beginnings of one. Figure 1-2 on page 13 shows this temporary stage of the environment.

- The next step is to federate the node (NodeA in Figure 1-2) to the deployment manager's cell by using the **addNode** command. After being federated, NodeA is no longer part of the defaultCell, but rather is part of the deployment manager's cell (dmgrCell).

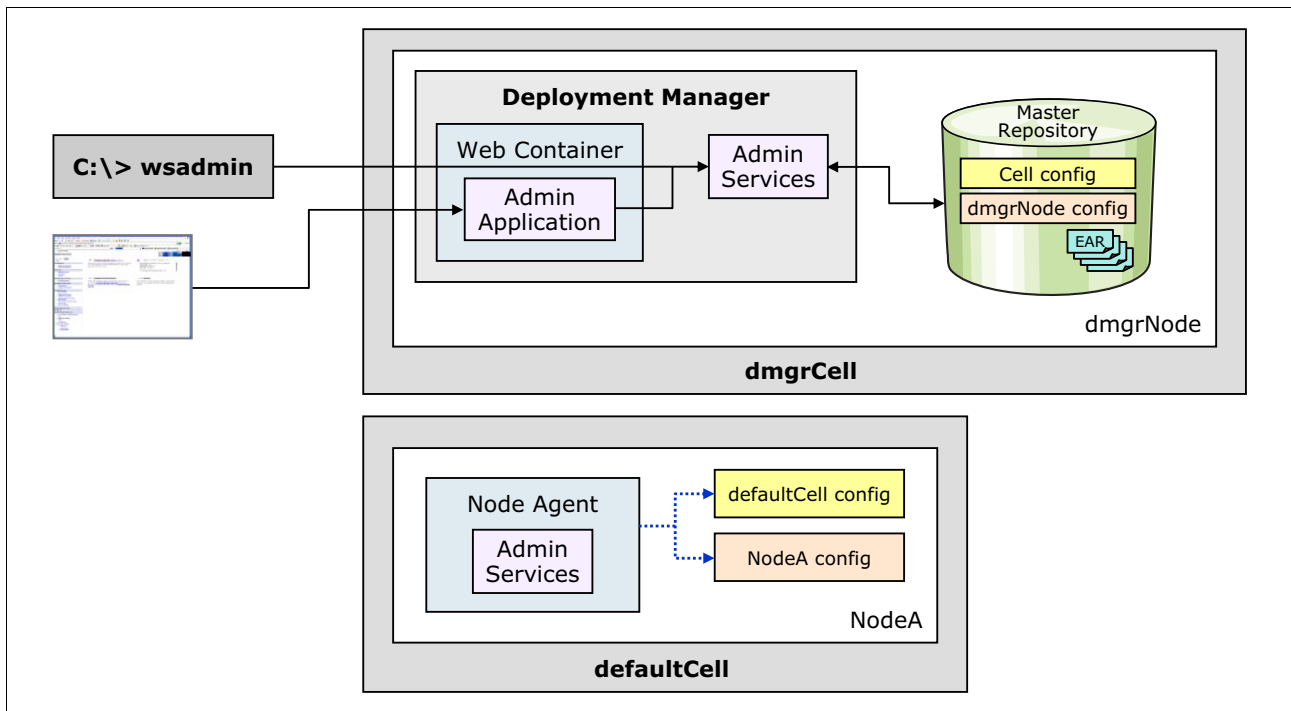


Figure 1-2 A deployment manager and unfederated custom profile

- After the federation is complete, all administration of NodeA is delegated to the deployment manager, and new application servers can be created on the node using the administrative tools for the deployment manager. This environment is illustrated in Figure 1-3 on page 14. Additional nodes can be added and servers created to create a distributed server environment.

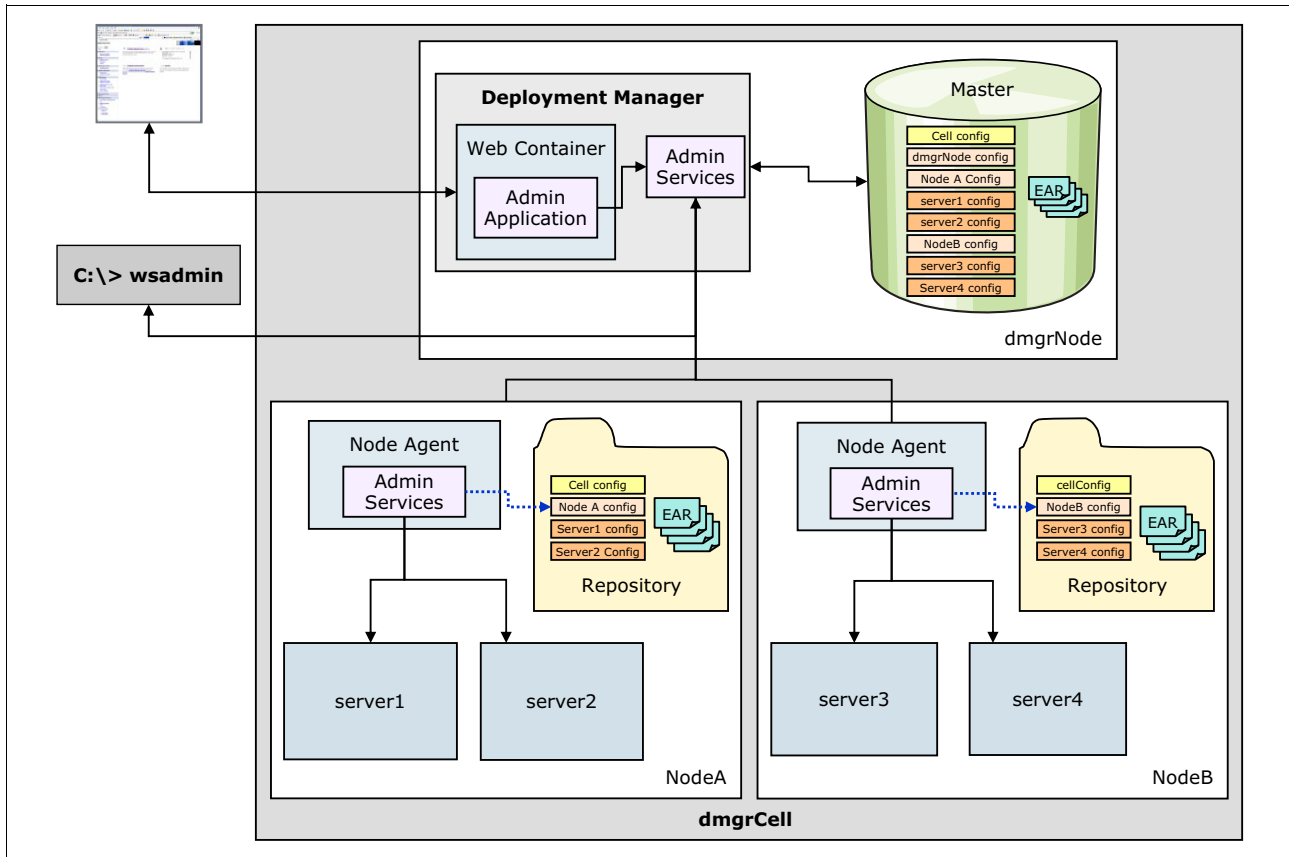


Figure 1-3 The distributed server environment

1.6.1 Centralized changes to configuration and application data

The deployment manager maintains a master repository of all the configuration files for nodes and servers in the cell. When configuration changes are made with the deployment manager, the changes are first stored in the master repository. After that, automatic or manual synchronization pushes the changes down to the affected nodes. Information about synchronization appears in 1.6.4, “File synchronization in distributed server environments” on page 20.

The configuration and application data repository is a collection of files that contain all of the information that is necessary to configure and execute servers and their applications. Configuration files are stored in XML format, while application data is stored as EAR files and deployment descriptors.

Configuration repository directory structure

Each node containing a deployment manager, application server, administrative agent, or job manager has its own profile directory under the `install_root/profiles` directory.

The repository files are arranged in a set of cascading directories within each profile directory structure, with each directory containing a number of files relating to different components of the cell, as shown in Figure 1-4 on page 15.

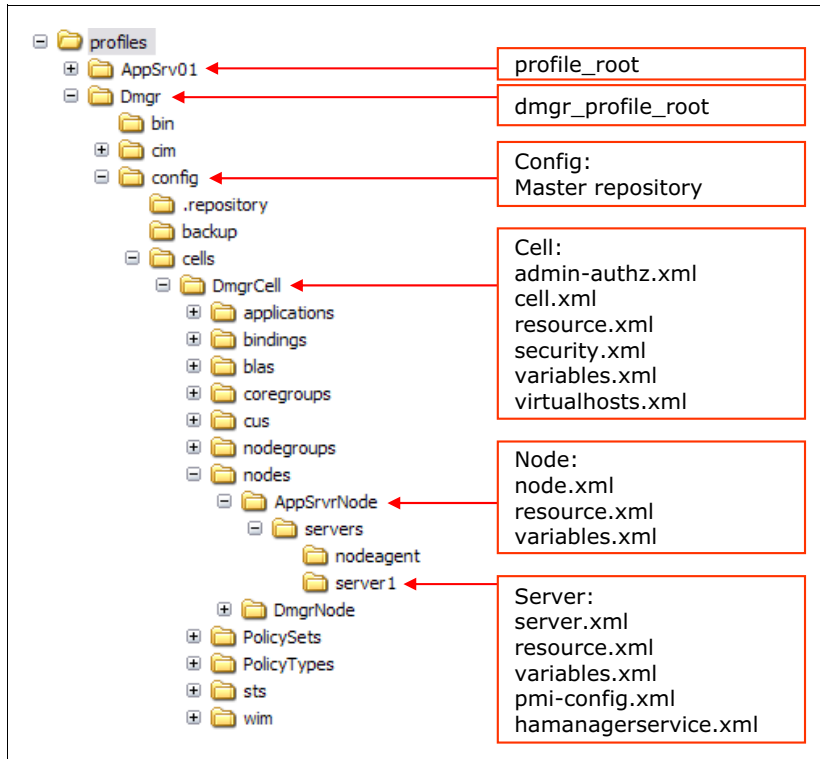


Figure 1-4 Repository directory structure

The *profile_root*/config directory is the root of the repository for each profile. It contains the following directory structure:

► *cells/cell_name/*

This is the root level of the configuration for the cell. Depending on the types of resources that are configured, you might see the following subdirectories:

- *cells/cell_name/applications/* contains one subdirectory for every application that is deployed within the cell.
- *cells/cell_name/buses/* contains one directory for each service integration bus (SIBus) that is defined.
- *cells/cell_name/coregroups/* contains one directory for each defined core group.
- *cells/cell_name/nodegroups/* contains one directory for each defined node group.
- *cells/cell_name/nodes/* contains one directory per node. Each *cells/cell_name/nodes/<node>* directory contains node-specific configuration files and a server directory that contains one directory per server and node agent on that node.
- *cells/cell_name/clusters/* contains one directory for each of the clusters managed as part of the cell. Each cluster directory contains a single file, *cluster.xml*, which defines the application servers of one or more nodes that are members of the cluster.

► *temp/*

The configuration repository uses copies of configuration files and temporary files while processing repository requests. The default location for the configuration temporary directory is *profile_root*/config/temp.

► backup/

During administrative processes, such as adding a node to a cell or updating a file, configuration files are temporarily backed up to a backup location. The default location for the backup configuration directory is *profile_root/config/backup*.

The overall structure of the master repository is the same for both a stand-alone server environment and a distributed server environment. But there are some critical differences, including:

► Stand-alone server environment:

- The master repository is held on a single machine. There is no copy of it on any other node.
- The repository contains a single cell and node.
- Because each application server is stand-alone, there is no *nodeagent/* directory.
- Clusters are not supported. Therefore, the repository tree does not contain the *clusters/* directory or subdirectories.

► Distributed server environment:

- The master repository is held on the node containing the deployment manager.
- Each node also has a local copy of the relevant configuration and application data files from the master repository.
- When changes are made to the configuration in the master repository, those changes must be synchronized to the configuration files on the nodes. Permanent changes to the configuration requires changes to the file or files in the master repository.
- Changes can be made to the configuration files on a node, but the changes are temporary and are overwritten by the next file synchronization from the deployment manager. Configuration changes made to node repositories are not propagated up to the cell.

Application data files

The *profile_root/config* directory of the master repository contains the following directory structure that holds application binaries and deployment settings:

► *cells/cell_name/applications/*

This directory contains a subdirectory for each application deployed in the cell. Names of the directories match the names of the deployed applications.

Important: The name of the deployed application does not have to match the name of the original EAR file that was used to install it. Any name can be chosen when deploying a new application, as long as the name is unique across all applications in the cell.

► *cells/cell_name/applications/app_name.ear*

Each application's directory in the master repository contains the following items:

- A copy of the original EAR, called *app_name.ear*, which does not contain any of the bindings specified during installation of the application.
- A deployments directory called *deployments/app_name/*.

- ▶ `cells/cell_name/applications/app_name.ear/deployments/app_name`

The deployment descriptors in this directory contain the bindings specified during application deployment. The deployment directory of each application contains these files:

- `deployment.xml` contains configuration data for the application deployment, including the allocation of application modules to application servers and the module startup order.
- `META-INF/`

This directory can contain these files:

- `application.xml`: J2EE standard application deployment descriptor
- `ibm-application-bnd.xmi`: IBM WebSphere-specific application bindings
- `ibm-application-ext.xmi`: IBM WebSphere-specific application extensions
- `was.policy`: Application-specific Java 2 security configuration.

The `was.policy` file is optional. If it is not present, the policy files defined at the node level apply for the application.

The subdirectories for all application modules (WARs, RARs, and EJB JARs) contain deployment descriptors and other configuration files for each module.

For further information about the individual files in each of these directories, refer to the following websites:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=rrun_rconfdoc_descriptions

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=trun_data

Configuration file location during application installation

Several things occur upon installation of an application onto WebSphere Application Server:

- ▶ The application binaries and deployment descriptors are stored within the master repository.
- ▶ The application binaries and deployment descriptors are published to each node that will host the application. These files are stored in the local copy of the repository on each node.
- ▶ Each node then installs the applications that are ready for execution by exploding the EARs under `profile_root/installedApps/cell_name/`, as follows:

- `profile_root/installedApps/cell_name/`

This directory contains a subdirectory for each application deployed to the local node.

- `profile_root/installedApps/cell_name/app_name.ear/`

Each application-specific directory contains the contents of the original EAR used to install the application, including:

- The deployment descriptors from the original EAR (which do not contain any of the bindings specified during application deployment)
- All application binaries (JARs, classes, and JSPs)

Variable scoped files

Identically named files that exist at different levels of the configuration hierarchy are called *variable scoped* files. There are two uses for variable scoped files:

- ▶ Configuration data contained in a document at one level of the configuration hierarchy is logically combined with data from documents at other levels. The most specific value takes precedence to resolve any conflicts. For example, if a variable is defined in the `variables.xml` file of both the cell and node, the entry at the node level is used.
- ▶ Documents representing data are not merged but rather are scoped to a specific level of the topology. For example, the `namestore.xml` document at the cell level contains the cell-persistent portion of the namespace, while at the node level the file contains the node-persistent root of the namespace.

1.6.2 Rules for process startup

When a managed server is starting up, it sends a discovery request message that allows other processes to discover its existence and establish communication channels with it. This action makes it possible to start the processes in a distributed server environment without following a strict order for startup, for example, a node agent can be running while the deployment manager is not active, and vice versa. When the stopped process is started, discovery occurs automatically:

- ▶ The deployment manager can be running while a managed server is not active and vice versa.

For example, if the node agent is not running when the deployment manager starts, the deployment manager tries to determine if the node agent is running but fails to set up the communication channel. When the node agent is started later, it contacts the deployment manager, creates a communication channel, and synchronizes data.

The execution of a managed server is not dependent on the presence of a running deployment manager. The deployment manager is only required when permanent configuration changes need to be written to the master repository.

- ▶ The node agent starts but no managed servers are started.

The node agent is aware of its managed servers and checks whether they are started. If so, it creates communication channels to these processes. Then, after a managed server starts, it checks whether the node agent is started and then creates a communication channel to it.

Remember: The node agent must be started before any application servers on that node are started. The node agent contains the Location Service Daemon (LSD) in which each application server registers on startup. However, the node agent is purely an administrative agent and is not involved in application serving functions.

1.6.3 Distributed process discovery

Each node agent and deployment manager maintains status and configuration information by using discovery addresses or ports. On startup, processes use these discovery addresses to discover other running components and to create communication channels between them.

Figure 1-5 on page 19 shows an example of the distributed discovery process for a topology containing two nodes that are located on different machines.

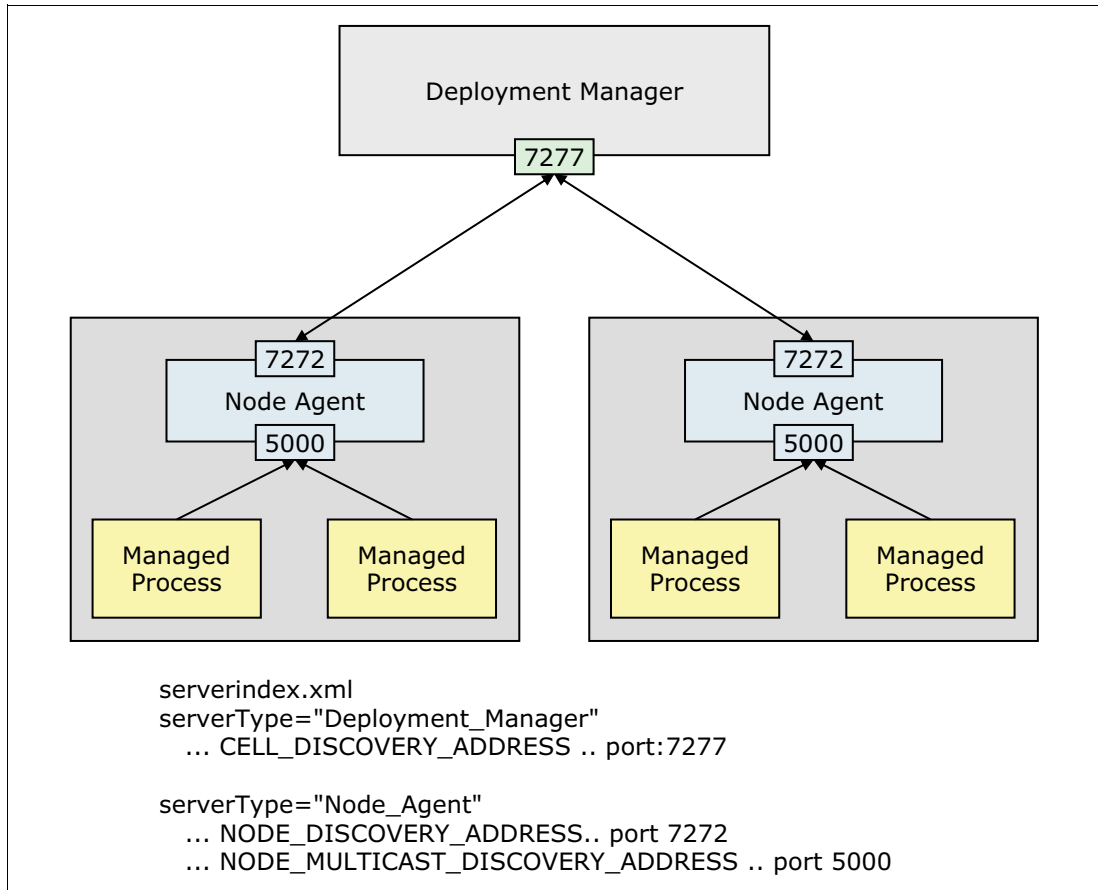


Figure 1-5 Distributed discovery process

In this example, both node agents use ports 7272 and 5000, which assumes that they reside on separate physical machines. If nodes are located on the same machine, they must be configured to use non-conflicting ports. The profile wizard automatically selects unique ports for you during profile creation.

During discovery, the following actions occur:

- ▶ The master repository located on the deployment manager installation contains the `serverindex.xml` file for each node. The deployment manager reads this file on startup to determine the host name and IP port of each node agent's `NODE_DISCOVERY_ADDRESS`.

The default port is 7272. You can display this port from the administrative console by selecting **System Administration** → **Node agents**. Then select each node agent and expand **Ports** under the Additional Properties section.

You can also verify this port by looking at the `NODE_AGENT` section in the `serverindex.xml` file of each node, which is located here:

```
dmgr_profile_root/config/cells/cell_name/nodes/node_name/serverindex.xml
```

- ▶ The copy of the configuration repository located on each node contains the `serverindex.xml` file for the deployment manager. The node agent reads this file on startup to determine the host name and IP port of the deployment manager's `CELL_DISCOVERY_ADDRESS`.

The default port is 7277. You can display this port from the administrative console by selecting **System Administration** → **Deployment manager**. Then expand **Ports** under the Additional Properties section.

You can verify this port by looking at the DEPLOYMENT_MANAGER section in the serverindex.xml file for the deployment manager node, which is located here:

profile_root/config/cells/cell_name/nodes/dmgr_node_name/serverindex.xml

- ▶ The copy of the configuration repository located on each node also contains the serverindex.xml file for the node. Each managed server reads this file on startup to determine the host name and IP port of the node agent's NODE_MULTICAST_DISCOVERY_ADDRESS.

A multicast address helps you avoid using too many IP ports for managed server-to-node agent discovery requests. Using multicast, a node agent can listen on a single IP port for any number of local servers.

The default port is 5000. You can display this port from the administrative console by selecting **System Administration** → **Node agents**. Then select the node agent and expand **Ports** in the Additional Properties section.

You can also verify this port by looking at the NODE_AGENT stanza in the serverindex.xml file of the node, which is located here:

profile_root/config/cells/cell_name/nodes/node_name/serverindex.xml

1.6.4 File synchronization in distributed server environments

The file synchronization service is the administrative service that is responsible for keeping the configuration and application data files that are distributed across the cell up to date. The service runs in the deployment manager and node agents, and ensures that changes made to the master repository are propagated out to the nodes, as necessary. The file transfer system application is used for the synchronization process. File synchronization can be forced from an administration client, or can be scheduled to happen automatically.

During the synchronization operation, the node agent checks with the deployment manager to see if any files that apply to the node were updated in the master repository. New or updated files are sent to the node, while any files that were deleted from the master repository are also deleted from the node.

Synchronization is a one-way process. The changes are sent from the deployment manager to the node agent. No changes are sent from the node agent back to the deployment manager.

Synchronization scheduling

You can schedule file synchronization using the administrative console. Click **System administration** → **Node agents** → *node_agent_name* → **File synchronization service** to choose from the available options, which are shown in Figure 1-6 on page 21.

Details of each option are:

- ▶ **Enable synchronization at server startup**
The synchronization occurs before the node agent starts a server. Note that if you start a server using the **startServer** command, this setting has no effect.
- ▶ **Automatic synchronization**
Synchronization can be made to operate automatically by configuring the file synchronization service of the node agent. The setting allows you to enable periodic synchronization to occur at a specified time interval. By default, this option is enabled with an interval of one minute.

- ▶ Startup synchronization

This setting specifies whether the node agent attempts to synchronize the node configuration with the latest configurations in the master repository prior to starting an application server. The default is to *not* synchronize files prior to starting an application server.

- ▶ Exclusions

This setting specifies files or patterns that must not be part of the synchronization of configuration data. Files in this list are not copied from the master configuration repository to the node and are not deleted from the repository at the node.

Tip: In a production environment, the automatic synchronization interval must be increased from the one-minute default setting so that processing and network impact is reduced.

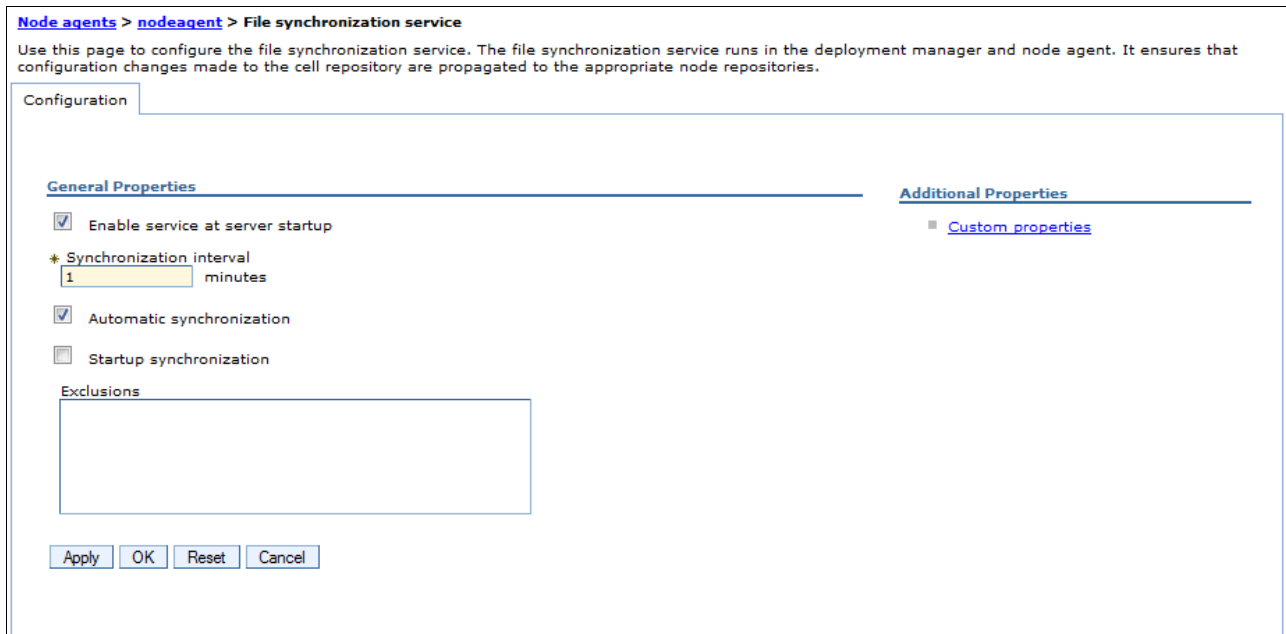


Figure 1-6 File synchronization service

How files are identified for synchronization

Deep dive: This section provides in-depth knowledge that can be useful when debugging or testing, but it is not necessary when trying to understand the overall architecture.

As part of synchronization, WebSphere Application Server must be able to identify the files that changed and therefore must be synchronized. To do this, it uses the following scheme:

- ▶ A calculated digest is kept by both the node agent and the deployment manager for each file in the configuration that they manage.

These digest values are stored in memory. If the digest for a file is recalculated and it does not match the digest stored in memory, this indicates that the file changed.

- ▶ An epoch for each folder in the repository and one for the overall repository are stored in memory.

These epochs are used to determine whether any files in the directory changed. When a configuration file is altered through one of the WebSphere Application Server administration interfaces, the overall repository epoch and the epoch for the folder in which that file resides are modified.

During configuration synchronization operations, if the repository epoch changed since the previous synchronize operation, individual folder epochs are compared. If the epochs for corresponding node and cell directories do not match, the digests for all files in the directory are recalculated, including the changed file.

Manually updating a configuration file does not cause the digest to change. Only files updated with administration tools are marked as changed. Manually updating the files is not recommended, but if you do, a forced synchronization will include any manually updated files.

Ensuring that manual changes are synchronized

Manually changing configuration files is not recommended. It must only be done as a diagnostic measure or on the rare occasion that you must modify a configuration setting that is not exposed by the administration clients. For a list of the configuration files that have settings not exposed in the administration tools, refer to the product information center at this website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=rrun_rconfdoc_descriptions

Important: Manual editing is not recommended for these reasons:

- ▶ When using wsadmin and the administrative console, you have the benefit of a validation process before the changes are applied. With manual editing, you have no such fail-safe.
- ▶ Updates made manually are not marked for synchronization and are lost at the next synchronization process unless you manually force synchronization.

Manual edits of configuration files in the master cell repository can be picked up if the repository is reset so that it re-reads all the files and recalculates all of the digests. You can reset either the master cell repository epoch or the node repository epoch, but be sure to keep these facts in mind:

- ▶ Resetting the master cell repository causes any manual changes made in the master configuration repository to be replicated to the nodes where the file is applicable.
- ▶ Resetting the node repository causes any manual changes to the local node files to be overwritten by whatever is in the master cell repository. Any manual changes in the master repository are picked up and brought down to the node.

When you manually change installed applications, they are treated the same as other configuration files in the repository in these respects:

- ▶ If you manually change the EAR file and reset the master cell repository, the changed EAR file is replicated out to the nodes where it is configured to be served and is expanded in the appropriate location on that node for the application server to find it. The application on that node is stopped and restarted automatically so that whatever is changed is picked up and made available in the application server.
- ▶ If you manually edit one of the deployment configuration files for the application and reset the repository, that change is replicated to the applicable nodes and is picked up the next time the application on that node is restarted.

Resetting the master cell repository

To perform a reset of the master cell repository, complete the following steps:

1. Make sure that the deployment manager is running.
2. Open a command prompt, change to the `dmgr_profile_root/bin` directory, and start a `wsadmin` session.

```
cd dmgr_profile_root\bin
wsadmin
```

3. Enter the following statements:

```
wsadmin>set config [$AdminControl queryNames
*:*,type=ConfigRepository,process=dmgr]
wsadmin>$AdminControl invoke $config refreshRepositoryEpoch
```

4. If the commands can be executed successfully, you can see a number returned by the `refreshRepositoryEpoch` operation.

Note: The use of `wsadmin` is covered in Chapter 8, “Administration with scripting” on page 319.

Figure 1-7 shows an example of resetting the master cell repository.

```
[root@saw211-RHEL2 profiles]# cd Dmgr/bin/
[root@saw211-RHEL2 bin]# ./wsadmin.sh
WASX7209I: Connected to process "dmgr" on node was85Dmgr01 using SOAP connector; The
type of process is: DeploymentManager
WASX7029I: For help, enter: "$Help help"
wsadmin>set config [$AdminControl queryNames *:*,type=ConfigRepository,process=dmgr]
WebSphere:name=repository,process=dmgr,platform=common,node=was85Dmgr01,version=5.0,t
ype=ConfigRepository,mbeanIdentifier=repository,cell=was85DmgrCell01,spec=1.0
wsadmin>$AdminControl invoke $config refreshRepositoryEpoch
1339541712734
wsadmin>
```

Figure 1-7 Reset the master cell repository

Resetting the master node repository

To perform a reset of the master node repository, complete the following steps:

1. Make sure that the deployment manager is running.
2. Open a command prompt, change to the `profile_root/bin` directory, and start a `wsadmin` session, as shown in the next example.

```
cd profile_root\bin
wsadmin
```

3. Enter the following statements:

```
wsadmin>set config [$AdminControl queryNames
*:*,type=ConfigRepository,process=nodeagent]
wsadmin>$AdminControl invoke $config refreshRepositoryEpoch
```

4. If the commands can be executed successfully, you can see a number returned by the `refreshRepositoryEpoch` operation.

Figure 1-8 on page 24 shows an example of resetting the master node repository.

```

[root@saw211-RHEL2 profiles]# cd Node01/bin/
[root@saw211-RHEL2 bin]# ./wsadmin.sh -port 8878
WASX7209I: Connected to process "nodeagent" on node was85Node01 using SOAP connector;
The type of process is: NodeAgent
WASX7029I: For help, enter: "$Help help"
wsadmin>set config [$AdminControl queryNames *:*,type=ConfigRepository,process=nodeagent]
WebSphere:name=repository,process=nodeagent,platform=common,node=was85Node01,version=5.0,type=ConfigRepository,mbeanIdentifier=repository,cell=was85DmgrCell01,spec=1.0
wsadmin>$AdminControl invoke $config refreshRepositoryEpoch
1339541564049
wsadmin>

```

Figure 1-8 Reset the master node repository

You can also use the explicit node synchronization process to complete the node repository reset and synchronization.

Explicit or forced synchronization

Synchronization can be explicitly forced at any time using the administrative console, the `syncNode` command, or the wsadmin scripting tool. Here are details of each option:

- ▶ Administrative console

Click **System administration** → **Nodes**, select the check box beside the node whose configuration files you want to synchronize, and click **Synchronize** or **Full Resynchronize**. Figure 1-9 shows an example of node synchronization on the administrative console.

Nodes

Use this page to manage nodes in the application server environment. A node corresponds to a physical computer system with a distinct IP following table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell clicking Add Node.

Preferences

Add Node Remove Node Force Delete **Synchronize** Full Resynchronize Stop

| Select | Name | Host Name | Version | Discovery Protocol |
|---|-----------------------------|--------------|------------|--------------------|
| You can administer the following resources: | | | | |
| | was85Dmgr01 | saw211-RHEL2 | ND 8.5.0.0 | TCP |
| <input type="checkbox"/> | was85Node01 | saw211-RHEL2 | ND 8.5.0.0 | TCP |
| Total 2 | | | | |

Figure 1-9 Node synchronization on administrative console

The Synchronize button initiates a normal synchronizing operation with no re-reading of the files. The Full Resynchronize button is the reset and recalculate function.

- ▶ `syncNode` command

This command has no cache of epoch values that can be used for an optimized synchronization and therefore performs a complete synchronization. Note that this action requires the node agent to be stopped.

The `syncNode` command resides in the bin directory under the base install or the node profile directory. To begin synchronization using this option, give the following commands:

```

cd profile_root\bin
syncNode cell_host

```


Figure 1-10 shows an example of `syncNode` command.

```
[root@saw211-RHEL2 profiles]# cd Node01/bin/
[root@saw211-RHEL2 bin]# ./stopNode.sh
ADMU0116I: Tool information is being logged in file
           /home/opt/IBM/WebSphere/AppServer/profiles/Node01/logs/nodeagent/stopSe
r.log
ADMU0128I: Starting tool with the Node01 profile
ADMU3100I: Reading configuration for server: nodeagent
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server nodeagent stop completed.

[root@saw211-RHEL2 bin]# ./syncNode.sh saw211-RHEL2
ADMU0116I: Tool information is being logged in file
           /home/opt/IBM/WebSphere/AppServer/profiles/Node01/logs/syncNode.log
ADMU0128I: Starting tool with the Node01 profile
ADMU0401I: Begin syncNode operation for node was85Node01 with Deployment
           Manager saw211-RHEL2: 8879
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0402I: The configuration for node was85Node01 has been synchronized with
           Deployment Manager saw211-RHEL2: 8879
[root@saw211-RHEL2 bin]#
```

Figure 1-10 `syncNode` command example

- ▶ wsadmin scripting tool

For information about using the wsadmin scripting tool for synchronization, refer to the product information center at this website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=xml_sync

You can use file synchronization to propagate unique configuration data that needs to be used on all nodes. To synchronize to all nodes, put the file in the `config/cells/cell_name` folder. If the file applies to just one node, put it only in the folder corresponding to that specific node. The same approach can be applied for any additional documents in a server-level folder.

1.7 Advanced system management of multiple stand-alone servers

Based on business requirements, an organization can have multiple stand-alone application servers installed on the same system or on multiple systems. These servers might be used for development, testing, staging, and so on.

A multiple stand-alone server environment can offer advantages when compared to a stand-alone server:

- ▶ Isolation for critical applications

Critical applications can be deployed on their own server to prevent negative impacts that can be caused by other, faulty applications on the same server.

- ▶ Dedicated resources

To help customize tuning, each profile has a unique JVM and unique applications, configuration settings, data, and log files.

- ▶ Enhanced serviceability

Profiles share a single set of product core files. When the product is updated, all of the profiles are updated, too.

There are two options for administering the application servers in a multiple stand-alone server environment:

- ▶ Independent administration
- ▶ Administrative agent

Table 1-1 compares the two methods of administration.

Table 1-1 Comparison of administration options for multiple stand-alone servers

| | Independent administration | Administrative agent |
|--|---|--|
| Centralized control point | No. An administrator has to juggle multiple consoles. | Yes. An administrator can use an administrative agent as the central control point. |
| System resources used for administrative functions | Each application server runs its own administrative service and the administrative console application. | After a node containing a stand-alone server is registered with the administrative agent, the administrative console application and administrative service are stopped on that application server. The administrative agent is responsible for managing all of the servers on the registered node. System resources are dedicated to running applications. |
| Management capabilities when server is not running | The administrative application and administrative service are not available if the server is not running. An administrator must start the server locally. | The administrative agent modifies the stand-alone server's configuration repository directly using the administrative service. The administrative agent can also start, stop, and create new servers within the managed node. |

Note: Combining the administrative agent with multiple stand-alone servers is a great starting point for simplifying administration. However, features, such as failover, workload management, session data replication, and many other features, cannot be configured in anything except a distributed server environment.

Figure 1-11 on page 27 shows an environment with independently managed application servers.

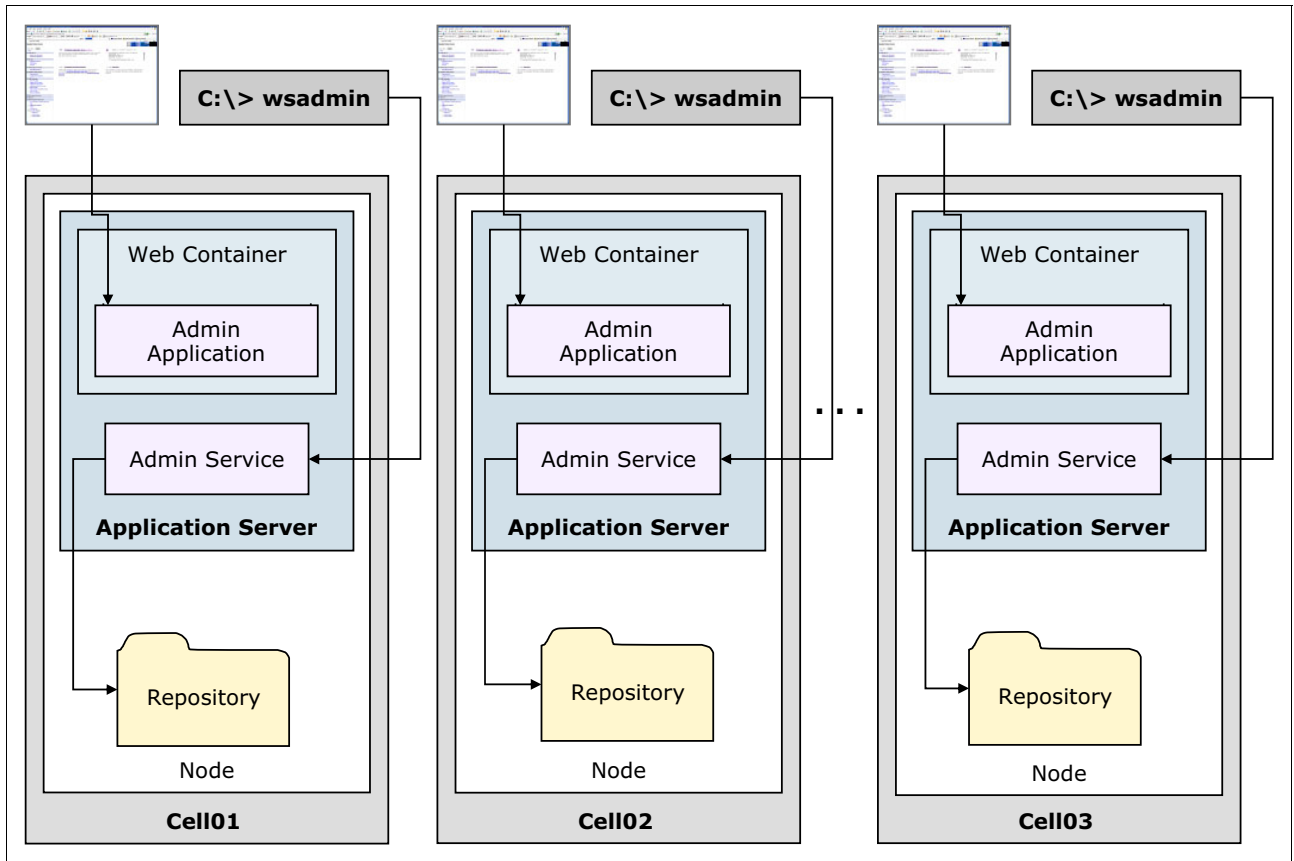


Figure 1-11 Multiple stand-alone servers with independent administration

Figure 1-12 on page 28 shows an environment using the administrative agent as the centralized control point for multiple application servers.

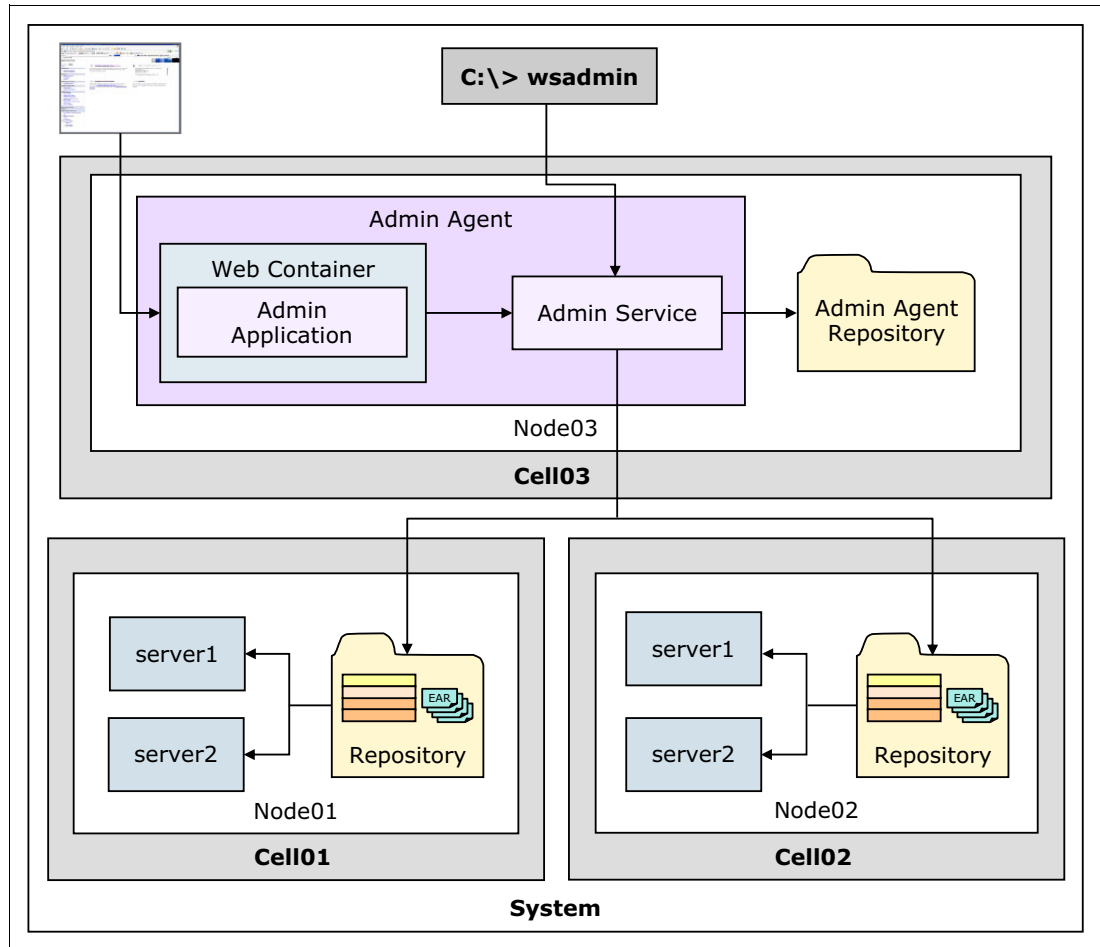


Figure 1-12 Multiple stand-alone servers managed with the administrative agent

1.8 Advanced management of distributed and stand-alone servers

The job manager can be used to administer multiple distributed environments and stand-alone servers. The job manager administers the environment asynchronously using the concept of jobs. Because jobs are submitted asynchronously, a low-latency network is sufficient, which can be useful when the environment is distributed over distant geographical areas.

The job manager is available only with the WebSphere Application Server Network Deployment offering and with WebSphere Application Server for z/OS.

The job manager administers the registered environments by submitting jobs that perform tasks, for example:

- ▶ Start and stop servers
- ▶ Create and delete servers
- ▶ Install and uninstall applications
- ▶ Start and stop applications

- ▶ Run wsadmin scripts
- ▶ Distribute files

To administer a distributed environment, the deployment manager is registered with the job manager. To administer stand-alone servers, the nodes managed by the administrative agent are registered with the job manager.

Figure 1-13 shows the relationship between the job manager and the environments with which it can interact.

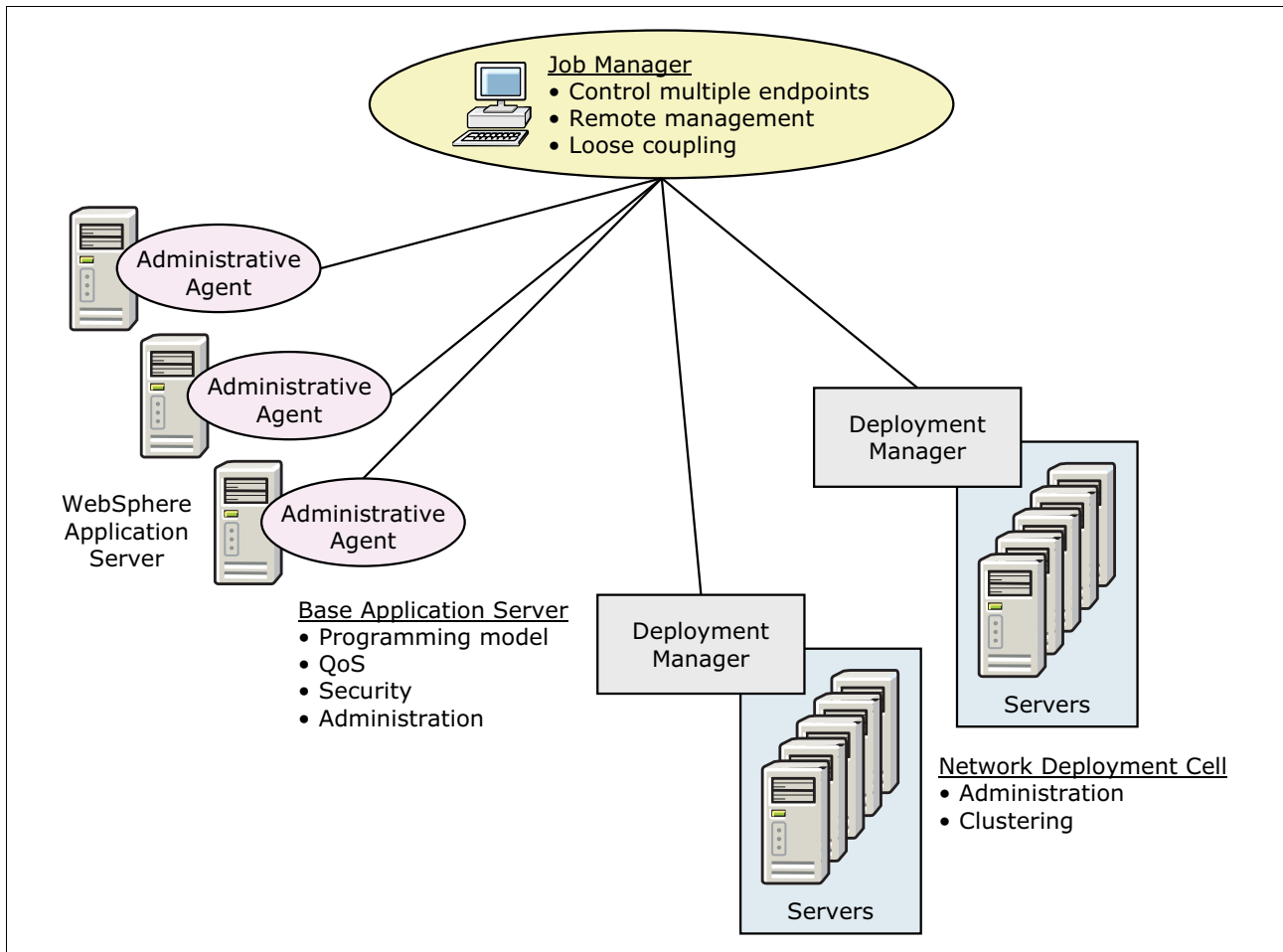


Figure 1-13 Flexible management

The job manager has a repository for its own configuration files, which are related to security, administration of the job manager, configurations, and so on. However, unlike a deployment manager, the job manager does not maintain a master repository. Instead, it allows the administrative agents and deployment managers to continue managing their environments as they if they were not registered with the job manager. The job manager can administer multiple administrative agents and deployment managers, and each administrative agent and deployment manager can be registered with multiple job managers.

Figure 1-14 on page 30 shows how the job manager provides a control point for administration in a flexible environment.

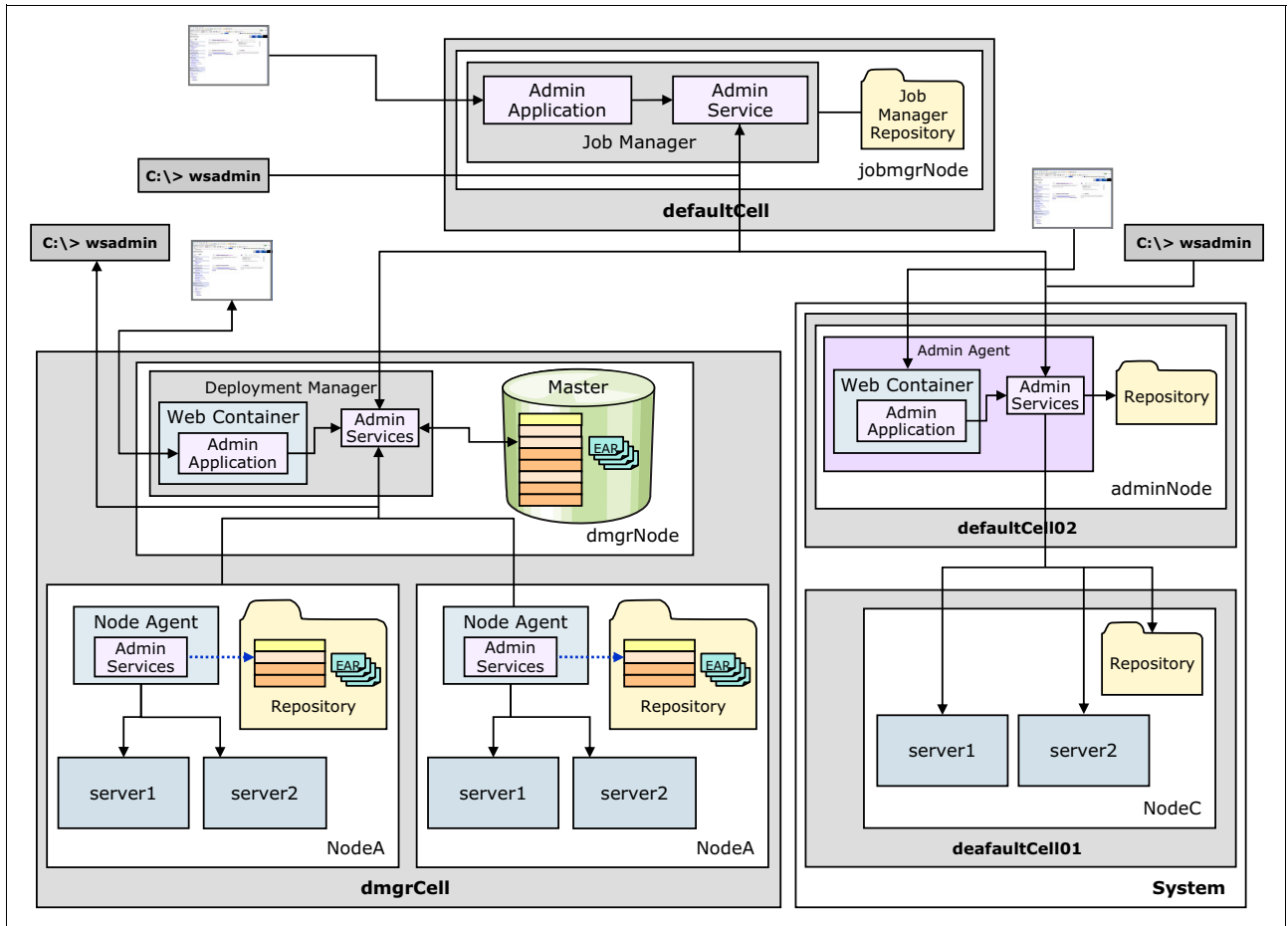


Figure 1-14 A Job manager administration environment



Installing WebSphere Application Server on distributed systems

This chapter provides an overview of IBM Installation Manager and describes how to install and use Installation Manager to install WebSphere Application Server V8.5 and its components.

This chapter contains the following sections:

- ▶ IBM Installation Manager overview
- ▶ Installation Manager installation
- ▶ Using Installation Manager
- ▶ Customizing Installation Manager
- ▶ Installing WebSphere Application Server
- ▶ Installing additional software

2.1 IBM Installation Manager overview

Installation Manager is a tool that installs and maintains Installation Manager-based software packages. It is an Eclipse-based tool that enables you to install and modify packages, search for updates, uninstall, and roll back. Installation Manager makes it easier for you to download and install code for a number of IBM software packages.

Starting from WebSphere Application Server V8, Installation Manager replaced the InstallShield MultiPlatform (ISMP) and Update Installer tools, which were used to install, update, and uninstall previous versions of WebSphere Application Server. It also replaced the functionality previously provided by the Installation Factory tool. The new WebSphere Application Server V8.5 is shipped with Installation Manager V1.5.2, but use newer versions of Installation Manager if they are available. For the current version of Installation Manager, refer to the following website:

http://www-947.ibm.com/support/entry/portal/Recommended_fix/Software/Rational/IBM_Installation_Manager

Installation Manager was originally introduced to support installation of IBM Rational® products and is currently available for all platforms and supports installation of WebSphere, Rational, and other products. It provides the following benefits:

- ▶ Consistency across all platforms using the same methodology
- ▶ Lifecycle management of any Installation Manager-installed products
- ▶ Several methods for performing lifecycle management activities
- ▶ Common packaging
- ▶ Validation and system checking performed before downloading binaries
- ▶ More efficiency when delivering new fixes and files for rollback

In the next sections, key features of Installation Manager are explained.

2.1.1 Terminology

The following installation-related concepts and terminology are used in this chapter:

- ▶ A *package* is a software product that can be installed by Installation Manager. It is a separately installable unit that can operate independently from other packages of that software. A package can include a product, a group of components, or a single component. Each package has a name, version, and an identifier, for example:

- Package name: `com.ibm.websphere.ND.v85`
- Package version: `8.5.0.20120501_1108`
- Package identifier: `com.ibm.websphere.ND.v85_8.5.0.20120501_1108`

Packages are installed to a defined directory location in the file system. Installation Manager allows you to control where products are installed.

- ▶ *Package groups* are packages that are installed to the same location that share UI elements. They are used when more than one product is installed at the same location. Some packages can be installed to the same package group and other packages must be installed to a new package group. Package group names are set automatically by Installation Manager.
- ▶ A *repository* is a place where the packages to be installed can be found. It has a list of files organized in a tree structure and includes metadata that describes the software version and how to install it. A repository can reside on a local directory or on a remote, reachable server.

- ▶ *Shared resources* are software files and plug-ins that are shared by packages and stored in a central location or shared resources directory. You can only specify the shared resources directory the first time you install a package, and you cannot change the location of the directory while packages are being installed.

2.1.2 Capabilities

Installation Manager does more than just install products. It helps update, maintain, and retire (uninstall) them, also.

Installing

Installing software is the primary task of Installation Manager. Its features allow you to install any product that is designed for use with Installation Manager, including WebSphere Application Server Version 8 or later.

Updating

Using the update feature, you can locate and install product updates and new features for packages that were installed using Installation Manager. You can find and install updates using automated searches that Installation Manager does for you, or you can download and apply updates manually. Types of updates are:

- ▶ **Fix packs:** Formal updates to software. Installation Manager indicates when a new version of the software is available.
- ▶ **Interim fixes:** Updates that apply to a specific version of software. Interim fixes are typically issued to resolve critical issues.

You can check for any fix packs or interim fixes for WebSphere Application Server at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg27004980>

You can download fix packs and interim fixes from IBM Fix Central at the following website:

<http://www-933.ibm.com/support/fixcentral/>

Modifying

Using the modify feature, you can make changes to software packages that were installed using Installation Manager. These changes include adding or removing features for an installed package, such as when you want to add an additional language pack to your current installation of WebSphere Application Server.

Rolling back

Using the rollback feature, you can remove an update and revert to a previous version of the software. For example, if you applied a fix to your existing environment but now want to remove it, the rollback feature of Installation Manager allows you to return to the previously installed package as it existed before the fix was applied.

Installation Manager saves earlier versions of packages in its local file store and uses these files to roll back. If you remove these files, Installation Manager must have access to your installation repository or disk media to roll back.

Uninstalling

Using the uninstall feature, you can remove packages that were previously installed by Installation Manager. For example, you can uninstall unnecessary language packs from WebSphere Application Server.

2.2 Installation Manager installation

Installation Manager comes in the form of an installation kit, which is an archive file containing a set of Installation Manager binaries and a flat-file repository for the Installation Manager product. The installation kit is only used for set up and maintenance of Installation Manager.

You must run Installation Manager only on systems on which you install or update product code. Typically, you need only one Installation Manager instance on a computer because one instance can track any number of product installations.

To begin an installation, obtain the Installation Manager product packages in one of the following ways:

- ▶ Copy the files from the physical disk media.
- ▶ Download the files from the IBM Passport Advantage® website:
<http://nasoftware.ibm.com/imts/us.nsf/doc/RHIS-7GUMXE>
- ▶ Download the files from an IBM repository site:
http://www-947.ibm.com/support/entry/portal/Downloads/Software/Software_support_%28general%29

The physical disk contains the rollup of all Installation Manager versions for each supported operating system, so when you insert the disk, the installation process starts automatically because it recognizes your local operating system. If you download the files, you must start the installation process yourself using the appropriate installation file.

Before installing Installation Manager, you must decide the mode in which it will run and where the binaries and runtime data will reside.

You can install Installation Manager in administrator, non-administrator, or group mode. On UNIX systems, you can install it in group mode using a predefined user group. All users in the group can then install and run the same instance of Installation Manager to manage packages.

Only one administrator instance of Installation Manager can be installed. If using non-administrator mode, there can be one instance of Installation Manager for each user.

The following sections guide you through the different methods of installing Installation Manager.

2.2.1 Using the GUI installer to install Installation Manager

The following commands are used to install Installation Manager using the graphical interface:

| | |
|------------------|------------------------------------|
| install | Installs in administrator mode |
| userinst | Installs in non-administrator mode |
| groupinst | Installs in group mode |

Group mode: Group mode is not available on Windows or IBM i systems.

To begin installation using the GUI, run the appropriate install command from the list just provided. Then perform the following steps as an administrator (these steps are written for a machine running the Windows operating system):

1. Run the **install.exe** command from your version of the unpacked installation kit.
2. In the pop-up window, the Installation Manager package is selected by default, and the status indicates the package will be installed. Click **Next**.
3. In the License Agreement window, read the license agreement, and then select the **I accept the terms of the license agreement** option to proceed with the installation. Click **Next**.
4. In the next window, provide the Installation Manager directory. You can use the **Browse** option to select a directory, or leave the default value. Click **Next**.
5. In the Summary window, review the packages and the installation paths to be installed, and then click **Install** to begin the installation.
6. During the installation process, you can observe the progress of the installation. At any time, you can select to pause or cancel the installation. At the end, a message appears indicating that the installation is complete.
7. When installation finishes, click **Restart Installation Manager** to continue to work with the tool. You can also review the installation logs by clicking **View Log File**.

2.2.2 Using console mode to install Installation Manager

Console mode is a non-graphical, text-based, interactive method for installing Installation Manager.

To install Installation Manager using console mode, run one of the following commands:

| | |
|----------------------|------------------------------------|
| installc -c | Installs in administrator mode |
| userinstc -c | Installs in non-administrator mode |
| groupinstc -c | Installs in group mode |

Example 2-1 shows samples from the installation process using console mode. Note that for each operation, the installer prompts the user for a specific action, such as typing **L** to change the default installation directory.

Example 2-1 Step-by-step installation in console mode

```
C:\installs\IM_1.5.2>installc.exe -c
Preprocessing the input.
Loading repositories...
Preparing and resolving the selected packages...
=====> IBM Installation Manager> Install
Select packages to install:
    1. [X] IBM? Installation Manager 1.5.2
    0. Check for Other Versions, Fixes, and Extensions
    N. Next,          C. Cancel
-----> [N] N
[...]
=====> IBM Installation Manager> Install> Licenses> Location
Installation Manager installation location:
    C:\Program Files (x86)\IBM\Installation Manager\eclipse
```

```

Options:
    L. Change Installation Manager Installation Location
    B. Back,      N. Next,      C. Cancel
-----> [N] L
=====> IBM Installation Manager> Install> Licenses> Location>
    Enter the Installation Manager location
Enter a new value for the Installation Manager installation location. To skip, p
ress Enter:
-----> C:\IBM\InstallationManager\eclipse
[...]
=====> IBM Installation Manager> Install> Licenses> Location> Summary
Target Location:
    Package Group Name      : IBM Installation Manager
    Installation Directory   : C:\IBM\InstallationManager\eclipse
Packages to be installed:
    IBM? Installation Manager 1.5.2
Options:
    G. Generate an Installation Response File
    B. Back,      I. Install,      C. Cancel
-----> [I] I
                25%                50%                75%                100%
-----|-----|-----|-----|
.....
=====> IBM Installation Manager> Install> Licenses> Location> Summary>
    Completion
The install completed successfully.
Options:
    R. Restart Installation Manager
-----> [R] R

```

2.2.3 Using the command line to install Installation Manager

To install Installation Manager using the command line, run the `imc1` command. This can be done either as an administrator, non-administrator, or a group user. The `imc1` command can be found in the `<IM_install>\eclipse\tools` directory.

When using the `imc1` command, you must identify the following installation attributes in the command line:

| | |
|------------------------------|---|
| packageId | Indicates the package ID or feature ID that is defined in the <code>install.xml</code> file. This ID is required because it specifies the offering to be installed. |
| repositories | Indicates the source repository for the installation. |
| installationDirectory | Indicates the installation directory for Installation Manager, which must include a path that contains spaces in quotation marks. |
| accessRights | Defines the user you are using to install. If this is not defined, <code>admin</code> is used by default. |
| acceptLicense | Indicates that you accept the license agreement. |

You can obtain a full listing of supported attributes by running the `imc1 -help` command.

Example 2-2 shows how to install Installation Manager using the command line in administrator mode. In this example, the software package was downloaded to the local machine under the C:\installs\IM_1.5.2 directory.

Example 2-2 Installing IBM Installation Manager using command-line mode

```
C:\installs\IM_1.5.2\tools>imcl.exe install com.ibm.cic.agent -repositories
C:\installs\IM_1.5.2\repository.config -installationDirectory
C:\IBM\InstallationManager2\eclipse -accessRights admin -acceptLicense
```

2.2.4 Using the silent installer to install Installation Manager

To install Installation Manager silently, run one of the following commands:

| | |
|-------------------|--|
| installc | Installs in administrator mode |
| installc | Installs in non-administrator mode on Windows and UNIX systems |
| userinstc | Installs in non-administrator mode on IBM i systems |
| userinstc | Installs as the current user |
| groupinstc | Installs in group mode |

Note: Group mode silent installation is not available on Windows systems.

Example 2-3 shows how to install Installation Manager silently as an administrator. Note that in this case, the installer chooses the default directory location.

Example 2-3 Installing IBM Installation Manager in silent mode

```
C:\installs\IM_1.5.2\> installc.exe -silent -acceptLicense
```

Installed com.ibm.cic.agent_1.5.2000.20120223_0907 to the **C:\Program Files (x86)\IBM\Installation Manager\eclipse** directory.

2.2.5 Uninstalling Installation Manager

Before you uninstall Installation Manager, you must uninstall all of the packages that were previously installed using it. Be sure to close Installation Manager before starting the uninstall process. You must also log into the machine with the identity you used when installing Installation Manager.

To uninstall Installation Manager, use one of the following options:

► Windows systems:

- GUI uninstall:

Click **Control Panel** → **Add or Remove Programs**. Click **IBM Installation Manager** and click **Uninstall**.

- Silent uninstall:

Navigate to the IBM Installation Manager `uninstall` directory (by default it is under the `ProgramData\IBM\Installation Manager` directory). Run **uninstallc.exe** in admin mode or **userinstc.exe** in non-admin mode.

► UNIX systems:

- GUI uninstall:

Navigate to the `/var/ibm/Installation Manager/uninstall` directory, and run `uninstall`.

- Silent uninstall:

Navigate to the `/var/ibm/Installation Manager/uninstall` directory, and run `uninstallc`.

2.3 Using Installation Manager

After you install Installation Manager, you can use it to install packages, update or modify them, and so on. Installation Manager tracks the packages that it installs, including selectable features and maintenance updates for products.

There are a number of ways you can interact with Installation Manager:

- ▶ Wizard mode: A graphical user interface
- ▶ Command-line mode: The command-line utility (`imcl`)
- ▶ Console mode: An interactive, text-based user interface
- ▶ Silent mode: Response file-based, run from the command line or a file

In this section, we provide information about the various ways to use Installation manger.

2.3.1 Wizard mode

Installation Manager includes a number of wizards to help maintain product packages:

- ▶ Installation wizards take you through the installation process for various operating systems and modes. They provide default settings that can be customized for your environment. You can install multiple packages simultaneously.
- ▶ The Update wizard allows you to update currently-installed packages.
- ▶ The Modify wizard helps you change certain elements of installed packages, such as adding or removing features.
- ▶ The Rollback wizard allows you to revert to a previous version of a package.
- ▶ The Uninstall wizard removes installed packages.

To use wizard mode, navigate to the Installation Manager install directory, and run the appropriate command for your system:

- ▶ Windows: **IBMIM.exe**
- ▶ UNIX: **IBMIM**

Figure 2-1 on page 39 illustrates the main Installation Manager window with available wizard mode operations.

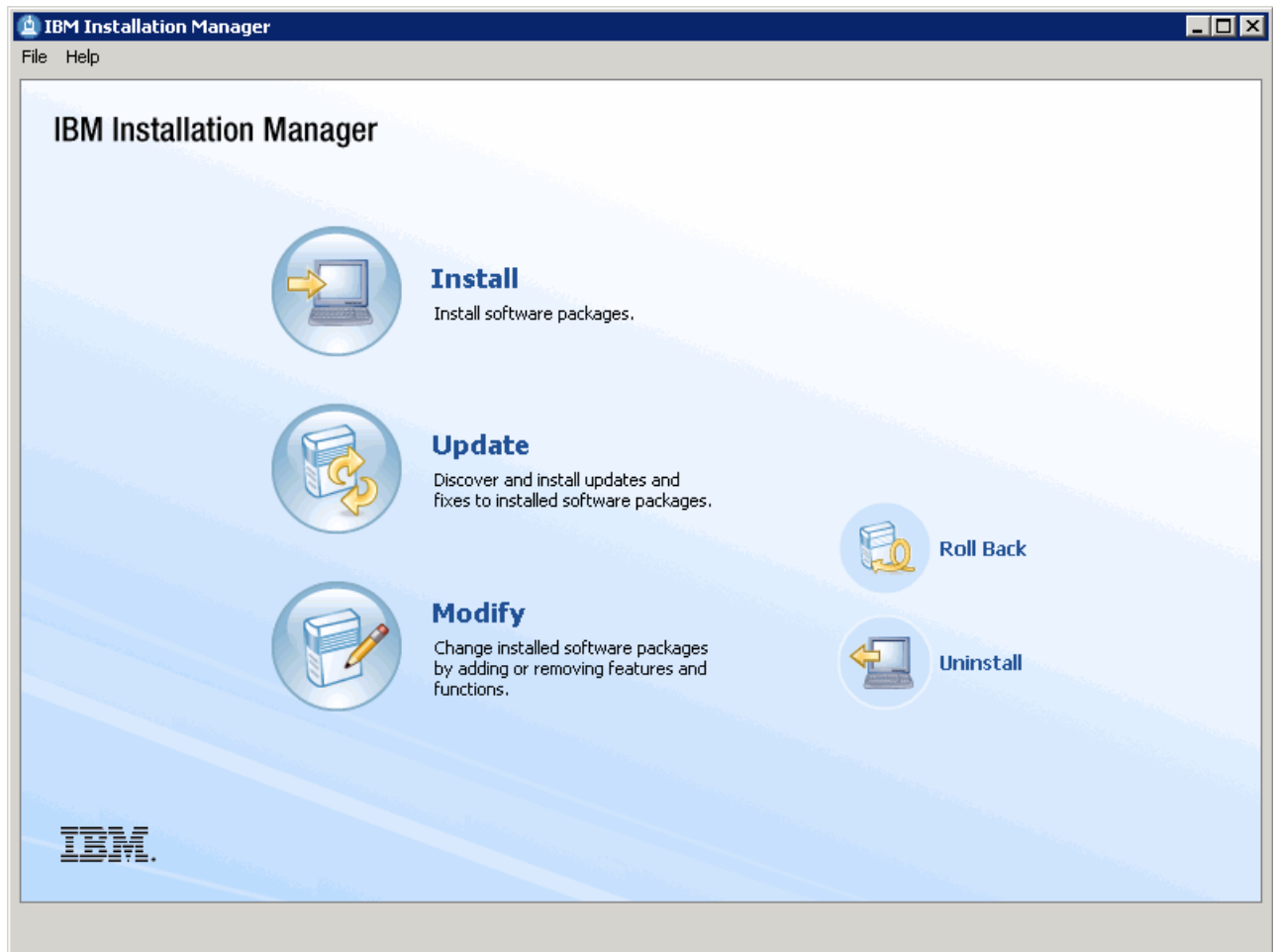


Figure 2-1 Main Installation Manager window viewed in wizard or graphical mode

2.3.2 Command-line mode

If you cannot use the graphical user interface, or have a preference for a non-GUI environment, you can operate Installation Manager in command-line mode. Using the command line, you can install, update, and uninstall packages, list installed features and packages, list available packages, display version information, or import a response file to be used for a silent installation.

Command line operations are invoked using the `imcl` command, which can be found in the `<IM_install>/tools/` directory. For a list of help topics for using this mode, type the following command:

```
imcl.exe help
```

2.3.3 Console mode

Another option for interacting with Installation Manager is through the console mode.

Important: Console mode does not support installation of WebSphere Application Server V8.5.

To start console mode, use the command appropriate for your system:

- ▶ Windows: Navigate to <IM_install>\tools\ and run **imcl.exe -c**
- ▶ UNIX: Navigate to <IM_install>/tools/ and run **imcl -d**

Example 2-4 shows the Installation Manager console-mode command and welcome text.

Example 2-4 Installation Manager welcome text in console mode

```
C:\IBM\InstallationManager\eclipse\tools>imcl.exe -c
```

```
=====> IBM Installation Manager
```

```
Select:
```

1. Install - Install software packages
2. Update - Find and install updates and fixes to installed software packag
3. Modify - Change installed software packages
4. Roll Back - Revert to an earlier version of installed software packages
5. Uninstall - Remove installed software packages

```
Other Options:
```

- L. View Logs
 - S. View Installation History
 - V. View Installed Packages
 -
 - P. Preferences
 -
 - E. Export Data for Problem Analysis
 - A. About IBM Installation Manager
 -
 - X. Exit Installation Manager
-

2.3.4 Silent mode

Silent mode allows you to install packages in a non-interactive and non-GUI mode. It uses a response file to provide the input for each installation.

The key to silent mode installations, then, is to create the response files that guide each effort. Response files can be used to install, update, modify, roll back, and uninstall software packages.

Creating response files

A response file can be recorded using Installation Manager or created manually using a documented list of commands.

To use the Installation Manager GUI to record a response file for the installation of a package, use the **skipInstall** argument from the command line. This argument makes Installation Manager simulate the package-installation process instead of actually installing anything, after which an XML response file can be recorded containing the steps from the simulation. The response file can then be used to automate the recorded installation process each time it is needed.

Based on this approach, start the GUI with the following options:

- ▶ **skipInstall** (indicates to skip the install)
- ▶ **record** (specifies the response file to be created)

Details about available arguments can be obtained using the following command:

```
IBMIM -help
```

Example 2-5 shows the command string to create a response file using the Installation Manager GUI on a Windows platform. Running these commands launches the GUI wizards where you select the appropriate repositories and packages, and then click **Install** to begin the simulation and recording. To finish generation of the recording, close the Installation Manager.

Example 2-5 Creating a response file using Installation Manager GUI mode

```
C:\IBM\InstallationManager\ eclipse>IBMIM.exe -skipInstall "c:\temp" -record "c:\temp\myResponseFile.xml"
```

To manually create a response file, you can use a sample response file and customize it to suit your environment. Sample response files are at this website:

http://publib.boulder.ibm.com/infocenter/install/v1r5/index.jsp?topic=/com.ibm.silentinstall12.doc/topics/c_sample_response_files.html

Important: You must record a response file on the same platform that you plan for the installation. For example, to install on a computer running Microsoft Windows, you must record the response file on a computer that runs Windows. If you plan installations for multiple platforms, you must have a response file for each platform.

Using response files

After you create a response file, you can use it to silently manage a package with Installation Manager.

To use a response file, start the installation from the command line, and use the input parameter to pass your response file, as shown in Example 2-6.

Example 2-6 Managing packages with a response file using silent mode

```
C:\IBM\InstallationManager\ eclipse\tools>imcl.exe -acceptLicense -input C:\temp\myResponseFile.xml -log C:\temp\silentInstall.xml
```

Important: Silent installation cannot install packages that are contained on multiple media disks. Ensure that you are using a single repository location when you use silent mode.

2.4 Customizing Installation Manager

Installation Manager operations can be customized by setting preferences, defining the way repositories are configured, and so on. There are also tools to help with troubleshooting and easy methods of keeping the Installation Manager software up to date.

2.4.1 Installation Manager preferences

You can influence how Installation Manager operates by configuring preferences for repositories, appearance, files for rollback, help, Internet settings, Passport Advantage settings, and updates. You can keep the default settings or modify the preferences to suit your environment.

Installation Manager preferences can be modified using the GUI wizard mode console mode, which is considered the easiest method.

To verify or modify preferences using the GUI, select **File** → **Preferences**. Figure 2-2 shows available preferences that can be configured.

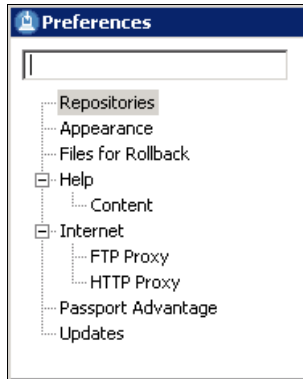


Figure 2-2 Installation Manager preferences

Here are the details for each preference setting:

► **Repositories**

This preference identifies any number of repositories to be used by Installation Manager for installing, modifying, and updating packages.

► **Appearance**

This preference allows you to select whether or not to display the internal version of the package being installed. With this setting, the internal version can be displayed during the installation process, including the release number, year, month, day, and package ID. By default, the internal version is not displayed.

► **Files for Rollback**

This preference allows use of the rollback feature to revert to a previous installed version of an updated package. It is enabled by default.

► **Help**

This preference allows you to choose how help information is displayed. For example, help contents can be launched in the help browser, which is the default setting or launched in an external browser. You can also configure access to multiple information centers for products you are installing, with options for including local or remote help and prioritizing which information center is used first.

► **Internet**

This preference allows you to set options for proxy servers.

► **Passport Advantage**

This preference is used to establish the settings for a Passport Advantage site. Internet connectivity is required for this option to be selected. By default, this option is not selected. To enable it, select **Connect to Passport Advantage**, and you are prompted for a user name and password for the service. Select **Save password** to save the user name and password credentials.

- ▶ Updates

This preference is used to indicate if Installation Manager searches for updates to its own software when it is installing, modifying, or updating packages. Internet connectivity is required for this option to be selected. By default, this option is disabled.

2.4.2 Repositories overview

Installation Manager uses repositories to identify the packages or updates to install. A repository is a location that stores data for installing, modifying, rolling back, updating, or uninstalling packages. Each installed package has an embedded location for its default update repository. You can add, edit, or remove repositories.

Installation Manager uses the configured repositories to determine and list all of the available packages to install, which can include products, fix packs, interim fixes, and so on. It checks prerequisites and dependencies and then installs the selected packages.

An Installation Manager repository contains one or multiple product offerings, each containing both metadata and the actual offering payload. The metadata describes the following aspects of offerings:

- ▶ Name, version, and supported platforms
- ▶ Required and optional features
- ▶ Relationships and dependencies between offerings and features of offerings

Typically, a repository contains all of the files needed for installation on various platforms, operating systems, and so on.

Repository topologies can be generalized as fitting within the following categories:

- ▶ Public repository: Accessible to the general public at an IBM hosted site, such as IBM Passport Advantage
- ▶ Local repository: Used by a single user and not shared with others
- ▶ Enterprise repository: Located behind the firewall and accessed by multiple machines within the enterprise

2.4.3 Repository configuration

In this section, we provide information about the configuration steps for working with repositories.

Service repository

By default, Installation Manager is configured to use a service repository that is located on an IBM repository website. In this case, Internet access is required.

If a machine does not have Internet access, Installation Manager can be configured to look for a local repository. Updates can be downloaded and placed in a temporary directory on the local machine for Installation Manager to find them. You must manually configure local repositories.

To verify or modify the service repository setting, click **File** → **Preferences** → **Repositories** in the Installation Manager GUI. The Repositories window contains the area for repository configuration, as illustrated in Figure 2-3 on page 44.

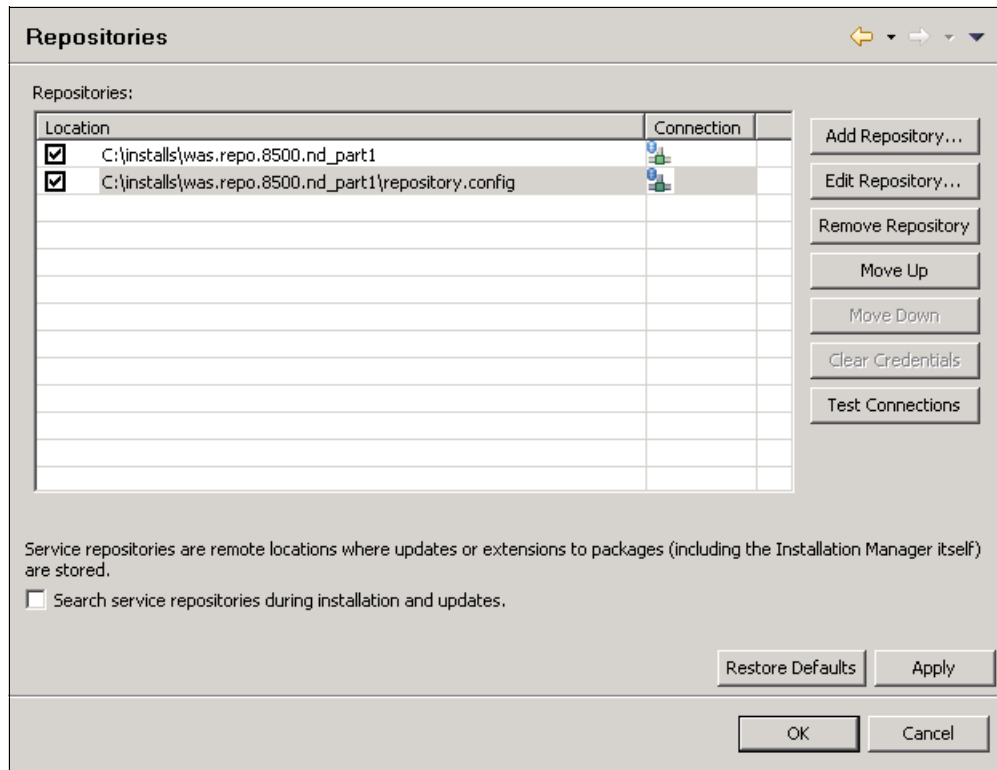


Figure 2-3 Installation Manager repositories

Tip: If a machine does not have access to an IBM repository site or you do not want the machine to access that site, clear the **Search service repositories during installation and updates** check box. When this option is enabled, the Install, Modify, and Update wizards try to access the IBM repository site. If a connection is not made, the activity times out and Installation Manager tries again to reconnect before starting the install, update, or modify process.

Adding and selecting repositories

You can add any number of repositories for Installation Manager to search when installing, modifying, or updating a product package. To add a repository, select **Add Repository**, and then choose the repository location and file type. The repository file type can be any of the following items:

- ▶ A repository.config file included in the product repository files
- ▶ A diskTag.inf file that tells the Installation Manager that the files are from a disk
- ▶ A JAR file that contains a repository (an option often used for license kits)
- ▶ A compressed file that contains a diskTag.inf file (must be extracted to the local system prior to use)

Installation Manager searches the repositories in the order that they are listed in the repository window. If two repositories use the same package, the repository that is listed higher in the order is used. You can move a repository up and down in the order list by selecting the appropriate repository and clicking either **Move Up** or **Move Down**.

Installation Manager only searches the repositories that you selected in the repository page. To select a repository, select the check box beside the repository name. If you clear the check box beside a name, Installation Manager will not search that repository.

After you add a repository, you can test the connection to it. Select the check box beside the repository name, and click **Test Connections**. If Installation Manager can access the repository, the repository is connected and the icon in the Connection column reflects this status, as shown in Figure 2-3 on page 44. If a connection cannot be made to a repository, a message and icon indicate the failed connection status.

You can also edit and remove repositories from the repository listing.

2.4.4 Updating Installation Manager

Installation Manager can be configured to automatically search for updates to itself. To verify or modify this setting, click **File** → **Preferences** → **Updates** in the Installation Manager GUI. If this option is selected, network connectivity is required so Installation Manager can look for any updates. If it finds an update, you are prompted to take action.

If the updating option is cleared, Installation Manager does not look for updates to itself. For more information about installing or updating Installation Manager, refer to the product information center at this website:

<http://publib.boulder.ibm.com/infocenter/install/v1r5/index.jsp>

2.4.5 Managing packages

When using Installation Manager, you can list the installed packages or list all packages that are available to be installed, updated, modified, and rolled back.

Example 2-7 shows how to list installed packages using the Installation Manager command-line mode. The only package installed in this case is the one for Installation Manager itself.

Example 2-7 Lists Installed packages

```
C:\IBM\InstallationManager\eclipse\tools>imcl.exe listInstalledPackages  
com.ibm.cic.agent_1.5.2000.20120223_0907
```

You can also list installed packages using the Installation Manager GUI by clicking **File** → **View Installed Packages**.

In Windows, you can also click **Start** → **All Programs** → **IBM Installation Manager** → **View Installed Packages** to open the installed packages' information in a web browser.

2.4.6 Examining log files

Installation Manager creates log files that you can use to troubleshoot any installation problems. Consider verifying the log files after any installation to ensure that everything in that process went successfully.

If you use the GUI mode, you can view log files immediately after installation by clicking **View Log File** on the summary page. This launches the Installation Log window, shown in Figure 2-4 on page 46.

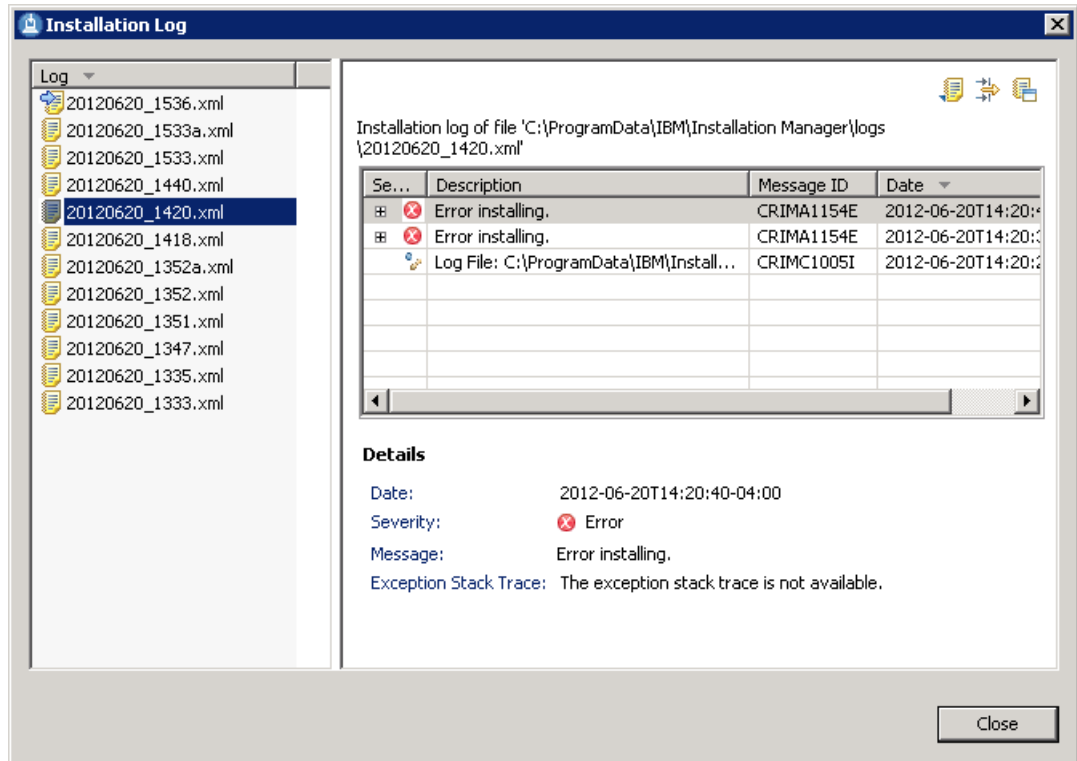


Figure 2-4 Installation Manager log viewer

You can view the Installation Log window at any time by clicking **File** → **View Log**.

The window provides an easy and convenient interface where all Installation Manager log files can be examined. In addition, you can perform the following actions:

- ▶ Export an XML log file to a location in the file system
- ▶ Filter search contents to narrow down the results, such as by the severity level of the event (Error, Warning, Information, and Note, all of which are selected by default except Information)
- ▶ Open a selected log file in a web browser

To examine the logs manually, locate the Installation Manager logs directory. The default location for this directory varies according to the operating system:

- ▶ Windows: C:\ProgramData\IBM\Installation Manager\logs
- ▶ UNIX: /var/ibm/InstallationManager/logs

2.5 Installing WebSphere Application Server

Starting with WebSphere Application Server V8, Installation Manager is required to install the product. You can use the GUI or silent installation modes. This section guides you through the process.

2.5.1 Prerequisites

Prior to installation, verify the list of hardware and software requirements and supported platforms for WebSphere Application Server V8.5 at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg27023941>


You should also familiarize yourself with the planning considerations for installing WebSphere Application Server. Refer to the planning portion of the product information center website for more information:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.base.doc/ae/tins_scenario3.html

2.5.2 Using GUI mode

To install WebSphere Application Server using GUI mode:

1. Start Installation Manager using the **IBMIM** command.
2. Add a repository for your WebSphere Application Server V8.5 package. Refer to “Adding and selecting repositories” on page 44 for details.
3. Click the **Install** option in the main Installation Manager window to trigger the installation wizard.
4. Installation Manager searches all defined and enabled repositories and lists the available packages in the Installation Packages window. Select the WebSphere Application Server package (see Figure 2-5), and click **Next**.



| Installation Packages | Status | Vendor |
|---|-------------------|--------|
| <input checked="" type="checkbox"/> IBM WebSphere Application Server Network Deployment | | |
| <input checked="" type="checkbox"/> Version 8.5.0.0 | Will be installed | IBM |

Figure 2-5 Selecting the WebSphere Application Server package for installation

Important: If WebSphere Application Server is already installed on the machine, the Status column will indicate *Installed*. However, you can still install another instance of WebSphere Application Server on the same machine. When you select the package, a message states that it is already installed. In this case, click **Continue** to install the second instance of the package. The second package must be installed to a new package group and in a different location.

5. In the License Agreement window, read the license agreement, and then select **I accept the terms of the license agreement** to proceed with the installation. Click **Next**.
6. In the Location window (Figure 2-6 on page 48), provide a directory location for shared resources. The shared resources directory is used by multiple packages and is configured only during the first product package installation. You cannot change this directory later. Decide whether to keep the default directory or modify it to suit your environment, and then click **Next**.

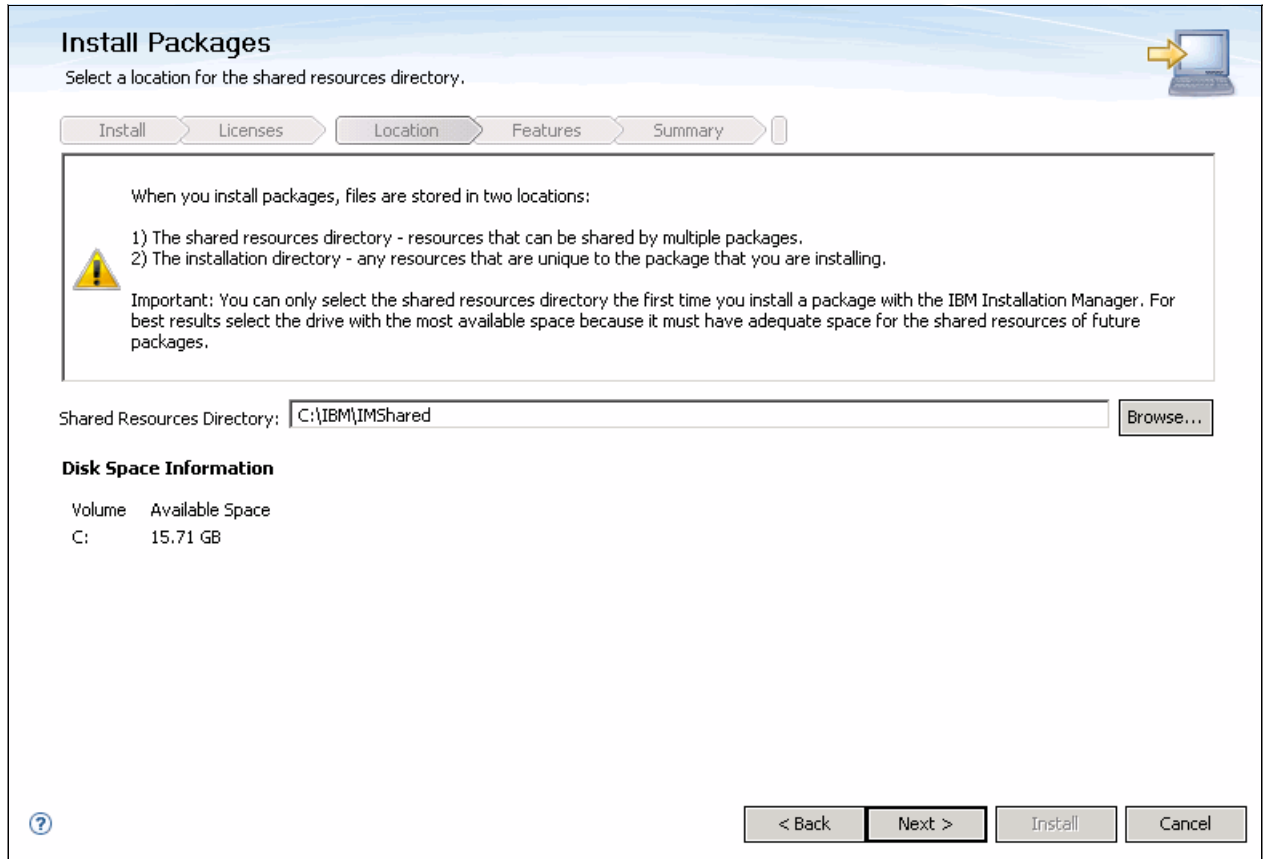


Figure 2-6 Defining the shared resources directory location in the IBM Installation Manager

- In the next window, provide the installation directory, and click **Next**. The default directory location differs depending upon the operating system. In this example, the directory location is modified to C:\IBM\WebSphere\AppServer.

Note: On a Windows Vista, Windows 7 or Windows Server 2008 operating system, WebSphere Application Server V8.5 does not function properly if a non-administrator installs it into the Program Files or Program Files (x86) directory with User Account Control (UAC) enabled.

- In the next window, you can select additional language packs to install along with the product. The English language pack is selected by default and cannot be disabled. When finished, click **Next**.
- The Features window (Figure 2-7 on page 49) opens next and lists all of the available product features. Select the features you want to install, and click **Next**. For this example, only default features are used.

Important: For WebSphere Application Server V8 and later versions, the only sample application shipped with the product is PlantsByWebSphere. You can select to install the sample application now or modify the installation later to add the sample application. You can no longer deploy samples during profile creation. All previous sample applications that were included in Version 7 and are still relevant, and several new samples, were placed online for download from the product information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.samples.doc/ae/welcome_samples.html

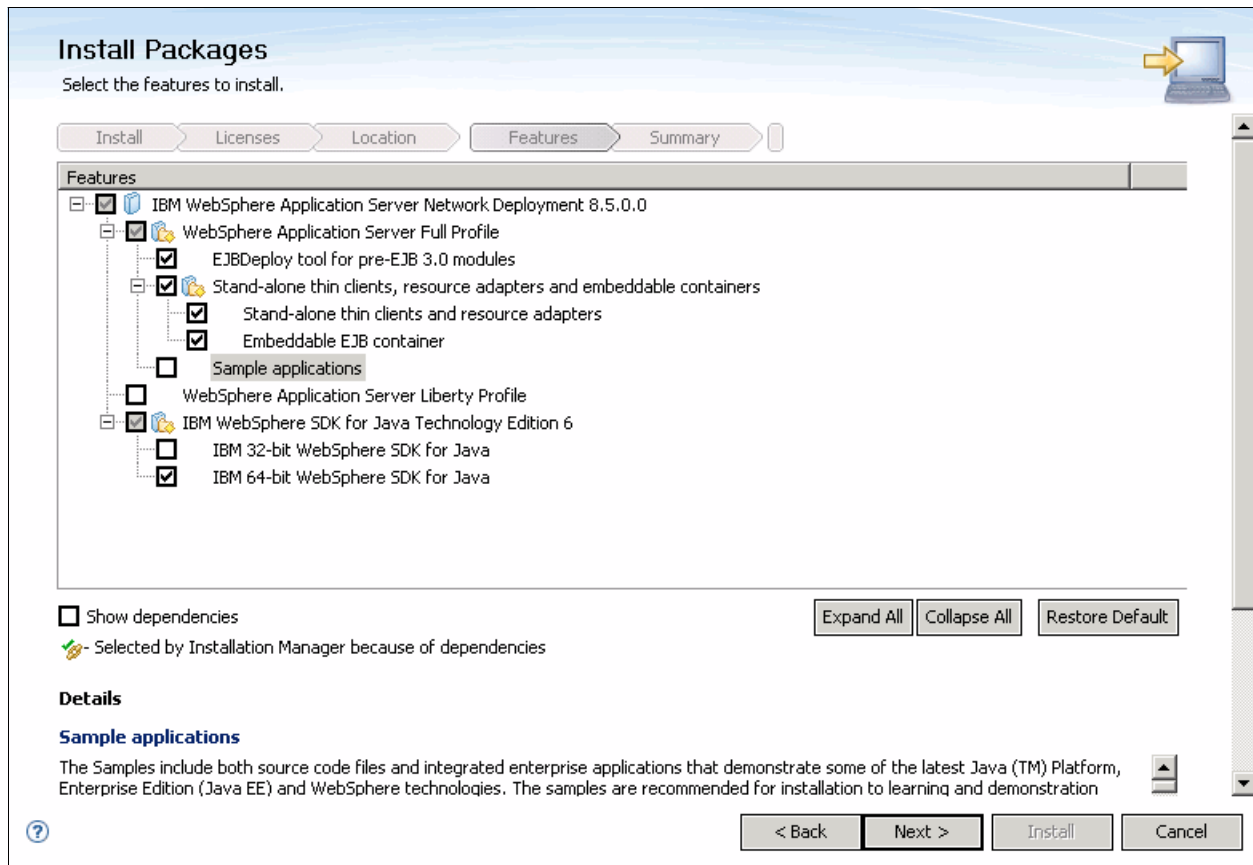


Figure 2-7 Selecting features of WebSphere Application Server V8.5 for installation

The Software Development Kit (SDK) feature only appears on the Features window when installing on a 64-bit operating system. Either the IBM 32-bit SDK for Java Version 6 or IBM 64-bit SDK for Java Version 6 feature must be selected. After this feature is installed, it cannot be modified.

If the installation is being performed on a 32-bit operating system, the choice between the two SDKs is not available and the installation automatically defaults to the IBM 32-bit SDK for Java Version 6. Starting with Version 8, there is just one installable package for both 32-bit and 64-bit operating systems.

Selecting the IBM 32-bit SDK for Java Version 6 feature in this example is equivalent to installing a 32-bit WebSphere Application Server on a 64-bit operating system.

10. Review the summary information, and click **Install** to begin the WebSphere Application Server V8.5 installation process.

11. When the installation completes, the results are displayed as shown in Figure 2-8. If problems are evident, you can review the installation log file to troubleshoot any problems by clicking the **View Log File** link.

Note that from the installation complete window, you can start another tool or exit the installation process. You have these options:

- Profile Management Tool to create a profile: This option allows you to create a new profile using the Profile Management Tool.
- Profile Management Tool to create an application server profile for a development environment: This option allows you to create a new application server profile with settings for a development environment. If the application server will be used primarily for development purposes, select this option to create it from a special, development template. This approach reduces the startup time and allows the server to run using fewer resources. Do not use this option for production servers.
- None: This option indicates that you do not want to create a profile at this time.

Select an option, and click **Finish**. Additional information about creating profiles for WebSphere Application Server V8.5 is provided in 3.3, “Building systems with profiles” on page 64.

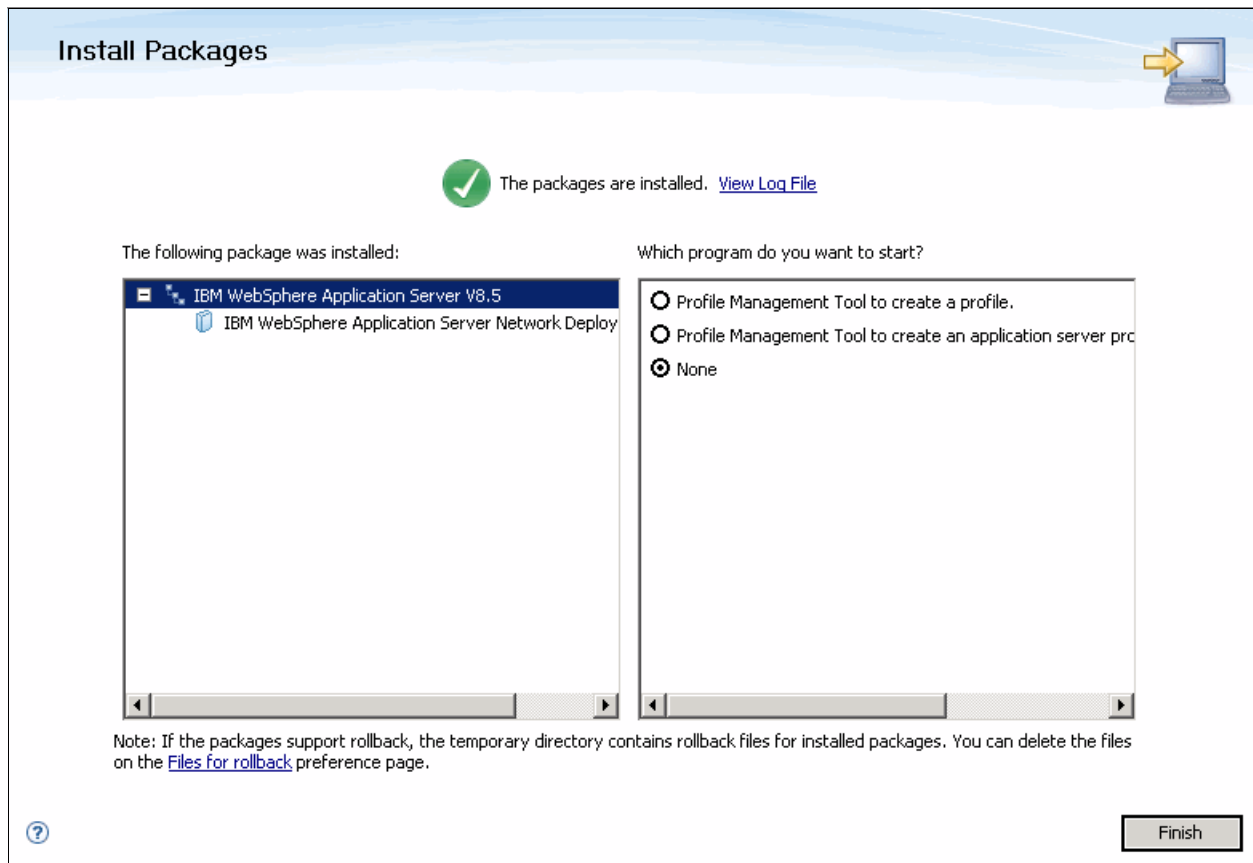


Figure 2-8 Summary window after successful installation of WebSphere Application Server V8.5

2.5.3 Using silent mode

To install WebSphere Application Server silently, you must first create a response file and then perform a series of steps in command-line mode.

Creating the response file

This procedure was introduced in “Creating response files” on page 40, where it was explained, among other things, that creating a response file involves running a simulated installation process. Follow these steps to create a response file for installation of WebSphere Application Server V8.5 using Installation Manager:

1. Run the **IBMIM** command, as shown in Example 2-8.

Example 2-8 Creating a response file using the Installation Manager GUI mode

```
C:\IBM\InstallationManager\ eclipse>IBMIM.exe -skipInstall "c:\temp" -record  
"c:\temp\was85_install_response.xml"
```

Note that this time the title of the Installation Manager window is marked with an additional *Recording* label, as illustrated in Figure 2-9.

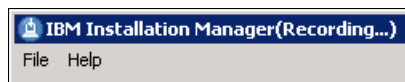


Figure 2-9 Indication that Installation Manager is running in recording mode

2. When the Installation Manager GUI launches, add the repository for the WebSphere Application Server package, if required. See “Adding and selecting repositories” on page 44 for details about adding a repository.
3. Click the **Install** wizard to begin the simulated installation.
4. Select the WebSphere Application Server package, and click **Next**.
5. Accept the license agreement, and click **Next**.
6. Enter the shared resources directory location, and click **Next**.
7. Enter the WebSphere installation directory location, and click **Next**.
8. Select the optional language packs, and click **Next**.
9. Select WebSphere Application Server features, and click **Next**.
10. Review the installation configuration, and click **Install**. The simulated installation process is fast because no binaries are being installed.
11. Click **Finish** to conclude the simulated installation process.
12. Exit Installation Manager to end the recording of the response file.

After it is created, examine the response file (for example `was85_install_response.xml`) and modify it as needed.

Installing the product

With the response file created, perform the following steps to install WebSphere Application Server silently using command-line mode:

1. Run the silent installation command used in Example 2-9, specifying the response file that you just created.

Example 2-9 Silent installation of WebSphere Application Server

```
C:\IBM\InstallationManager\ eclipse\tools\>imcl.exe -acceptLicence input  
C:\temp\was85_install_response.xml -log C:\temp\silent_was85_install.xml
```

Important: You can also include the **showProgress** argument to see the progress of the silent installation in the command line.

2. Run the **imcl listInstalledPackages** command to verify that the package was installed. See Example 2-10.

Example 2-10 Listing the installed package

```
C:\IBM\InstallationManager\ eclipse\tools\>imcl.exe listInstalledPackages
com.ibm.cic.agent_1.5.2000.20120223_0907
com.ibm.websphere.ND.v85_8.5.0.20120501_1108
```

3. Examine the installation log file to verify that the installation was successful. You can view the log file using a text editor, a browser, or the Installation Manager GUI. To use the Installation Manager GUI, click **File** → **View Log**, and then select the log file to be viewed.

2.6 Installing additional software

The WebSphere Application Server V8.5 package comes with additional software that can be used with the product, such as IBM WebSphere HTTP Server V8.5, WebSphere Plug-ins, WebSphere Customization Toolbox V8.5, and the Application Client for WebSphere Application Server 8.5.

This section guides you through installing the WebSphere Customization Toolbox and the Application Client. For installation of the HTTP server software, refer to Chapter 12, “Configuring and managing web servers” on page 417.

2.6.1 WebSphere Customization Toolbox

The WebSphere Customization Toolbox (WCT) existed in WebSphere Application Server Version 7. It was used only to configure z/OS servers. WebSphere Application Server V8 enhanced the functionality of the tools for managing, configuring, and migrating various parts of the product environment.

The toolbox is available as two different offerings, each with various combinations of tools for different platforms:

- ▶ Embedded WCT
- ▶ Stand-alone WCT

Each WebSphere Customization Toolbox offering is installed, modified, rolled back, or updated using Installation Manager. It can be installed silently using the command line or interactively using the GUI or console modes.

Embedded WebSphere Customization Toolbox

The embedded WebSphere Customization Toolbox comes as a part of the WebSphere Application Server V8.5 package. Tools included in the embedded version are:

- ▶ Profile Management Tool (PMT)
- ▶ Configuration Migration Tool (CMT)

With the embedded WebSphere Customization Toolbox offering, both of the included tools are automatically installed. They are not listed for selection in Installation Manager.

Important: In WebSphere Application Server V8 and later versions, IBM AIX 64-bit systems can use GUI-based tools. However, AIX requires that the GNU Toolkit (GTK) be installed to run the GUI. If you do not have the GTK installed, you receive an Eclipse error when trying to launch the GUI. For details about configuring AIX to support the GUI, refer to the product information center at this website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tins_aixsetup.html

Other supported platforms include HPUX 64-bit, Windows 2000 (32-bit and 64-bit), Linux on x86 (32-bit and 64-bit), Linux on Power (32-bit and 64-bit), Linux on s390 (32-bit and 64-bit), Solaris Sparc, and AIX (32-bit and 64-bit).

Stand-alone WebSphere Customization Toolbox

The stand-alone WebSphere Customization Toolbox comes as its own product offering. It can be found in the WebSphere Application Server V8.5 supplements package and is installed using Installation Manager.

Tools included in the stand-alone WebSphere Customization Toolbox are the following:

- ▶ Web Server Plug-ins Configuration Tool
- ▶ z/OS Profile Management Tool (zPMT)
- ▶ z/OS Migration Management Tool (zMMT)
- ▶ Remote Installation Tool for IBM i

When installing the stand-alone WebSphere Customization Toolbox, you select the tools you want to install. However, zPMT and zMMT must be installed together because they have dependencies that Installation Manager recognizes.

Supported platforms include HPUX, Windows, Linux on x86, Linux on Power, Linux on s390, Solaris (Sparc and x86), and AIX. Regardless of whether it is installed on a 32-bit or 64-bit operating system, the stand-alone toolbox operates as a 32-bit component on a 32-bit JDK.

Toolbox tools

Tools in the two WebSphere Customization Toolbox offerings include:

- ▶ Profile Management Tool (PMT)

The Profile Management Tool provides a user interface for profile creation and augmentation. To learn more about the Profile Management Tool, refer to Chapter 3, “Working with profiles on distributed systems” on page 59.

- ▶ Configuration Migration Tool (CMT)

The Configuration Migration Tool provides a graphical interface to the migration tools included in WebSphere Application Server.

- ▶ Web Server Plug-ins Configuration Tool (PCT)

The Web Server Plug-ins Configuration Tool allows you to configure your web server plug-ins on distributed systems for communicating with the application server. If possible, it creates a web server configuration definition in the application server. To learn more about the Web Server Plug-ins Tool, refer to 12.3, “Web server configuration using the WebSphere Customization Toolbox” on page 425.

- ▶ Remote Installation Tool for IBM i

The Remote Installation Tool for IBM i can be used on an Intel-based operating system only to install Installation Manager or a WebSphere Application Server component from a Windows machine to a target IBM i system.
- ▶ z/OS Profile Management Tool (zPMT)

The z/OS Profile Management Tool can be used on an Intel-based or Linux operating system to generate jobs and instructions for creating profiles for WebSphere Application Server on z/OS systems. The jobs are then uploaded and run on a target z/OS system. To discover more about the z/OS Profile Management Tool, refer to Chapter 5, “Working with profiles on z/OS systems” on page 121.
- ▶ z/OS Migration Management Tool (zMMT)

The z/OS Migration Management Tool can be used on an Intel-based or Linux operating system to create migration definitions that are used to migrate a WebSphere Application Server on z/OS node. Each migration definition is a set of jobs and instructions that can then be uploaded and run on a target z/OS system.

Installing the stand-alone WebSphere Customization Toolbox

Use Installation Manager to install the stand-alone WebSphere Customization Toolbox offering. Make sure that the Installation Manager preferences point to a repository containing the WebSphere Customization Toolbox.

You can install the toolbox using the GUI, command-line, or silent modes. For silent mode installations, Installation Manager supports the creation of the necessary response file.

Installing WebSphere Customization Toolbox is similar to installing other packages using Installation Manager, including the option of selecting the tools you want to install in the stand-alone offering. If you do not install all of the available tools during the initial installation process, you can later modify the installation to add the other tools.

To install the stand-alone WebSphere Customization Toolbox:

1. Start Installation Manger.
2. Add the repository to the WebSphere supplements package.
3. In the main Installation Manager window, click **Install**.
4. From the listed offerings, choose WebSphere Customization Toolbox, as illustrated on Figure 2-10. Click **Next**.

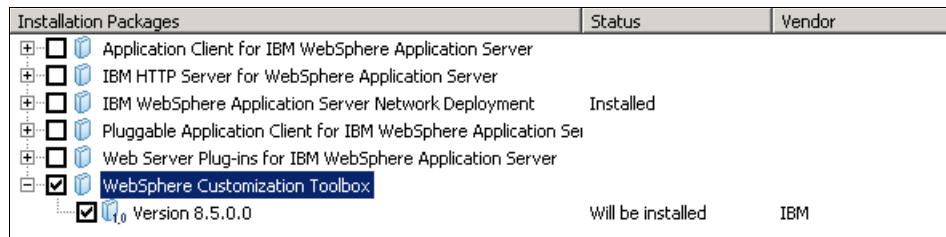


Figure 2-10 Installing a WebSphere Customization Toolbox package

5. Accept the license agreement, and click **Next**.
6. Provide the installation directory for the package, and ensure that you used the Create a new package group option, and then click **Next**.
7. Choose from among the features shipped with the WCT. Click **Next**. For this example, only the tools for managing web servers are needed (refer to Figure 2-11 on page 55).

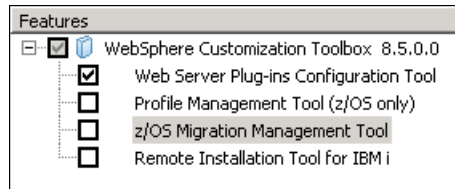


Figure 2-11 Selecting WebSphere Customization Toolbox features to install

8. Review the summary, and click **Install**.
9. When the installation successfully finishes, you can choose to start the WebSphere Customization Toolbox or close the window. Choose your preferred option, and click **Finish**.

Using the WebSphere Customization Toolbox

There are different ways to start the WebSphere Customization Toolbox, depending on the offering that was installed and the operating system. On Windows and Linux platforms, a start menu shortcut is created for both the embedded and stand-alone offering. Other options for starting the toolbox are:

- ▶ Embedded offering: Launch **wct** from `<was_install>\bin\ProfileManagement\WCT\`
- ▶ Stand-alone offering: Launch **wct** from `<wct_install>\WCT\`

Figure 2-12 shows the embedded WebSphere Customization Toolbox V8.5.

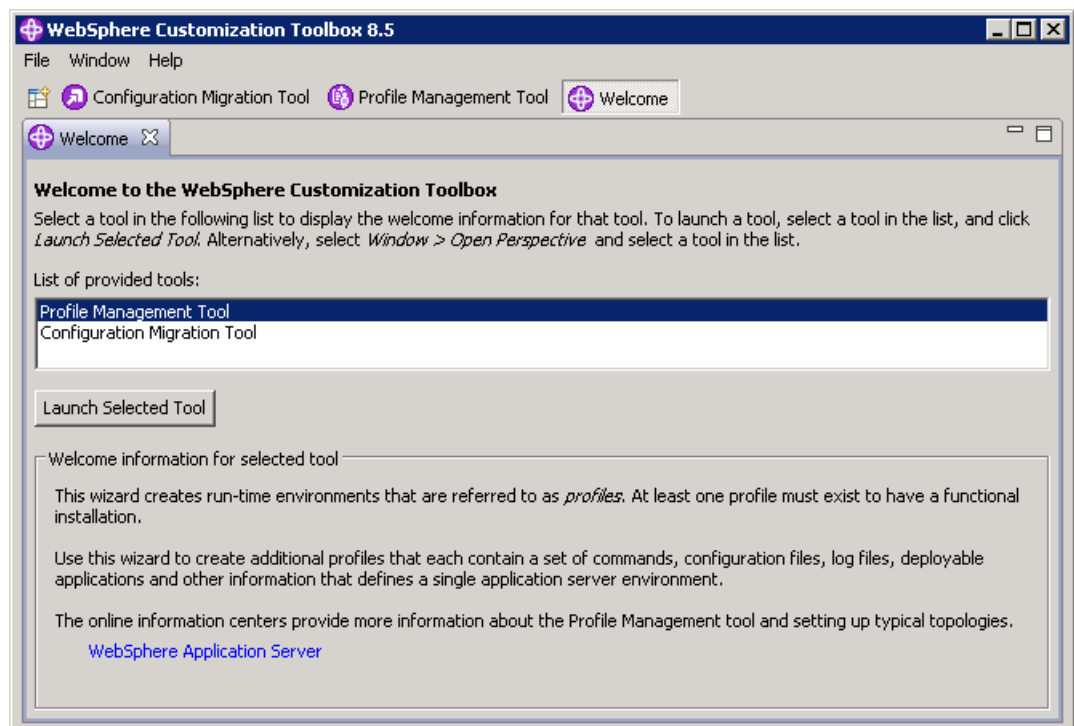


Figure 2-12 Embedded WebSphere Customization Toolbox V8.5

Figure 2-13 on page 56 shows the stand-alone WebSphere Customization Toolbox V8.5. Note that only the Web Server Plug-ins Configuration Tool is available because it is all that was installed in this example. You can modify this package at any time to add or remove additional features when needed.

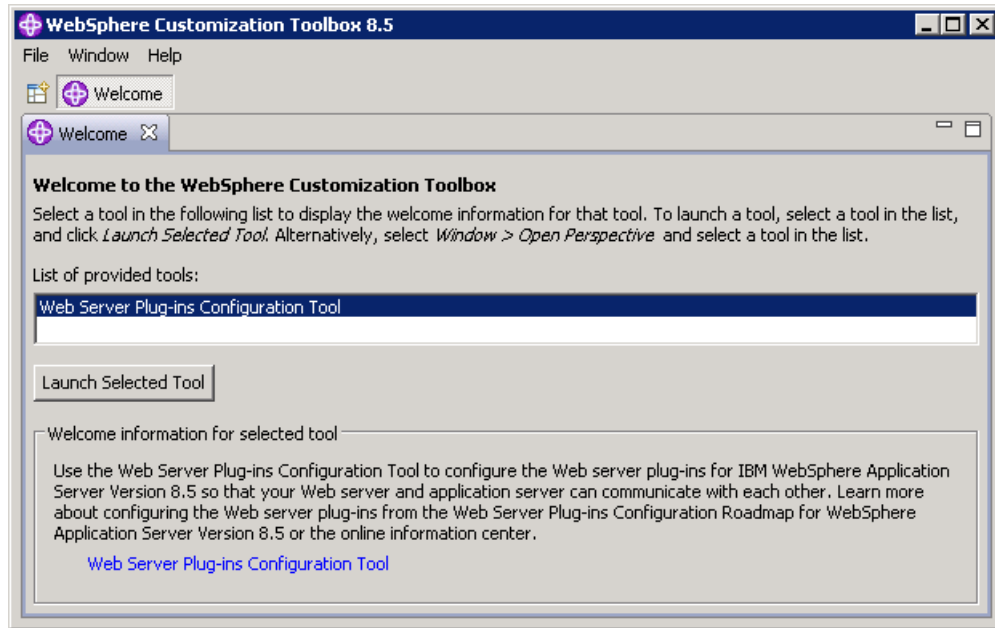


Figure 2-13 Stand-alone WebSphere Customization Toolbox V8.5

The stand-alone WebSphere Customization Toolbox also includes a command-line utility that launches a command-line version of the Web Server Plug-ins Configuration Tool (PCT). For further information about using this utility, refer to the product information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/tins_pctcl_using.html

2.6.2 Application Client for WebSphere Application Server V8.5

The Application Client for WebSphere Application Server V8.5 package allows you to create applications that run separately from your application server. A client application uses the framework provided by an underlying client to access the resources provided by WebSphere Application Server.

Application Client offerings

The Application Client for WebSphere Application Server is packaged with the following components:

- ▶ Java Runtime Environment (JRE) (or an optional full Software Development Kit) that IBM provides.
- ▶ The runtime environment for Java EE client applications (that uses services provided by the Java EE Client Container)
- ▶ The runtime environment for Java thin client applications (Java SE applications that do not use services provided by the Java EE Client Container)
- ▶ An ActiveX to EJB Bridge for ActiveX programs to access enterprise beans through a set of ActiveX automation objects (Windows only)
- ▶ IBM plug-in for Java platforms for Applet client applications (Windows only)
- ▶ A variety of stand-alone thin clients, provided as embeddable JAR files

To learn more about this offering, refer to the product information center website at this address:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/cli_clientapps.html

Installing the Application Client package

To install the Application Client for WebSphere Application Server V8.5:

1. Start Installation Manger.
2. Add the repository to the WebSphere supplements package.
3. In the main Installation Manager window, click **Install**.
4. From the list of offerings. choose the Application Client for WebSphere Application Server V8.5 (refer to Figure 2-14), and click **Next**.

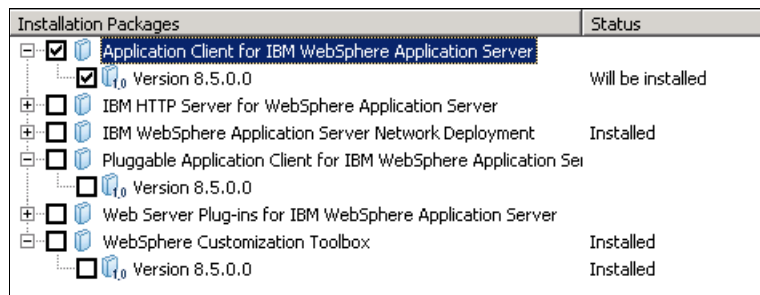


Figure 2-14 Installation of the Application Client for WebSphere Application Server package

Deprecated feature: The Pluggable Application Client is deprecated. It is replaced by the stand-alone, thin client, IBM Thin Client for EJB.

The Pluggable Application Client runs only on the Windows platform and requires that you previously installed the Sun Java Runtime Environment (JRE) files. In all other aspects, the Pluggable Application Client and the Java thin application client are similar.

5. Accept the license agreement, and click **Next**.
6. Provide the installation directory for the package, and ensure that you chose the Create a new package group option, and then click **Next**.
7. In the Features window (see Figure 2-15 on page 58), select the features you want to install from the lists of available features. For this example installation, Samples features are used.

Note: Samples features contain source code and executable files that can be helpful when working with Application Client for WebSphere Application Server.



Figure 2-15 Selecting features of Application Client for WebSphere Application Server V8.5

8. Specify the host name and bootstrap port of the WebSphere Application Server to which you want to connect using the Application Client (see Figure 2-16), and then click **Next**.

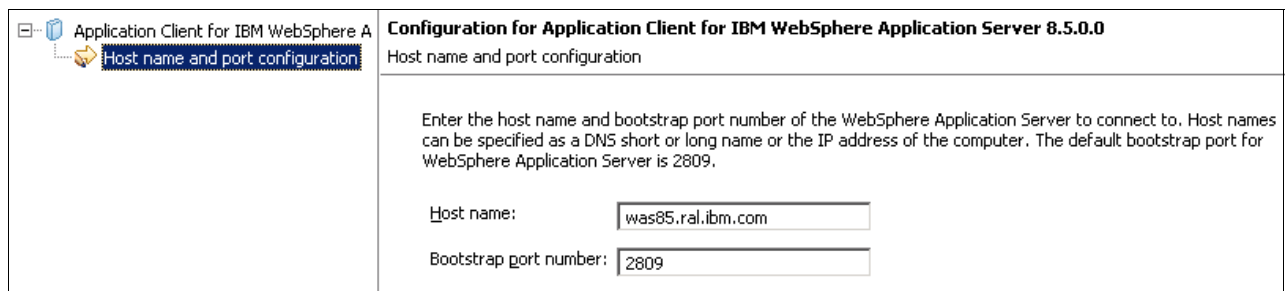


Figure 2-16 Configuring WebSphere Application Server for the Application Client

9. Review the summary, and then click **Install**.
10. Click **Finish** when the installation process ends.



Working with profiles on distributed systems

Installing a WebSphere Application Server environment requires careful planning. A major decision point is the topology for the system. You must consider, for example, whether you should use a stand-alone server, a distributed managed server, or a flexible management environment.

Planning for a topology design is covered in *WebSphere Application Server V8.5: Concepts, Planning and Design Guide*, SG24-8022. That book is designed to help you select a topology and develop a clear idea of what steps are needed to set up the environment of your choice.

The purpose of this chapter is to help you build your initial WebSphere Application Server environment after you install the product. In this chapter, we cover the following topics:

- ▶ Types of profiles
- ▶ Planning for profiles
- ▶ Building systems with profiles
- ▶ Managing profiles with the command line

3.1 Types of profiles

The WebSphere Application Server installation process simply lays down a set of core product files required for the runtime processes. After installation, you need to create one or more *profiles* that define the run time to have a functional system. The core product files are shared among the runtime components defined by these profiles.

The next section provides an overall description about each profile type.

3.1.1 Application server profile

The application server profile defines a single stand-alone application server. Using this profile gives you an application server that can be run in unmanaged (stand-alone) mode or managed mode (by federating it with the administrative agent profile). The environment has the following characteristics:

- ▶ The profile consists of one cell, one node, and one server. The cell and node are not relevant in terms of administration, but you see them when you administer the server through the administrative console scopes.
- ▶ The server uses a dedicated, built-in administrative console.

The primary uses for this type of profile are:

- ▶ To build a stand-alone server using the Base or Express installation packages.
- ▶ To build a stand-alone server in a Network Deployment installation that is not managed by the deployment manager (a test machine, for example).
- ▶ To build a server in a distributed server environment to be later federated and managed by the deployment manager. When you federate this node, the default cell becomes obsolete, the node is added to the deployment manager cell, and the administrative console is removed from the application server.

3.1.2 Deployment manager profile

The deployment manager profile defines a deployment manager in a distributed server environment. Although you can conceivably have the Network Deployment edition and run only stand-alone servers, this action bypasses the primary advantages of Network Deployment, which is workload management, failover, and central administration.

In a Network Deployment environment, create one deployment manager profile for each cell. This setup gives you:

- ▶ A cell for the administrative domain
- ▶ A node for the deployment manager
- ▶ A deployment manager with an administrative console
- ▶ No application servers

After you have the deployment manager, you can:

- ▶ Federate nodes built either from existing application server profiles or custom profiles.
- ▶ Create new application servers and clusters on the nodes from the administrative console.

3.1.3 Custom profile

A custom profile is an empty node without any server instance that is intended for federation to a deployment manager. After federation, the deployment manager uses it as a target on which it can create, for example, application server profile instances.

3.1.4 Cell profile

A cell profile combines two profiles: a deployment manager profile and an application server profile. In this case, the deployment manager and application server reside on the same system, and the application server profile is already federated to the deployment manager cell.

Using this type of profile is a good way to quickly set up a distributed server environment. It can be useful for test environments that can have all nodes on one test system.

3.1.5 Administrative agent profile

The administrative agent profile provides enhanced management capabilities for stand-alone application servers. An administrative agent profile is created on the same node as the stand-alone servers and can manage only servers on that node. The node configuration for each stand-alone server is separate from any other servers on the system, but it is managed using the administrative console on the administrative agent, as illustrated on Figure 3-1 on page 62. When a base application server registers with an administrative agent, much of the administrative code that was in the base server is now consumed by the administrative agent.

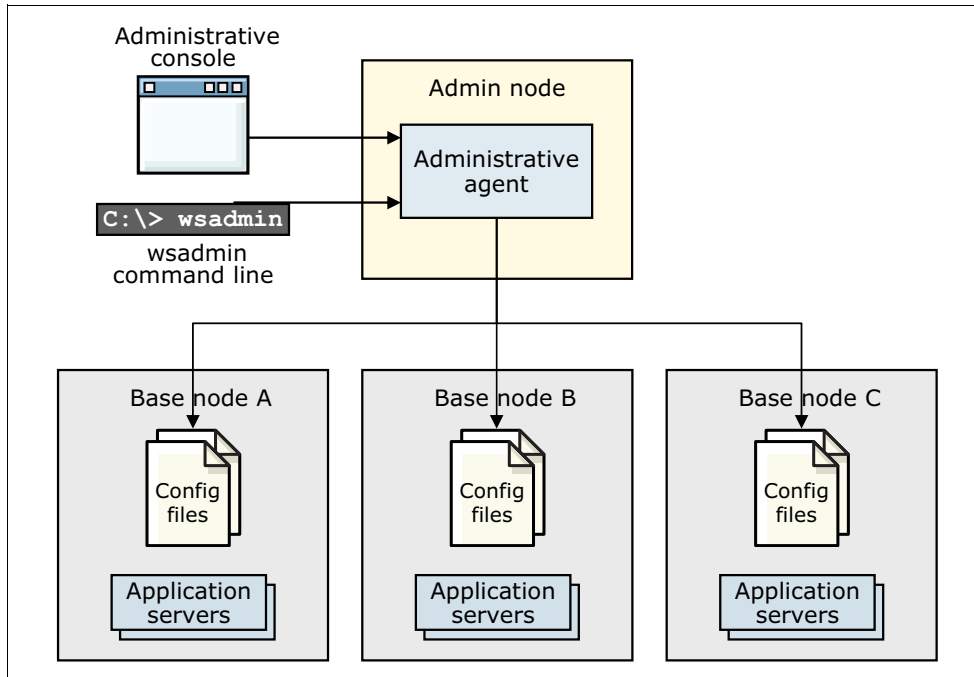


Figure 3-1 Overview of an administrative agent profile architecture

3.1.6 Job manager profile

A job manager is defined by a job manager profile. The job manager's primary purpose is to support flexible management of WebSphere Application Server profiles and to queue jobs to registered servers.

For stand-alone application servers, to participate in flexible management, first they are registered with the administrative agent, as described in 3.1.5, "Administrative agent profile" on page 61. After that, the administrative agent registers the node for the application server with the job manager.

If a deployment manager wants to participate in an environment controlled by a job manager, the deployment manager registers directly with the job manager. No administrative agent is involved in this case.

Liberty profile: The Liberty profile can also participate in the flexible management. Using job manager, it can be installed and managed just like the stand-alone profile. For more information, see *WebSphere Application Server V8.5 Administration Guide for the Liberty Profile*, SG24-8170.

Both the deployment manager and administrative agents retain autonomy and can be managed without the job manager. A job manager can submit jobs to one or more administrative agents or deployment managers, and an administrative agent or a deployment manager can register with more than one job manager, if desired.

The units of work that are handled by the flexible management environment are known as jobs. The semantics of these jobs are typically straightforward and require few parameters. The jobs are processed asynchronously and can have an activation time, expiration time, and a recurrence indicator. You can also specify to send an email notification upon completion of a job. Additionally, you can view the current status of a job by issuing a status command.

Figure 3-2 presents a sample flexible management topology, where the single job manager indirectly manages multiple stand-alone servers using administrative agents and also directly manages multiple deployment manager cells.

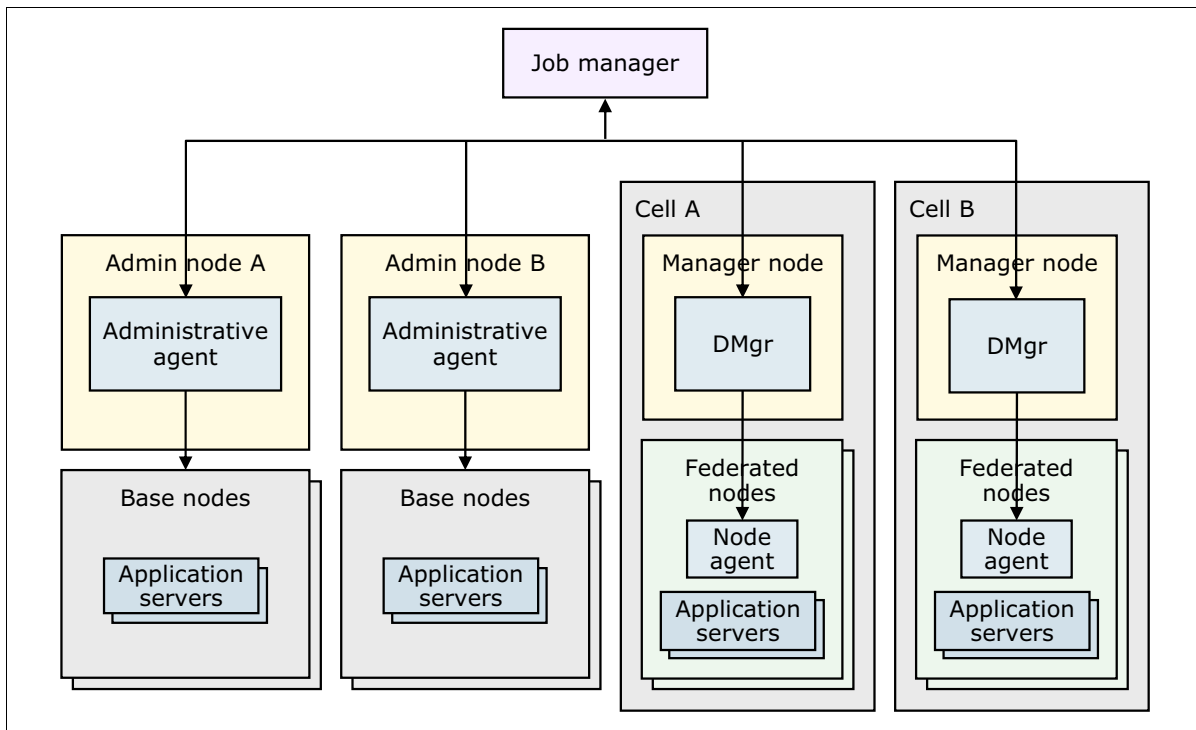


Figure 3-2 Overview of a flexible management topology using job manager to administer multiple environments

Note: From WebSphere Application Server V8, you can also use some of the job manager actions from a deployment manager profile web console.

3.2 Planning for profiles

Before creating WebSphere Application Server profiles, remember that a minimum amount of space must be available in the directory where you create a profile. This minimum requirements are documented in the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rpro_diskspace.html

Profiles grow when applications and associated log files are created, and therefore these increases must be considered at the planning stages.

Errors can occur when you do not provide enough space to create a profile. Verify that you have file system reserves in addition to the minimum space required for a particular profile, for log files, and temporary files. The amount and size of these files can vary on your configuration and on used applications.

3.3 Building systems with profiles

Profiles can be created at any time during or after installation using graphical or command-line tools. WebSphere Application Server provides the following profile management tools:

- ▶ The **manageprofiles** command: A command-line interface for profile management functions.
- ▶ Profile Management Tool (PMT): A GUI interface delivered by the WebSphere Customization Toolbox. This tool gathers user input and invokes the **manageprofiles** command-line tool to manage the profiles for you.
- ▶ Administrative console of the deployment manager or the job manager profile, which can create profiles on remote machines.

Note: When you use the Profile Management Tool with the Motif graphical user interface on the Solaris operating system, the default size of the Profile Management Tool might be too small to view all of the messages and buttons of the Profile Management Tool. To fix the problem, add the following lines to the `app_server_root/.Xdefaults` file:

```
Eclipse*spacing:0  
Eclipse*fontList:-misc-fixed-medium-r-normal-*-10-100-75-75-c-60-iso8859-1
```

After adding the lines, run the following command before launching the Profile Management Tool:

```
xrdb -load user_home/.Xdefaults
```

Note: Each profile you create is registered in a profile registry, which is under:

```
install_root/properties/profileRegistry.xml
```

This section shows how to create different profile types using both methods.

3.3.1 Starting the WebSphere Customization Toolbox Profile Management Tool

There are several ways to start the WebSphere Customization Toolbox:

- ▶ At the end of the installation process using the Installation Manager install wizard, select the option to start the Profile Management Tool to create a profile.
- ▶ Select the WebSphere Customization Toolbox option from the programs list:
 - Windows only:
From the Start menu, select **Start** → **Programs** → **IBM WebSphere** → **Application Server Network Deployment V8.5** → **WebSphere Customization Toolbox**.
 - For Linux only:
From the operating system menu to start programs, select **Applications** → **IBM WebSphere Application Server Network Deployment V8.5** → **WebSphere Customization Toolbox** → **Profile Management Tool**.
 - For all platforms:
Use the **wct.bat** or **wct.sh** command located in the `<install_root>/bin/ProfileManagement` directory.

Note: A `pmt.bat (sh)` shell script will also start the WebSphere Customization Toolbox, but it is provided only for backward compatibility. It is deprecated from WebSphere Application Server V8.

3.3.2 Common steps for all profiles

Many of the options that are available when you create a profile are the same, regardless of the type of profile. This section introduces the common steps that are used while defining different profiles.

Environment selection

The Profile Management Tool provides multiple profile templates, including the cell template, which has the ability to create a cell in a single step. During profile creation, you are asked to select the type of profile to create, as shown in Figure 3-3.

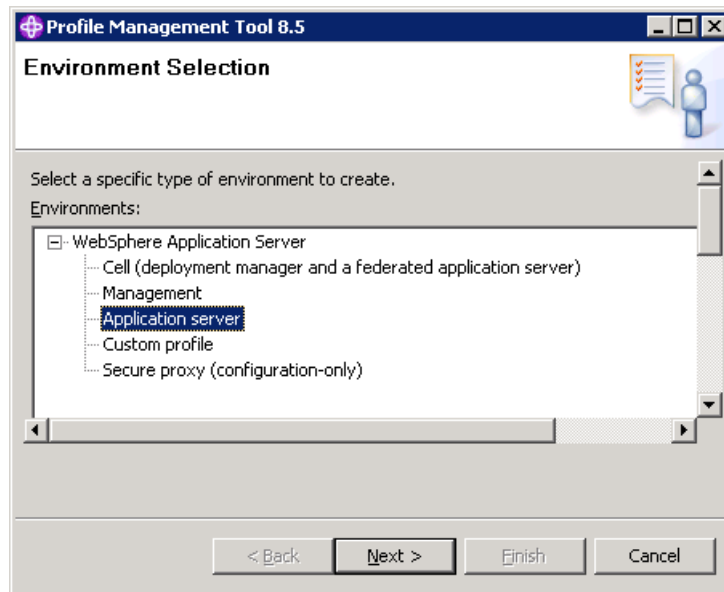


Figure 3-3 Profile type selection

You can select the following profiles:

- ▶ Cell (deployment and a federated application server)
- ▶ Management:
 - Administrative agent
 - Deployment manager
 - Job manager
- ▶ Application server
- ▶ Custom profile
- ▶ Secure proxy (configuration-only)

Profile creation options

While creating profiles, you are presented with a choice (Figure 3-4 on page 66) of following the *Typical* path, where a set of default values for most settings are used, or an *Advanced* path, which lets you specify values for more options.

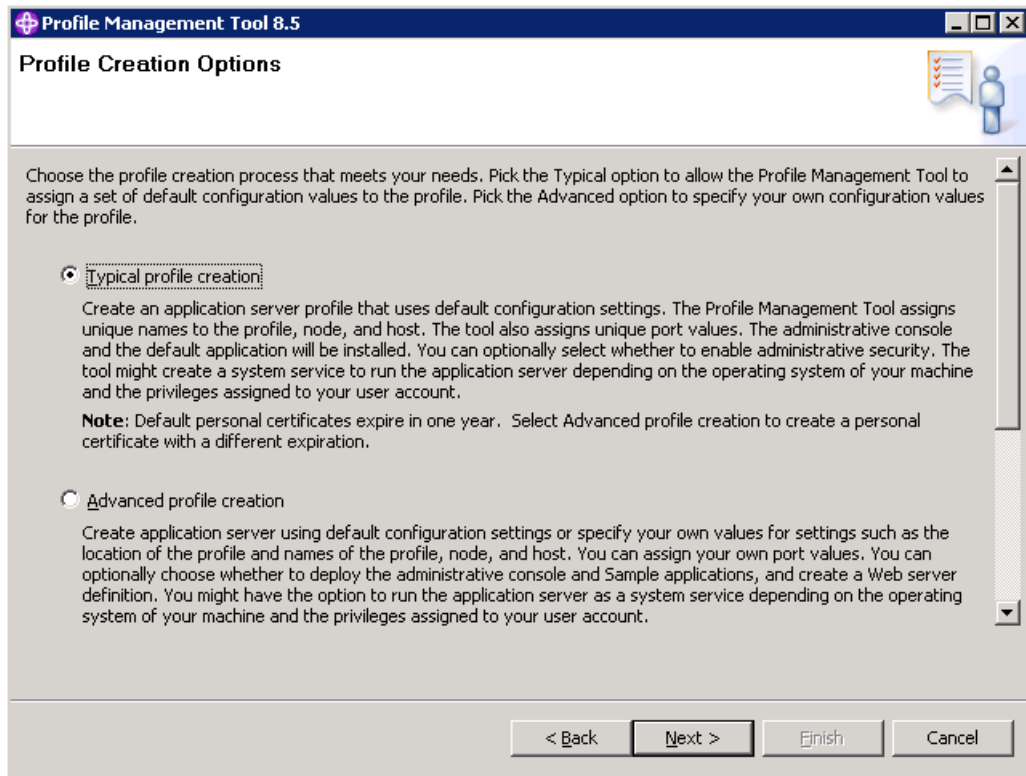


Figure 3-4 Profile creation path selection

The *Advanced* path is preferred because it gives you additional control over names and settings.

Administrative security

All profiles except the custom profile can be secured by enabling the administrative security. This setting prevents users from gaining unauthorized access to the administrative console. If you enable administrative security during profile creation, you are asked for a user ID and password that are added to a file-based user registry with the Administrator role, as shown on Figure 3-5 on page 67.

Enabling administrative security: Consider enabling administrative security. An XML file-based user repository is created during profile creation and can be later federated with other repositories to provide a robust user registry for both administrative and application security.

If you do not want to use the file-based repository, do not enable administrative security during profile creation and configure it manually afterwards.

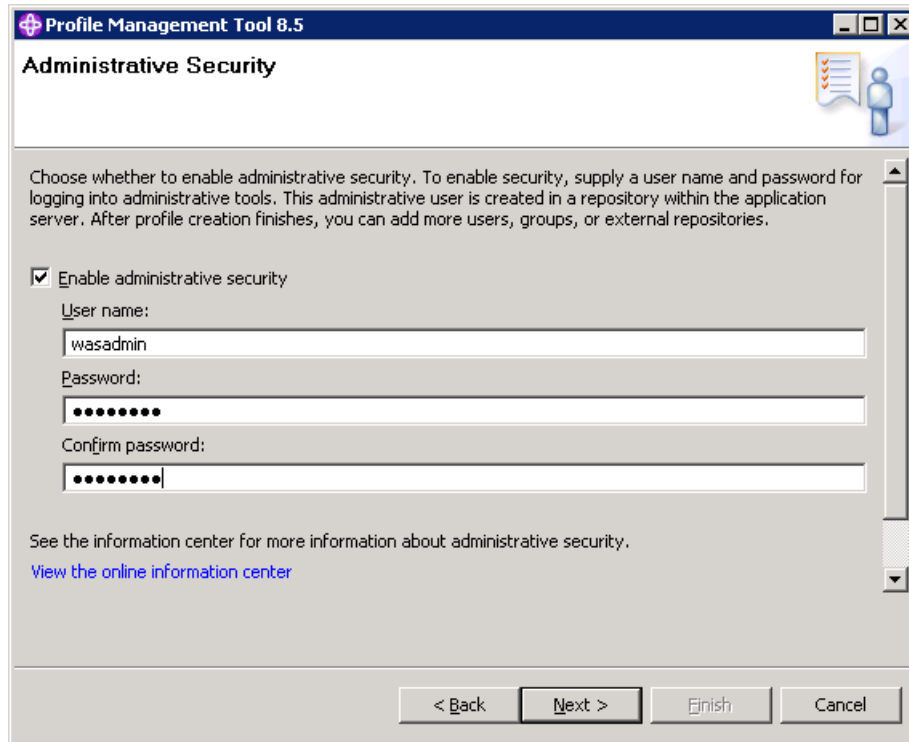


Figure 3-5 Configuration of administrative security during profile creation

Note: If you are going to create a job manager and register a deployment manager, keep in mind that you cannot register a deployment manager that has security enabled to a job manager that does not. So, plan your administrative security policy across the entire WebSphere environment.

You can find more information about administrative security in 6.2, “Securing the administrative console” on page 211.

Certificates

Each profile contains a unique chained certificate signed by a unique long-lived root certificate that is generated when the profile was created. When a profile is federated to a deployment manager, the signer for the root signing certificate is added to the common truststore for the cell, establishing trust for all certificates signed by that root certificate.

For a full description of the certificates and the keystore password, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/csec_7ssldefault_chainedcert_config.html

Two windows are used during profile creation to manage the import or creation of these certificates. The first window (Figure 3-6 on page 68) allows you to generate the certificates or import existing certificates.

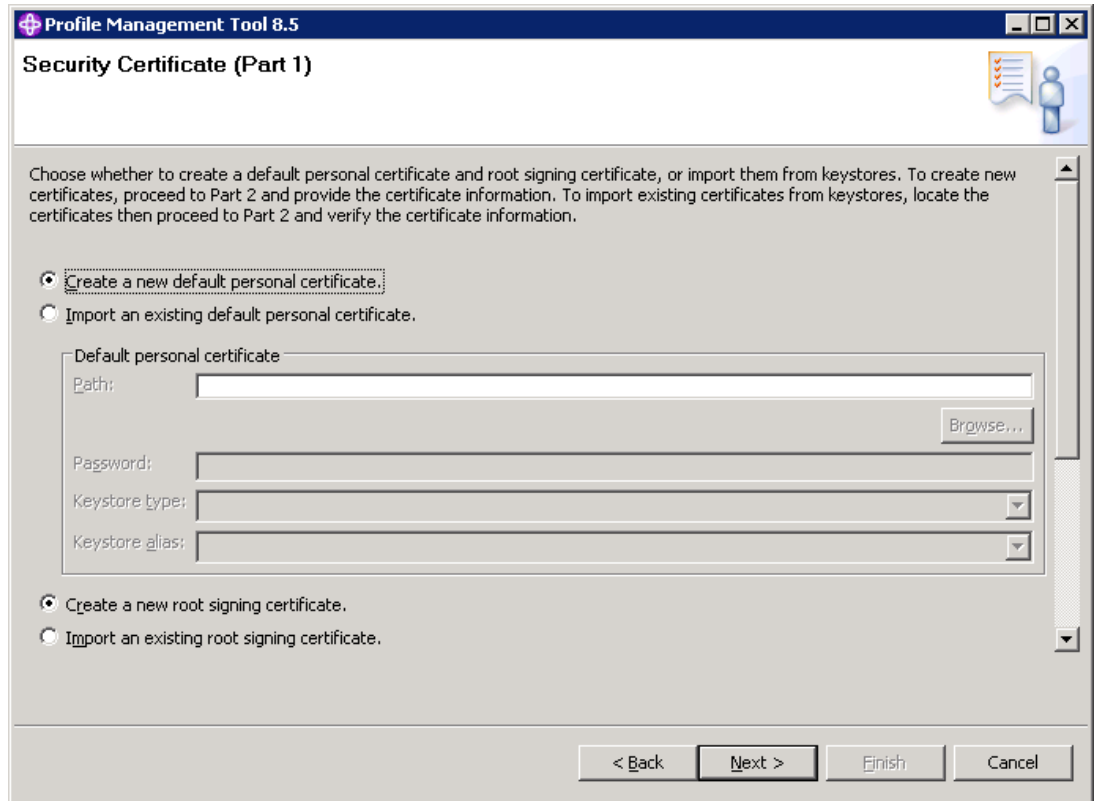


Figure 3-6 Creating self-signed certificates or importing existing personal or root certificates

The second window (Figure 3-7 on page 69) is used to modify the certificate information to create new certificates during profile creation. The auto-generated DN's for the certificates are usually long, so you might want to change them.

Important: The default password for the generated keystores is WebAS. Consider changing it with your own password to increase the server security.

Also review the expiration period of the certificates. Note that the root certificate minimal expiration period is 15 years.

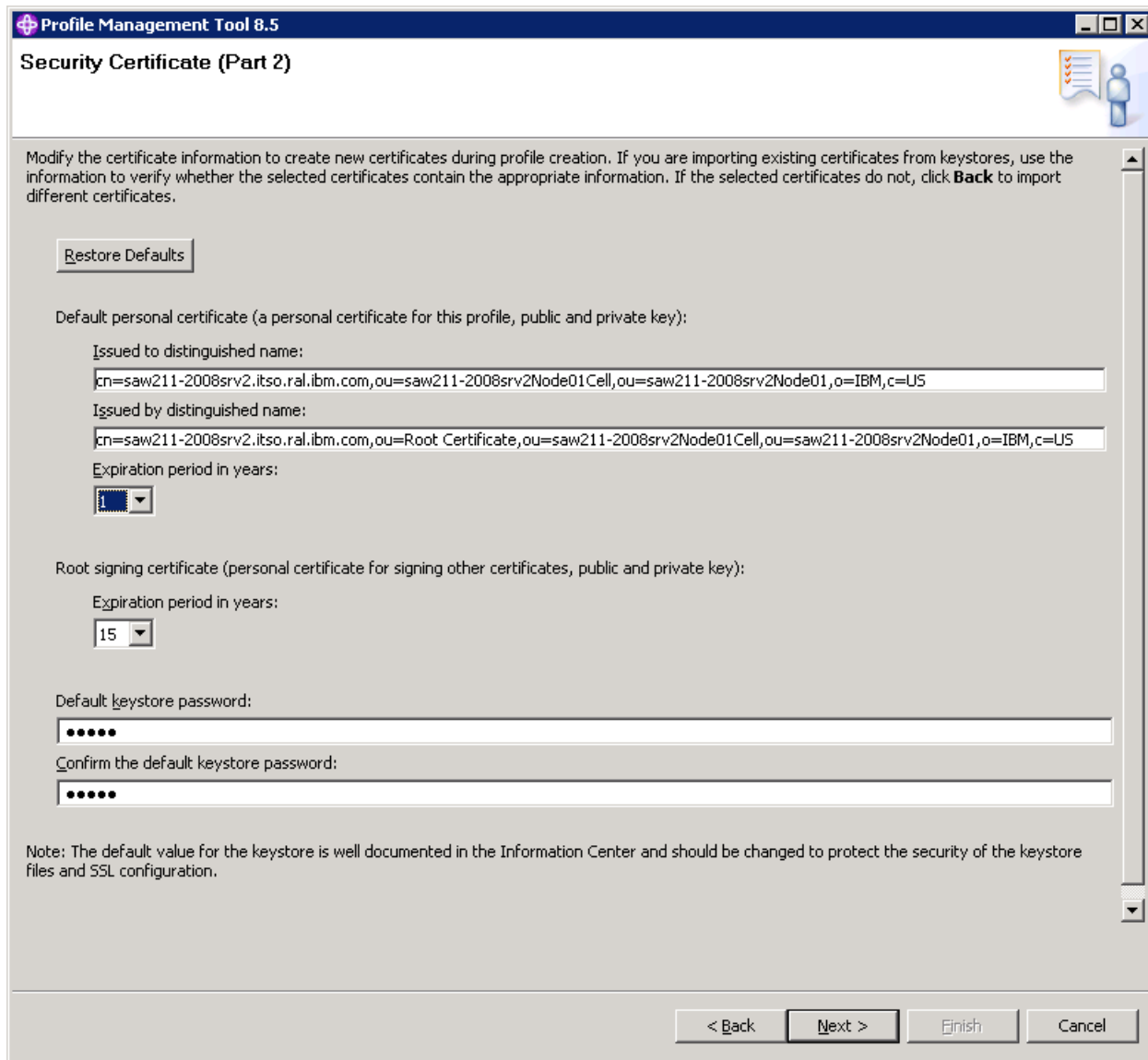


Figure 3-7 Modifying certificate information during profile creation

Port assignments

Every process uses a set of ports at run time. These ports must be unique to a system. For the default port assignment on a distributed platform, see the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.migration.n.d.doc/ae/rmig_portnumber.html

The profile management tool wizard assigns unique port numbers to a profile to omit port conflicts when multiple profiles are installed on the same system. Ensure that there are no port conflicts with other software installed on the same system. Figure 3-8 on page 70 shows a sample of ports generated by the installer for an application server profile.

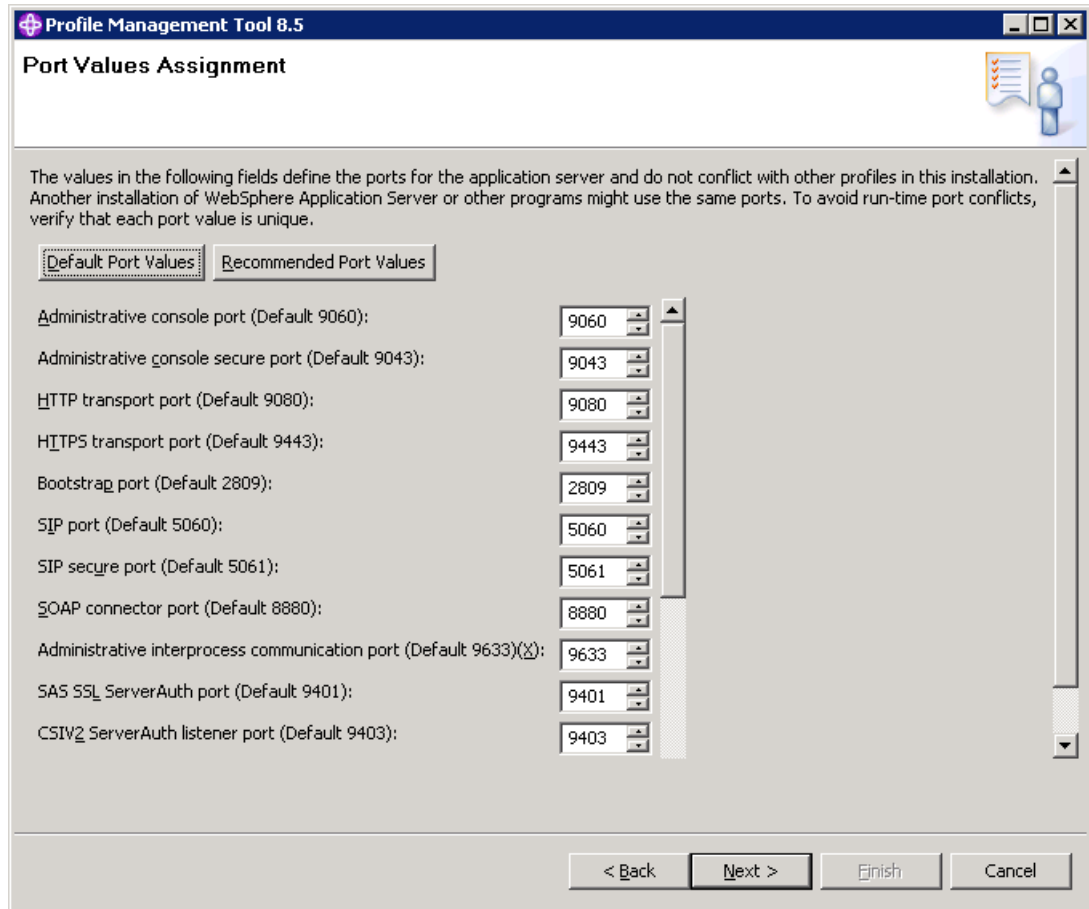


Figure 3-8 Assigning the server ports

When you take the Advanced path through the profile wizard, you have three options:

- ▶ Default Port Values: Use the default set of port numbers.
- ▶ Recommended Port Values: Use the recommended set of port numbers. These are selected as unique to the WebSphere installation.
- ▶ Manually customize the port numbers.

After profile creation, you can obtain port numbers by looking in the following file:

- ▶ `profile_home/properties/portdef.props`
- ▶ `profile_home/logs/AboutThisProfile.txt`

For example:

- ▶ `C:\IBM\WebSphere\AppServer\profiles\AppSrv01\properties\portdef.props`
- ▶ `C:\IBM\WebSphere\AppServer\profiles\AppSrv01\logs\AboutThisProfile.txt`

Information: You can also use the PortManagement command group for the AdminTask object in `wsadmin` to list application and server ports and modify server ports. For more information about the PortManagement command group, refer to the information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rxml_atportmgt.html

Note the following port numbers for later use:

- ▶ *SOAP connector port:* If you plan to federate this node to a deployment manager later using the deployment manager administrator console, you must know this port number. This port is also the port that you connect to when using the `wsadmin` administration scripting interface.
- ▶ *Administrative console port:* You must know this port to access the administrative console. When you turn on security, you must know the *Administrative console secure port*.
- ▶ *HTTP transport port:* This port is used to access applications running on the server directly versus going through a web server.

Running as a service

When you create a profile on a Windows or Linux system, you have the option of running the application server as a Windows service. This action provides you with a simple way of automatically starting the server process with the system.

If you want to run the process as a Windows service, select the check box, and enter the values for the logon and startup type. Note that the window lists the user rights that the user ID you select needs to have. If the user ID does not have these rights, the wizard automatically adds them (see Figure 3-9 on page 72).

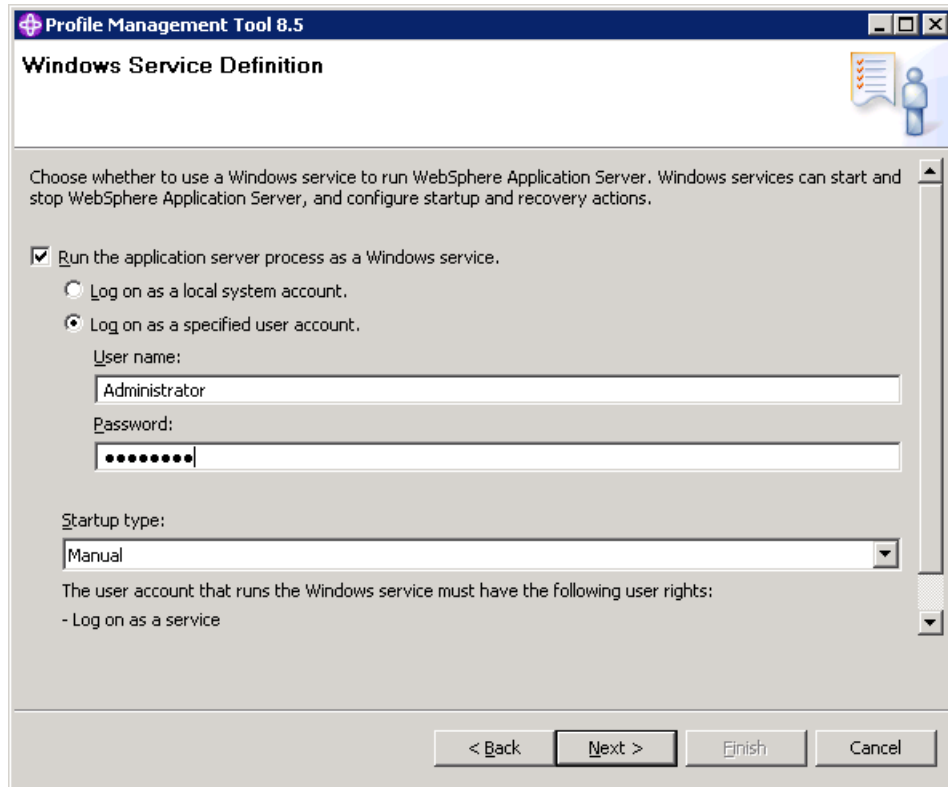


Figure 3-9 Configuring the profile to run as a Windows service

When you take the *Typical* path through the profile creation wizard on a Windows operating system, the default is to define the process as a Windows service. On Linux operating systems, the default setting is not to define the process as a service.

If you do not register the process as a Windows or Linux service during profile creation, you can do that later using the **WASService** command. This command enables to you create a service for a Java process on both Windows and Linux operating systems. Refer to the information center for more information:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rins_wasservice.html

Verification steps

At the end of the profile creation, you have the opportunity to start the First steps console (Figure 3-10 on page 73). This interface helps you start the server process and has other useful links, such as opening the administrative console, an information center and IBM Education Assistant link, starting the WebSphere Customization Toolbox, and installation verification.

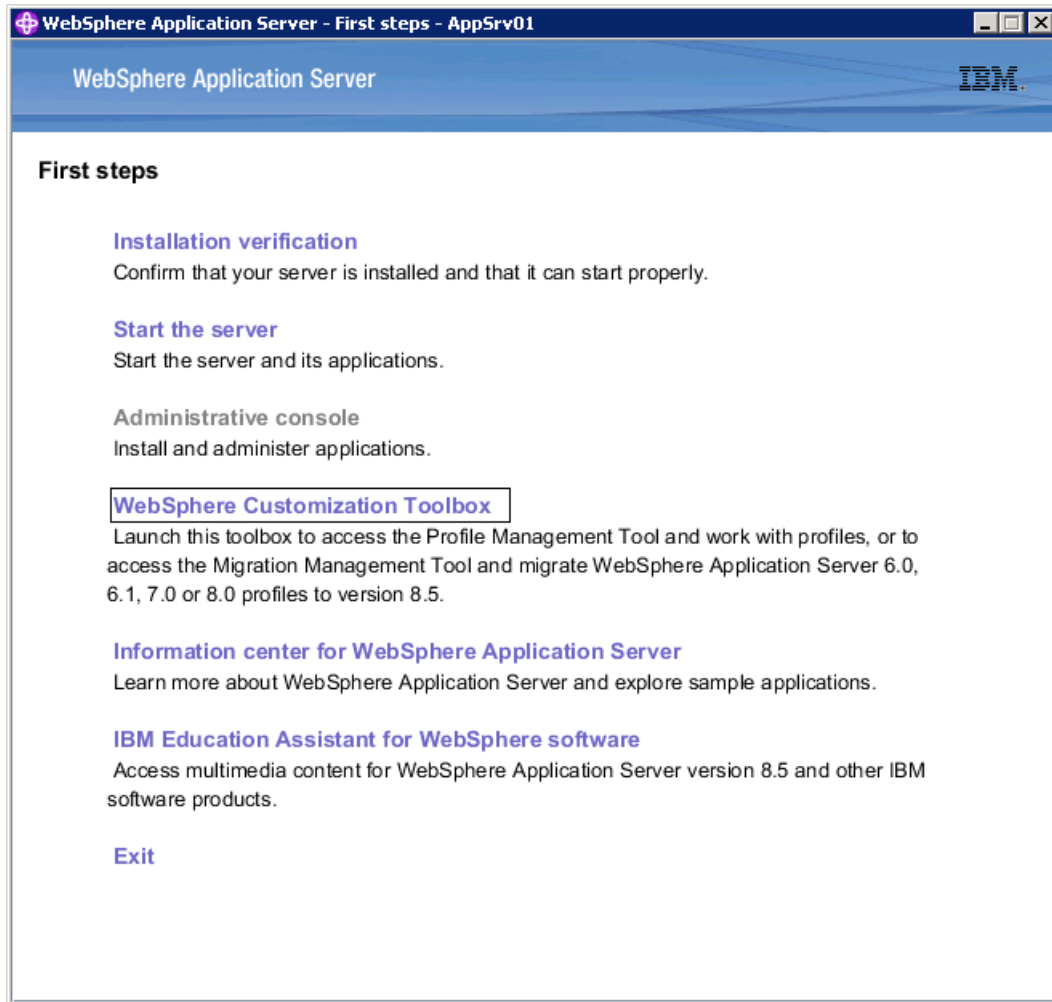


Figure 3-10 Using the First steps console after successful profile creation

To verify the new profile installation, you can use the **Installation verification** link. It launches the new profile and log information in a pop-up window, as illustrated on Figure 3-11 on page 74. You can also use it later by running the `firststeps` script from `profile_root` directory, for example:

```
C:\IBM\WebSphere\AppServer\profiles\AppSrv01\firststeps
```

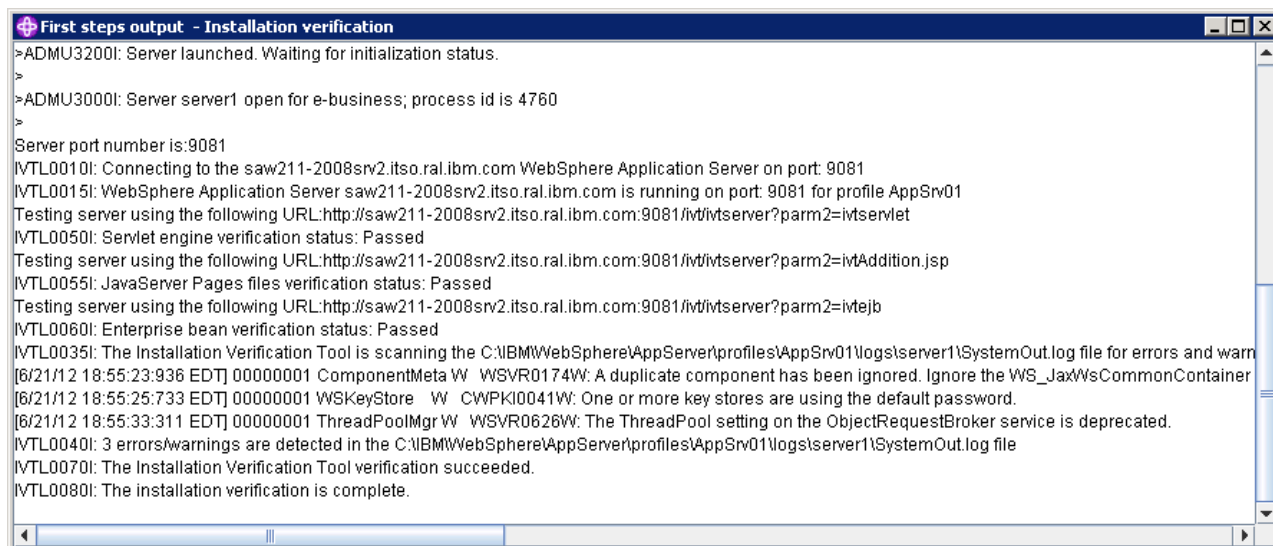


Figure 3-11 Using Installation verification tool to verify profile creation

To manually verify profile installation, refer to the following list of activities to perform:

- ▶ View the messages produced during profile creation:

install_root/logs/manageprofiles/*profile_name_create*.log

For example:

C:\IBM\WebSphere\AppServer\logs\manageprofiles\AppSrv01_create.log

- ▶ Review server logs for any problems, errors, or warnings:

- *profile_root*/logs/*server_name*/startServer.log
- *profile_root*/logs/*server_name*/SystemOut.log
- *profile_root*/logs/*server_name*/SystemErr.log

For example:

C:\IBM\WebSphere\AppServer\profiles\AppSrv01\logs\server1\SystemOut.log

- ▶ If applicable, log in to the administrative console hosted by the process. You can access the console from the First Steps menu or by accessing its URL from a web browser:

http://server_host:<admin_console_port>/ibm/console

For example (The administrative console port is selected during profile creation):

<http://localhost:9060/ibm/console/>

Click the **Log in** button. If security is not enabled, you can enter any or no user name. If you enabled security, enter the user ID and password you specified.

3.3.3 Creating an application server profile

An application server profile can be run stand-alone or can be later federated to a deployment manager cell for central management.

This section takes you through the steps of creating the application server profile using the WebSphere Customization Toolbox.

To create the profile:

1. Start the WebSphere Customization Toolbox, and open **Profile Management Tool**.
2. Click **Create**.
3. Select **Application server** as the profile type, and click **Next**.
4. Select whether to take a typical or advanced path to install the profile:
 - If Typical is selected, proceed to step 8 and continue from step 13.
 - If Advanced is selected, continue with the following steps.
5. Select both check boxes to deploy the administrative console and the default application.

Installing the administrative console is recommended. However, there might be some circumstances when you do not want to install an administrative console, such as if you plan to control all administrative tasks through scripting. If you do not install the administrative console during profile creation, you can install it using the `deployConsole.py` script at a later time.

Information: To install the administrative console after profile creation:

1. Navigate to `profile_root/bin`.
2. Start the server using the following command:
`startServer.bat(sh) server1`
3. Enter the following command to install the application:
`wsadmin.bat(sh) -lang jython -f deployConsole.py install`

If you configured administrative security during profile creation, you are prompted for an administrative user ID and password when running the `wsadmin` install command.

The second option of deploying the default application installs a default application that can be used to verify that your application server is running and serving application content. The default application contains a web module called `DefaultWebApplication` and an EJB module called `Increment`. The application includes a number of servlets that retrieve information that can be used for verification. For example you can try to invoke the `Snoop` servlet to verify if the application server is properly serving the content:

`http://localhost:9080/snoop`

You can find other sample applications that can be deployed after profile creation at the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.samples.doc/ae/welcome_samples.html

Click **Next**.

4. Enter a unique name for the profile or accept the default. Enter a unique directory path for the profile directory or accept the default, as shown on Figure 3-12. Click **Next**.

Note: From WebSphere Application Server V8, additional server runtime performance tuning option has been introduced during profile creation. You have three performance tuning options to choose from:

- ▶ Standard, which is the standard default configuration settings that are optimized for general purpose usage.
- ▶ Peak, which is appropriate for a production environment where application changes are rare and optimal runtime performance is important.
- ▶ Development, which is appropriate for a development environment where frequent application updates are performed and system resources are at a minimum. Do not use the development setting for production servers.

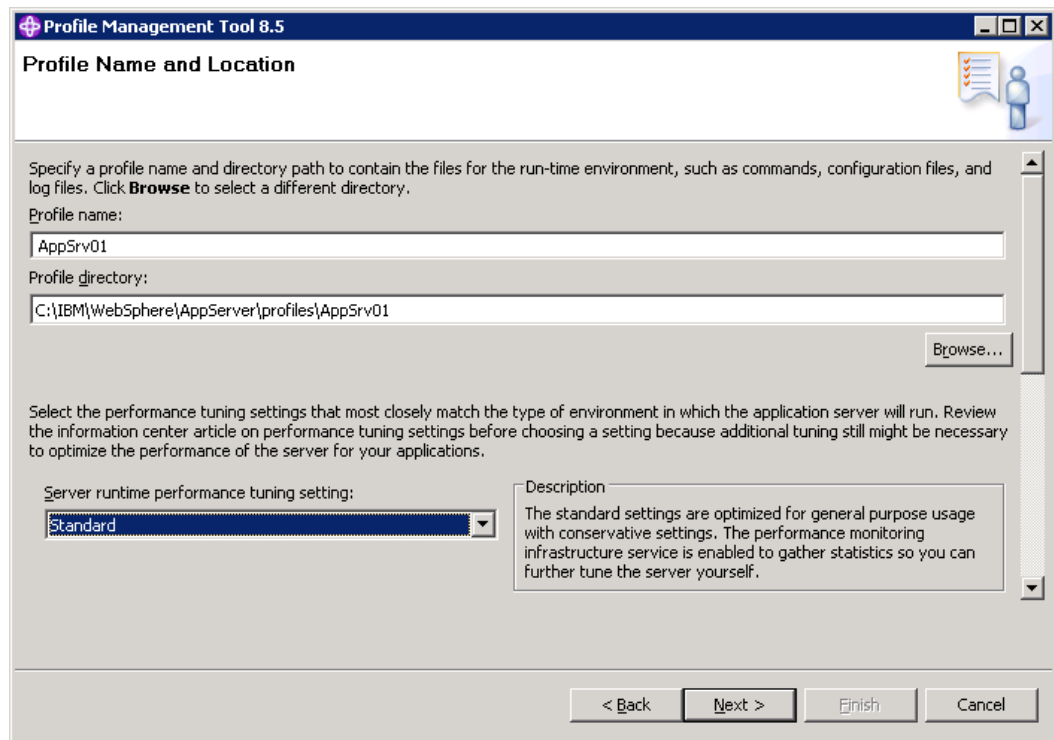


Figure 3-12 Specifying the profile name and its location

5. Enter the new node name and the system host name. The node name defaults to a name based on the host name of your system. The wizard checks if there are existing nodes in the installation and takes this situation into account when creating the default node name (Figure 3-13 on page 77). Click **Next**.

Note: If you plan to create multiple stand-alone application servers for federation later to the same cell, make sure that you select a unique node name for each application server.

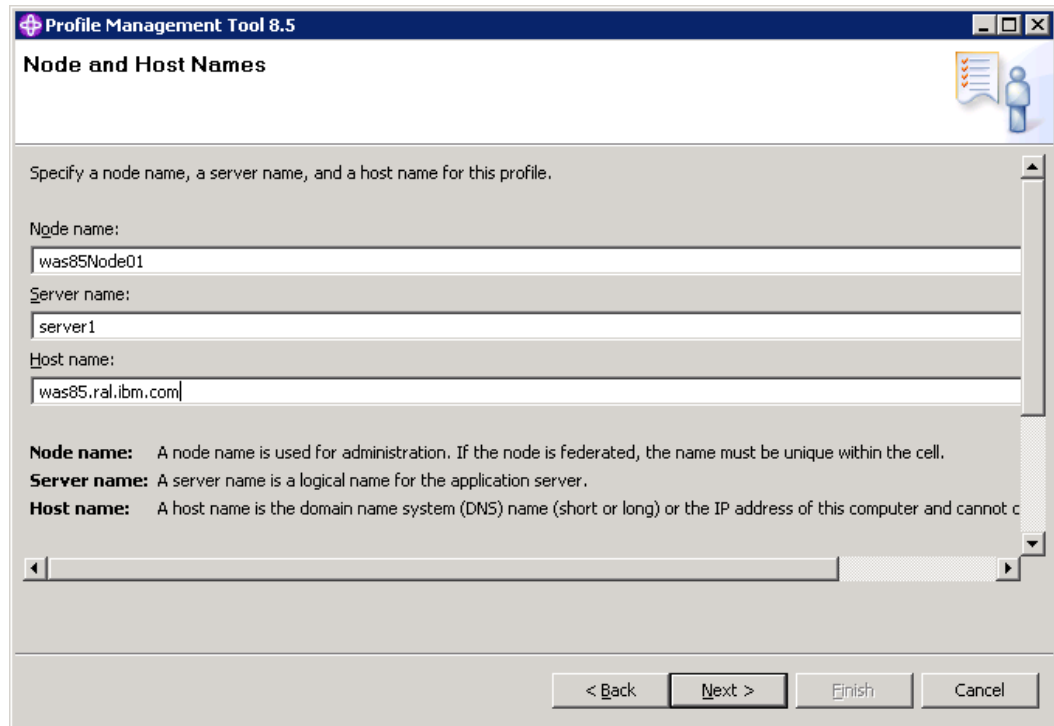


Figure 3-13 Specifying the node, host, and server names

6. Choose whether to enable administrative security. Refer to “Administrative security” on page 66 for more information. Click **Next**.
7. The next two screens guide you through certificate generation. Refer to “Certificates” on page 67 for more information. Click **Next**.
8. Configure TCP/IP ports for the server. Refer to “Port assignments” on page 69 for more information. Click **Next**.
9. If you install the server on Windows or Linux, configure the server to run as a service. Refer to “Running as a service” on page 71 for more information. Click **Next**.
10. The wizard allows you to create an optional web server definition, which defines an external web server. This setup allows you to manage web server plug-in configuration files for the web server and, in some cases, to manage the web server instance. If you have not installed a web server or want to perform this action later, you can easily do it from the administrative console. Working with web servers and WebSphere Application Server was covered in Chapter 12, “Configuring and managing web servers” on page 417. For this installation, disable the check box, and click **Next**.
11. Review the options you provided for the new profile, and click **Create** to create the profile.
12. Click **Finish** to close the wizard and start the First Steps console.
13. Use the First Steps console to verify the installation, start the server, and access the administrative console.
14. Log in to the console, if you disabled the security login without providing credentials.
15. Click **Servers** → **Server Types** → **WebSphere Application servers**. You can see the application server you just created (Figure 3-14 on page 78).

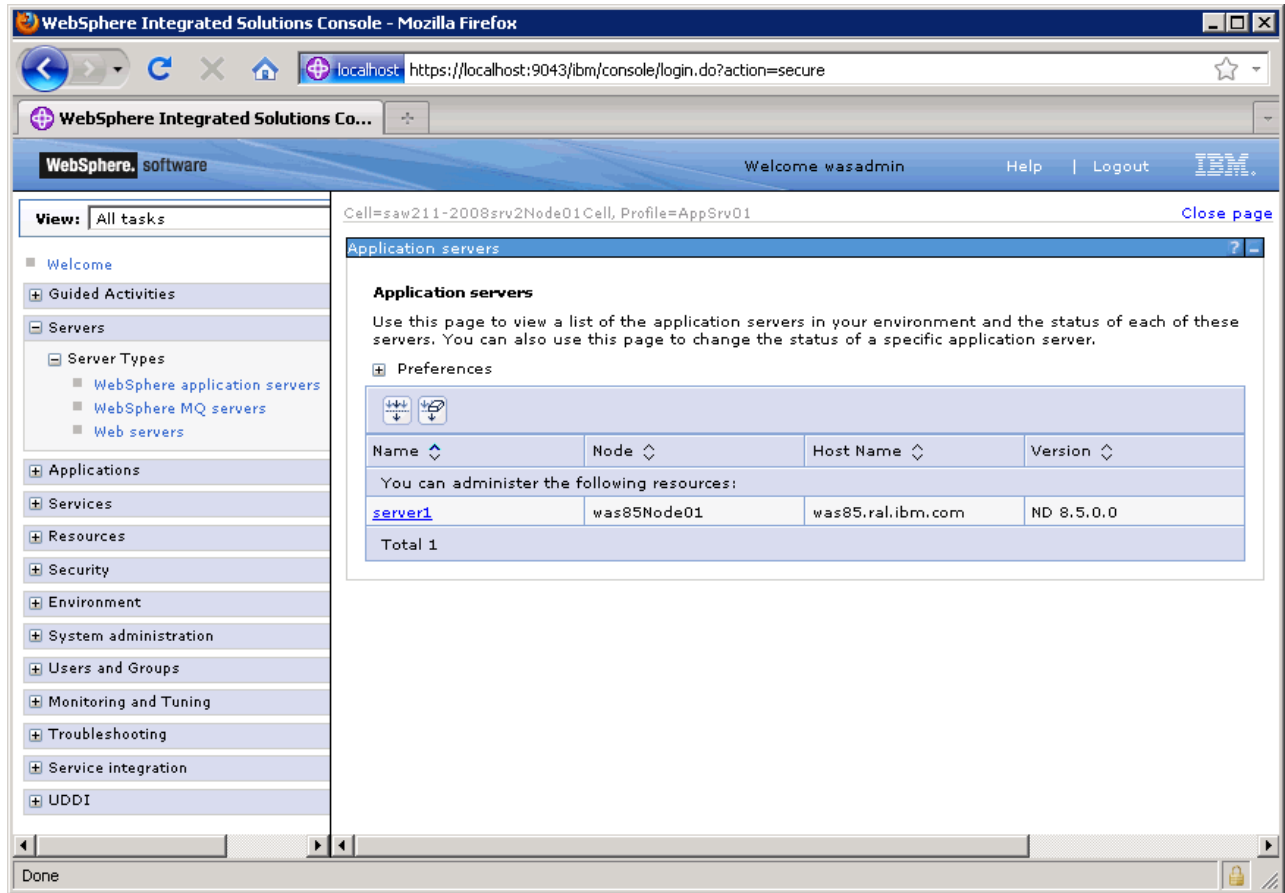


Figure 3-14 Application server profile viewed in the administrative console

For more information about working with the server, see Chapter 6, “Administration consoles and commands” on page 183.

3.3.4 Creating a deployment manager profile

To create the deployment manager profile:

1. Start the WebSphere Customization Toolbox, and open **Profile Management Tool**.
2. Click **Create**.
3. Select **Management**, and click **Next**.
4. Select **Deployment manager**, and click **Next**.
5. Select whether to take a typical or advanced path to install the profile:
 - If Typical is selected, then proceed to step 9 and continue from step 13
 - If Advanced is selected, continue with the following steps
6. Select the option to deploy the administrative console (the default), and click **Next**.

7. Enter a unique name for the profile or accept the default. The profile name becomes the directory name for the profile files. Click **Next**.

Note: If you enable the **Make this profile the default** check box, this profile receives console `wsadmin` commands by default. Otherwise you must specify the profile name for the command using the `profile` parameter.

8. Enter the node, host, and cell names. The defaults are based on the host name of your system, as illustrated on Figure 3-15. The wizard recognizes if there are existing cells and nodes in the installation and takes this setup into account when creating the default names. Click **Next**.

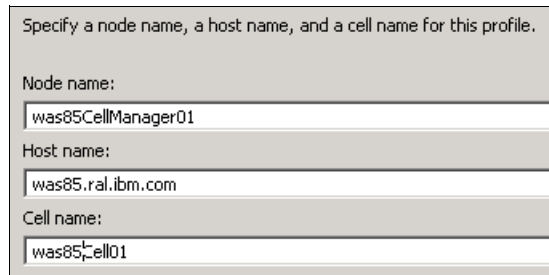


Figure 3-15 Creating a deployment manager profile - Enter cell, host, and node names

9. Choose whether to enable administrative security. Refer to “Administrative security” on page 66 for more information. Click **Next**.
10. The next two screens guide you through certificate generation. Refer to “Certificates” on page 67 for more information. Click **Next**.
11. Configure TCP/IP ports for the server. Refer to “Port assignments” on page 69 for more information. Click **Next**.
12. If you install the server on Windows or Linux, configure the server to run as a service. Refer to “Running as a service” on page 71 for more information. Click **Next**.
13. Review the options you provided for the new profile, and click **Create** to create the profile.
14. Click **Finish** to close the wizard and start the First Steps console.
15. Use the First Steps console to verify the installation, start the server, and access the administrative console.
16. Log in to the console, if you disabled the security login without providing credentials.
17. In the console, the following items are visible from the administrative console:
 - Deployment manager: Select **System administration** → **Deployment manager**.
 - Deployment manager node: Select **System administration** → **Nodes**.
 - The default node group: Select **System administration** → **Node groups**.
 - Cell information: Select **System administration** → **Cell** → **Local Topology**. You can see a similar topology, as illustrated on Figure 3-16 on page 80. Notice, that at completion of this process, you do not have any nodes or application servers in the cell.



Figure 3-16 Topology of the deployment manager profile installation

Working with deployment managers: For information about starting, stopping, and viewing deployment managers, see Chapter 6, “Administration consoles and commands” on page 183.

3.3.5 Creating a cell profile

Using this option, you create two distinct profiles: a deployment manager profile and an application server profile that is federated to the new cell. The Profile Management Tool windows give you basically the same options that you see if you create a deployment manager and then an application server.

Table 3-1 shows a summary of the available options during a cell profile creation using typical and advanced paths.

Table 3-1 Cell profile options

| Typical | Advanced |
|---|--|
| The administrative console and default application are deployed by default. | You have the option to deploy the administrative console (recommended), the default application, and the sample applications (if installed). |
| The profile name for the deployment manager is Dmgrxx by default, where xx is 01 for the first deployment manager profile and increments for each one created. The profile is stored in <i>install_root/profiles/Dmgrxx</i> . | You can specify the profile name and its location. |

| Typical | Advanced |
|--|---|
| The profile name for the federated application server and node is AppSrvxx by default, where xx is 01 for the first application server profile and increments for each one created. The profile is stored in <i>install_root/profiles/AppSrvxx</i> . | You can specify the profile name and its location. |
| Neither profile is made the default profile. | You can choose to make the deployment manager profile the default profile. |
| The cell name is <i><host>Cellxx</i> . The node name for the deployment manager is <i><host>CellManagerxx</i> . The node name for the application server is <i><host>Nodexx</i> . | You can specify the cell name, the host name, and the profile names for both profiles. |
| You can enable administrative security (yes or no). If you select yes, you are asked to specify a user name and password that is given administrative authority. | |
| TCP/IP ports default to a set of ports not used by any profiles in this WebSphere installation instance. | You can use the recommended ports for each profile (unique to the installation), Note that there are three different configurations for deployment manager, application server, and node agent. |
| If installing on Windows, the deployment manager runs as a service. | If installing on Windows or Linux, you can choose whether the deployment manager runs as a service. |
| Does not create a web server definition. | Allows you to define an external web server to the configuration. |

3.3.6 Creating a custom profile

A custom profile defines an empty node on a system. The purpose of this profile is to define a node to be federated to a cell for management through a deployment manager.

As you create the profile, you have the option to federate the node to a cell during the wizard or to simply create the profile for later federation. Before you can federate the custom profile to a cell, you must have a running deployment manager.

Note: With other profiles, you have the option of registering the processes as Windows or Linux services. This option is not available when you create a custom profile.

To create the custom profile, complete the following steps:

1. Start the WebSphere Customization Toolbox, and open **Profile Management Tool**.
2. Click **Create**.
3. Select **Custom profile**, and then click **Next**.
4. Select whether to take a typical or advanced path to install the profile:
 - If Typical is selected, only proceed with steps 7, 9 and 10.
 - If Advanced is selected, continue with every following steps.
5. Enter a unique name for the profile or accept the default. The profile name becomes the directory name for the profile files. If you enable the **Make this profile the default** check box, this profile receives console commands by default. Click **Next**.

6. Enter the node and host names. The defaults are based on the host name of your system. The wizard recognizes if there are existing cells and nodes in the installation and takes this setup into account when creating the default names. Click **Next**.
7. If you want to federate the new node defined by the profile to a cell as part of the wizard process, leave the **Federate this node later** check box disabled; however, we will federate the nodes later in “Federating a custom node to a cell” on page 84, so you can enable the check box, as illustrated on Figure 3-17, and click **Next**.

Note: If you choose to federate the node during the custom profile installation, enter the host name, SOAP port, user ID, and password of the administrator defined for the deployment manager profile you created in 3.3.4, “Creating a deployment manager profile” on page 78. The wizard uses this information to attempt a connection to the deployment manager. If you entered any of these values incorrectly, an error is displayed and you will have to correct the values.

To federate the node, the network deployment profile have to be up and running.

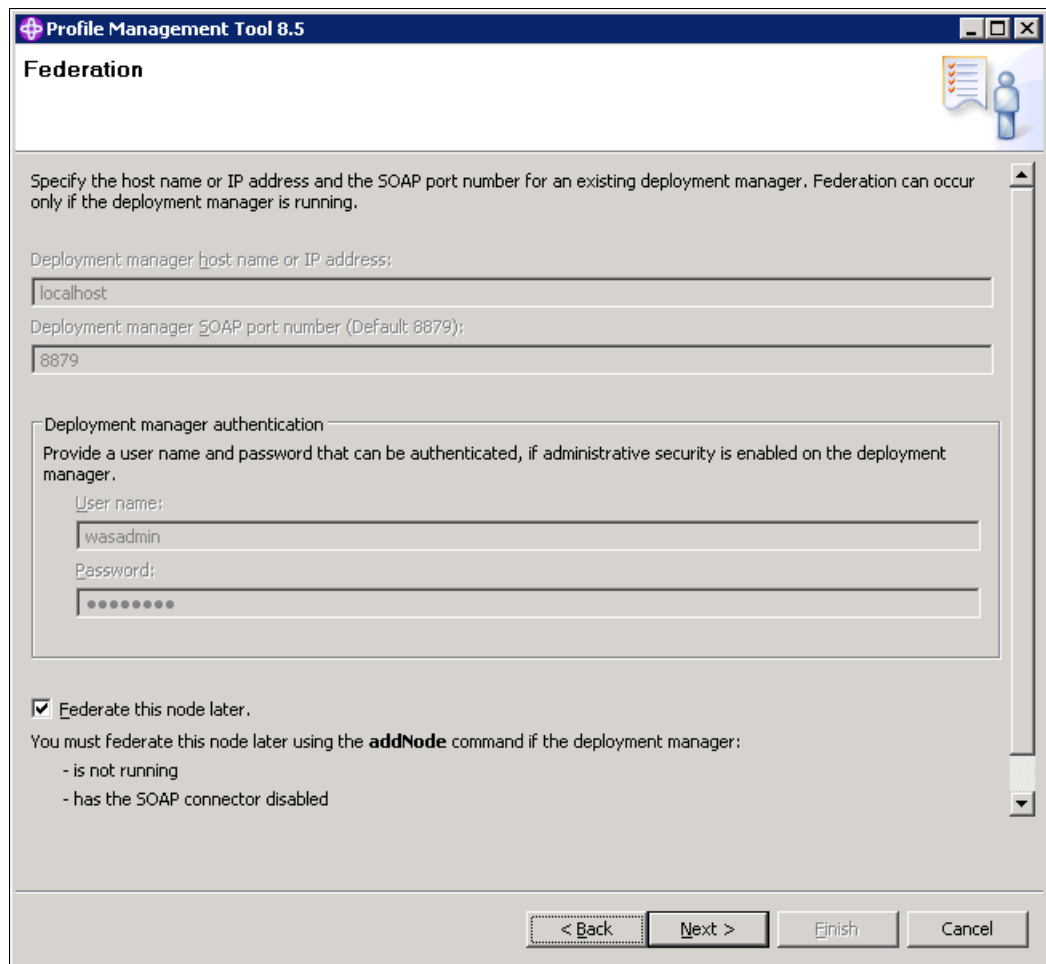


Figure 3-17 Federation option during custom profile installation

8. The next two screens guide you through certificate generation. Refer to “Certificates” on page 67 for more information. Click **Next**.
9. Review the options you provided for the new profile, and click **Create** to create the profile.

10. Disable the **Launch the First steps console** check box, and click **Finish**.

Note: Custom profiles do not create a server process, so you cannot verify, stop, or start this profile. The only reason to launch the First Steps menu is if you want to link to the information center or launch the migration wizard.

If you choose to federate the node to the deployment manager during the installation, you might want to check if it is available from the deployment manager console by accessing the cell local topology, as illustrated on Figure 3-16 on page 80, and continue by defining an application server on the new node, as described in 7.4.1, “Creating an application server” on page 248.

3.3.7 Federating nodes to a cell

A custom profile defines a node that can be added to a cell using the **addNode** command. A stand-alone application server can also be federated to a cell with the **addNode** command or from the deployment manager administrative console (the administrative console invokes the **addNode** command on the target system).

When you federate a node, the node name from the federated node is used as the new node name and must be unique in the cell. If the name of the node that you are federating already exists, the **addNode** operation fails.

Using the addnode command

The **addNode** command is run from the *install_root/bin* or *profile_root/bin* directory of the profile to be federated.

The most important addNode command parameters are:

► **dmgr_host, dmgr_port, username, password**

These parameters are used to obtain connection to the deployment manager.

► **startingport, portprops <filename>**

The new node agent is assigned a range of ports automatically. If you want to specify the ports for the node rather than taking the default, you can specify a starting port using the **startingport** parameter. The numbers are incremented from this number.

For example, if you specify 3333, the **BOOTSTRAP_ADDRESS** port will be 3333, **CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS** will be 3334, and so on.

As an alternative, you can provide specific ports by supplying a file with the port properties.

► **includeapps, includebuses**

If you are federating an application server, you can keep any applications or service integration buses that are deployed to the server. The default behavior is not to include any of these resources during federation, so they will be lost.

For more information about the **addNode** syntax and more options, see the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/xml_addnode.html

The **addNode** command performs the following actions:

1. Connects to the deployment manager process. This action is necessary for the file transfers performed to and from the deployment manager to add the node to the cell.
2. Attempts to stop all running application servers on the node.
3. Backs up the current stand-alone node configuration to the *profile_root/config/backup/base/* directory.
4. Copies the stand-alone node configuration to a new cell structure that matches the deployment manager structure at the cell level.
5. Creates a new local config directory and definition (*server.xml*) for the node agent.
6. Creates entries (directories and files) in the master repository for the new node's managed servers, node agent, and application servers.
7. Uses the FileTransfer service to copy files from the new node to the master repository.
8. Uploads application or service bus resources to the cell only if the **includeapps** or **includebuses** options are specified.
9. Performs the first file synchronization for the new node. This action synchronizes data from the cell to the new node.
10. Corrects the node's **setupCmdLine** and **wsadmin** scripts to reflect the new cell environment settings.
11. Launches the node agent (unless **noagent** is specified).

You can trace this procedure by viewing the federation logs provided on Example 3-1.

Federating a custom node to a cell

Note: You only have to do this action if you created a custom profile and chose *not* to federate it at the time.

To federate the custom node to the cell:

1. Ensure that the deployment manager is up and running. If it is stopped, start it.
2. Go to the *profile_root/bin* directory on the system where you created the custom profile for the new node.
3. Run the **addNode** command from Example 3-1, providing your information about the deployment manager. If administrative security is enabled, use the **username** and **password** arguments on the command line to provide the deployment manager user ID and password. If you do not provide the arguments, you are prompted for them.

Example 3-1 Federating node to a deployment manager cell using addNode command

```
C:\IBM\WebSphere\AppServer\profiles\Custom01\bin>addNode.bat was85.ra1.ibm.com 8879 -username
wasadmin -password passw0rd -includeapps
ADMU0116I: Tool information is being logged in file
C:\IBM\WebSphere\AppServer\profiles\Custom01\logs\addNode.log
ADMU0128I: Starting tool with the Custom01 profile
CWPKI0308I: Adding signer alias "CN=was85.ra1.ibm.com, OU=Root C" to local keystore
"ClientDefaultTrustStore" with the following SHA digest:
B8:3A:87:41:57:53:73:FB:B5:B3:8A:30:68:83:55:ED:06:12:BF:EB
CWPKI0309I: All signers from remote keystore already exist in local keystore.
ADMU0001I: Begin federation of node was85Node01 with Deployment Manager at
was85.ra1.ibm.com:8879.
```

```
ADMU0009I: Successfully connected to Deployment Manager Server:
was85.ra1.ibm.com:8879
ADMU0507I: No servers found in configuration under:
C:\IBM\WebSphere\AppServer\profiles\Custom01\config/cells/saw211-2008srv2Node02Cell/nodes/was85N
ode01/servers
ADMU2010I: Stopping all server processes for node was85Node01
ADMU0024I: Deleting the old backup directory.
ADMU0015I: Backing up the original cell repository.
ADMU0012I: Creating Node Agent configuration for node: was85Node01
ADMU0014I: Adding node was85Node01 configuration to cell: was85Cell01
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: was85Node01
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is:
1752
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent

ADMU0308I: The node was85Node01 and associated applications were successfully
added to the was85Cell01 cell.
ADMU0306I: Note:
ADMU0302I: Any cell-level documents from the standalone was85Cell01
configuration have not been migrated to the new cell.
ADMU0307I: You might want to:
ADMU0303I: Update the configuration on the was85Cell01 Deployment Manager with
values from the old cell-level documents.

ADMU0003I: Node was85Node01 has been successfully federated.

C:\IBM\WebSphere\AppServer\profiles\Custom01\bin>
```

4. Open the deployment manager administrative console, and view the new node and node agent details:
 - Select **System Administration** → **Nodes**. The new node is visible.
 - Select **System Administration** → **Node agents**. The new node agent and its status are visible
 - Select **System administration** → **Cell** → **Local Topology** to see the new node in the topology overview, as illustrated on Figure 3-18 on page 86.

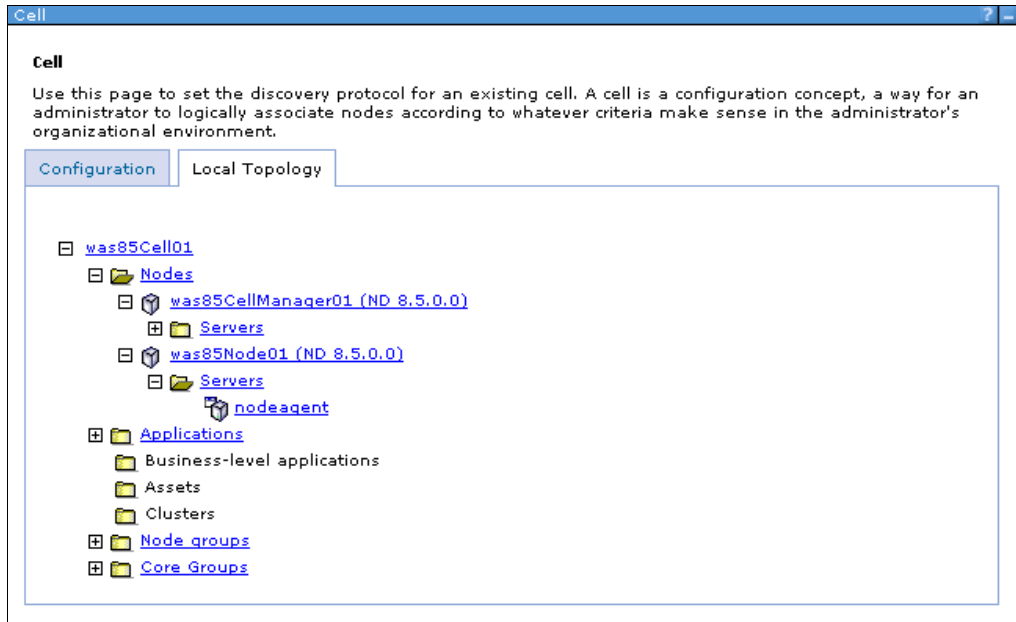


Figure 3-18 The new custom profile federated to the cell as a new node

The node is started as a result of the federation process. If it does not appear to be started in the console, you can check the status from a command window on the node system:

```
serverStatus.bat (sh) -all
```

If you find that it is not started, start it using the **startNode** command from its `profile_home\bin` directory:

```
startNode.bat (sh)
```

For more information about managing nodes, see 7.5, "Working with nodes in a Network Deployment environment" on page 282.

Note: If despite the successful **addNode** command usage, you do not see any new nodes in the deployment manager console, log out and log in to the console again. This forces the deployment manager GUI to pick the new changes to view them.

Be aware that the custom profile does not automatically give you an application server. You can complete the steps in 7.4.1, "Creating an application server" on page 248 to create a new server after the custom profile has federated to a cell.

Federating an application server profile to a cell

To federate an application server profile to a cell:

1. Ensure that both the target application server and the deployment manager are running.
2. Open the deployment manager administrative console, and log in with administrative privileges.
3. Click **System Administration** → **Nodes** → **Add Node**.
4. Select **Managed node**, and click **Next**.
5. Enter the information about your environment, as illustrated on Figure 3-19 on page 87, and click **OK**.

Note: You can choose the method to connect to the application server from the deployment manager. Consider using the default SOAP method if possible. Using the RMI method is also available but deprecated in WebSphere Application Server V8.5. Use the JSR160RMI connection type instead, if you want to stick to RMI.

Notice that if your application server profile is secured, you must provide credentials for both the application server and the deployment manager.

If you want to keep the applications you installed on the application profile, select the **Include applications** check box.

Use this page to identify a stand-alone application server process that is running. Start the application server, if necessary, or add the node from the command line by running the addNode command from the bin directory of the stopped application server profile.

Node connection

* Host
was85.ral.ibm.com

* JMX connector type
SOAP

* JMX connector port
8880

Application server user name
wasadmin

Application server password

* Deployment manager user name
wasadmin

* Deployment manager password

Config URL
file:\${USER_INSTALL_ROOT}/properties/sas.di

Options

Include applications

Include buses

Starting port

Use default

Specify

Port number

OK Cancel

Figure 3-19 Federating the application profile using the deployment manager console

6. If the node you are adding runs on a Windows machine, you can register the new node agent to run as a Windows service. Click **OK** to start the profile federation.

The federation process is similar to the process described in “Using the addnode command” on page 83. You can observe the state of this operation in the console window.

When the process completes:

- ▶ The profile directory for the application server still exists and is used for the new node.

- ▶ The old cell name for the application server is replaced in the profile directory with the cell name of the deployment manager:

profile_root/config/cells/dmgr_cell

- ▶ A new entry in the deployment manager profile directory is added for the new node:

dmgr_profile_root/config/cells/dmgr_cell/nodes/federated_node

- ▶ An entry for each node in the cell is added to the application server profile configuration. Each node entry contains the `serverindex.xml` file for the node:

profile_root/config/cells/dmgr_cell/nodes/federated_node

In turn, an entry for the new node is added to the nodes directory for each node in the cell with a `serverindex.xml` entry for the new node.

After successful federation, check the new cell member in the local cell topology in the deployment manager console, as illustrated on Figure 3-18 on page 86.

3.3.8 Creating a job manager profile

To create the job manager profile:

1. Start the WebSphere Customization Toolbox, and open **Profile Management Tool**.
2. Click **Create**.
3. Select **Management**, and click **Next**.
4. Select **Job manager**, and click **Next**.
5. Select whether to take a typical or advanced path to install the profile:
 - If Typical is selected, jump to step 9 and continue from step 13.
 - If Advanced is selected, continue with the following steps.
6. Select the option to deploy the administrative console (the default), and click **Next**.
7. Enter a unique name for the profile or accept the default. The profile name becomes the directory name for the profile files. Click **Next**.
8. Enter the node and host name. The defaults are based on the host name of your system. Click **Next**.
9. Choose whether to enable administrative security. Refer to “Administrative security” on page 66 for more information. Click **Next**.
10. The next two screens guide you through certificate generation. Refer to “Certificates” on page 67 for more information. Click **Next**.
11. Configure TCP/IP ports for the server. Refer to “Port assignments” on page 69 for more information. Click **Next**.
12. If you install the server on Windows or Linux, configure the server to run as a service. Refer to “Running as a service” on page 71 for more information. Click **Next**.
13. Review the options you provided for the new profile, and click **Create** to create the profile.
14. Click **Finish** to close the wizard and start the First Steps application.
15. Use the First Steps console to verify the installation, start the server, and access the administrative console.
16. Log in to the console, if you disabled the security login without providing credentials.
17. Click **System Administration** → **Job manager**. The main job manager administration panel is displayed, as illustrated on Figure 3-20 on page 89.

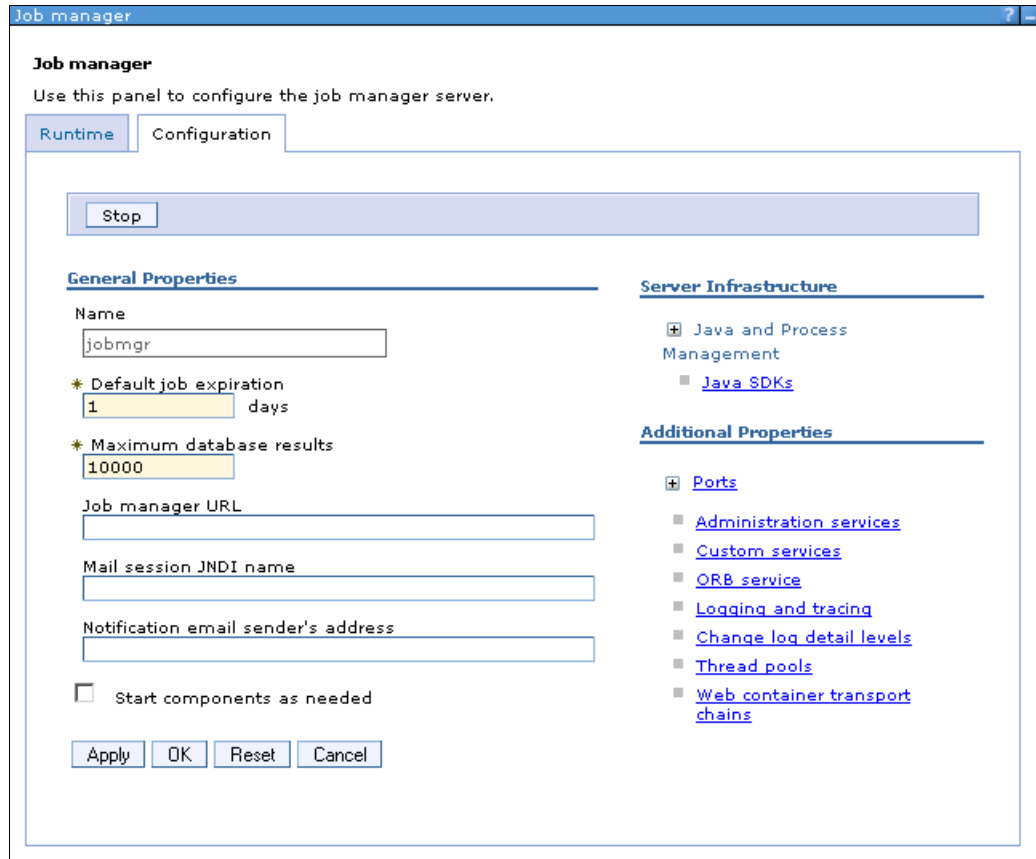


Figure 3-20 Job manager administration console

Job manager can send jobs to remote machines to remotely install or configure profiles. You can view the list of its targets by clicking **Jobs** → **Targets**. After initial creation of the job manager profile, the list is empty. To add new targets and work with jobs, refer to Chapter 29, “Managing an environment with the centralized installation manager” on page 1001.

To register an administrative agent node with job manager, see 3.3.12, “Registering administrative nodes with a job manager” on page 92.

3.3.9 Creating an administrative agent profile

To create the administrative agent profile:

1. Start the WebSphere Customization Toolbox and open **Profile Management Tool**.
2. Click the **Create** button.
3. Select **Management**, and click **Next**.
4. Select **Administrative agent**, and click **Next**.

The rest of the administrative agent profile installation is the same as the installation for the job manager profile. Follow the 3.3.8, “Creating a job manager profile” on page 88 from step 5 to 16 of the installation process.

5. Click **System Administration** → **Administrative agent**. The main administrative agent administration panel is displayed, as illustrated in Figure 3-21 on page 90.

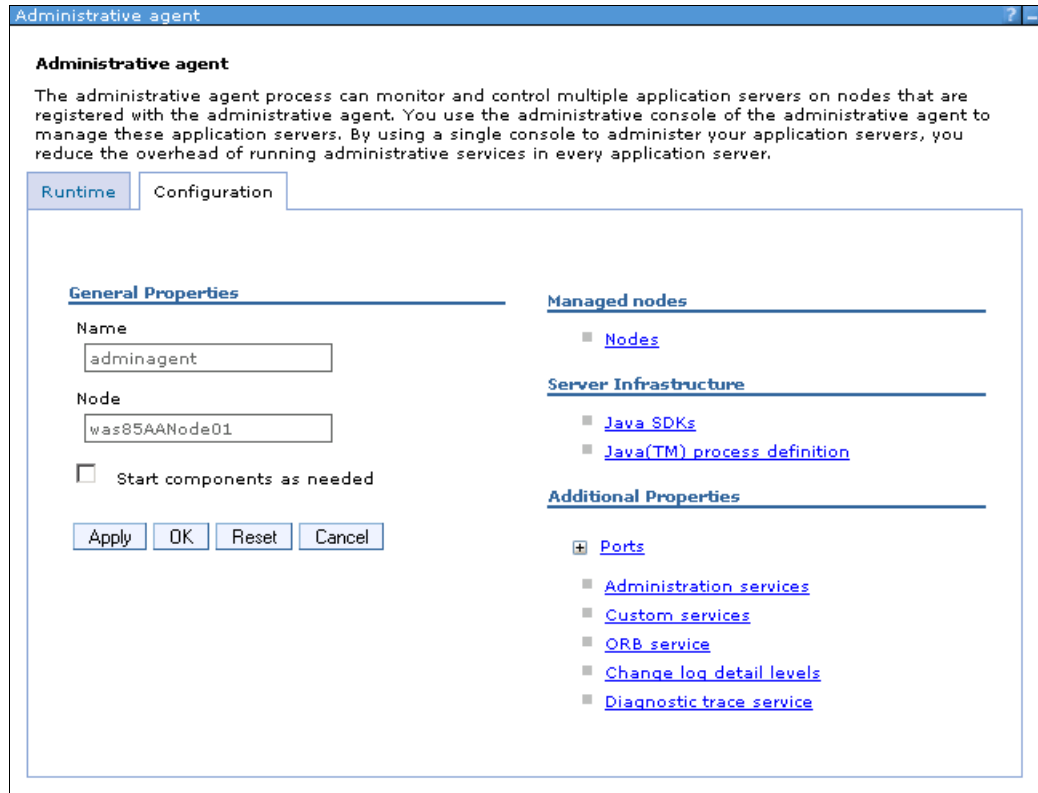


Figure 3-21 Administrative agent administration console

Nodes that are registered to the administrative agent can be viewed by clicking the **Nodes** link under Managed nodes. After initial creation of the administrative agent, the list will be empty. To register or deregister stand-alone application server nodes with the administrative agent, see 3.3.10, “Registering nodes to an administrative agent” on page 90 and 3.3.11, “Deregistering a node from the administrative agent” on page 92.

3.3.10 Registering nodes to an administrative agent

The administrative agent profile provides a single interface to manage unfederated application server nodes (stand-alone application server profiles).

Notes for use:

- ▶ The administrative agent and application servers must be on the same machine or sysplex.
- ▶ The administrative agent must be started before running the `registerNode` command.

You can only run the command on an unfederated stand-alone application server. When you run the command, the node for the stand-alone server is converted into a node that the administrative agent manages.

To register a node with an administrative agent, use the `registerNode` command. Example 3-2 on page 91 lists a sample command usage. In this case, the AppSrv02 profile is being registered to Admi nAgent01 administrative agent profile.

Example 3-2 Registering a stand-alone application server to an administrative agent

```
C:\IBM\WebSphere\AppServer\profiles\AdminAgent01\bin>registerNode.bat -profileName
AdminAgent01 -host was85.ra1.ibm.com -profilePath
"C:\IBM\WebSphere\AppServer\profiles\AppSrv02" -connType SOAP -port 8877 -username
aaadmin -password aapassw0rd -nodeusername wasadmin -nodepassword passw0rd
ADMU0116I: Tool information is being logged in file
          C:\IBM\WebSphere\AppServer\profiles\AdminAgent01\logs\registerNode.log
[...]
C:\IBM\WebSphere\AppServer\profiles\AppSrv02 has been successfully registered.
```

To learn more about the **registerNode** command, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/ragt_registerNode.html

After registering a new server to the administrative agent, log in to the administrative agent console to manage the server. Notice that now you can select which server to administrator, the administrative agent, or the new stand-alone profile, as illustrated on Figure 3-22.

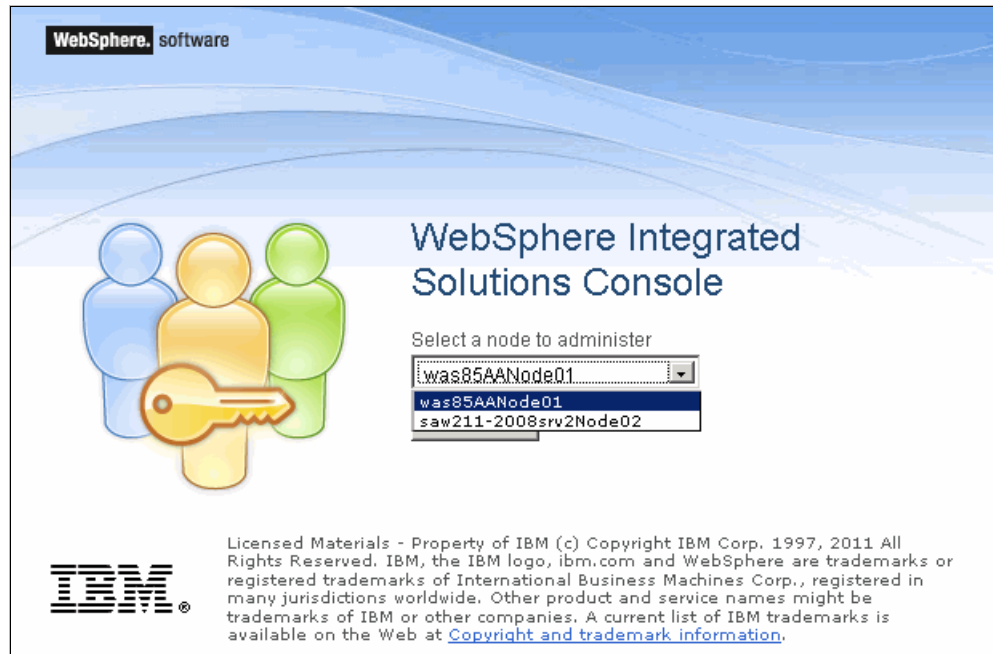


Figure 3-22 Choosing the profile to manage when logging to the administrative agent console

If you select the new server, provide its credentials to log in. Go to **Servers** → **Server Types** → **WebSphere application servers**. Figure 3-23 on page 92 illustrates the new management console view for the stand-alone server. Notice that from this console you can do additional operations to the profile, such as starting the server. There is no such option in standard stand-alone console.

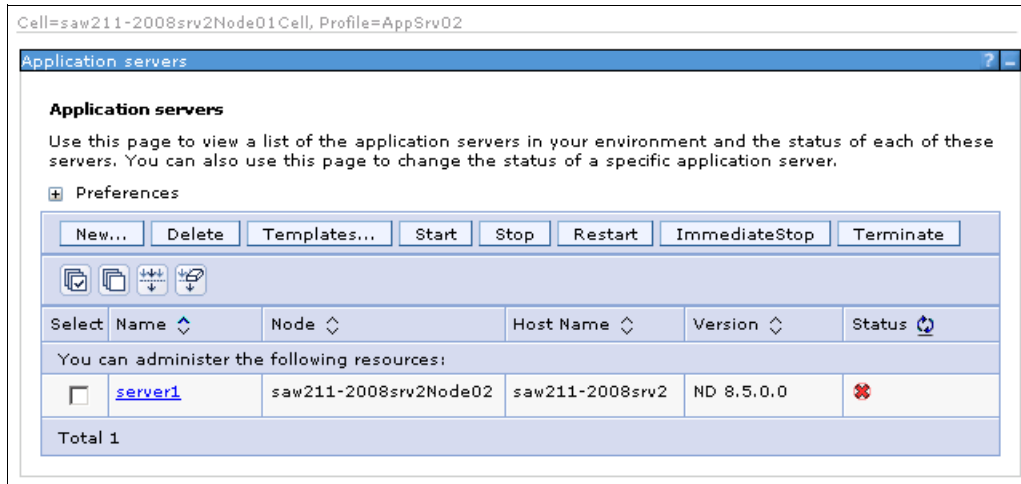


Figure 3-23 Administrative console of a registered stand-alone application server profile

Note: After registering the stand-alone application server it no longer provides the build-in management console. If you try to access the console using the server's administrative port, notice that the management console is no longer available.

It is important to understand that a stand-alone server can be federated to a deployment manager cell or registered to administrative agent. You cannot do both operations on the same stand-alone server.

3.3.11 Deregistering a node from the administrative agent

To deregister a node from the administrative agent, simply run the `deregisterNode` command from the `adminAgt_profile_root/bin` directory, as shown on Example 3-3.

Example 3-3 Deregistering a stand-alone application server from an administrative agent

```
C:\IBM\WebSphere\AppServer\profiles\AdminAgent01\bin>deregisterNode.bat -connType SOAP -port 8877 -profilePath "C:\IBM\WebSphere\AppServer\profiles\AppSrv02" -username wasadmin -password passw0rd
```

To learn more about the `deregisterNode` command, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipartform.doc/ae/ragt_deregisterNode.html

3.3.12 Registering administrative nodes with a job manager

The Job manager profile provides a single administrative interface for managing other WebSphere profiles. In this section, we register both the administrative agent and deployment manager profiles to a job manager.

Registering administrative agents

To register administrative agents with a job manager:

1. Log on to the administrative agent node.

2. Click **System administration** → **Administrative agent** → **Nodes**.
3. Select the node that you want to register with the job manager, as illustrated in Figure 3-24, and then click **Register with Job Manager**.

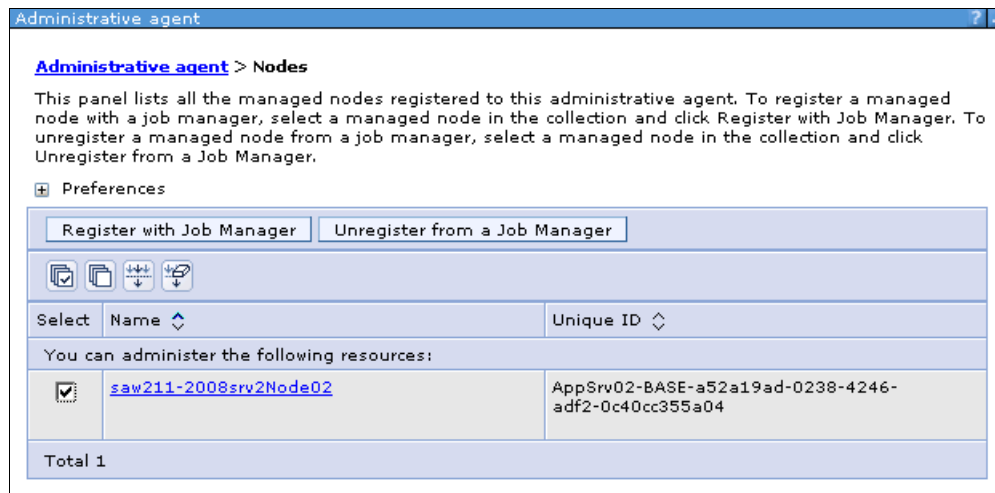


Figure 3-24 Select which node will be registered with the job manager

4. Enter the information required to connect to the job manager, including the host name, port, user ID, and password, as illustrated in Figure 3-25. Click **OK**.

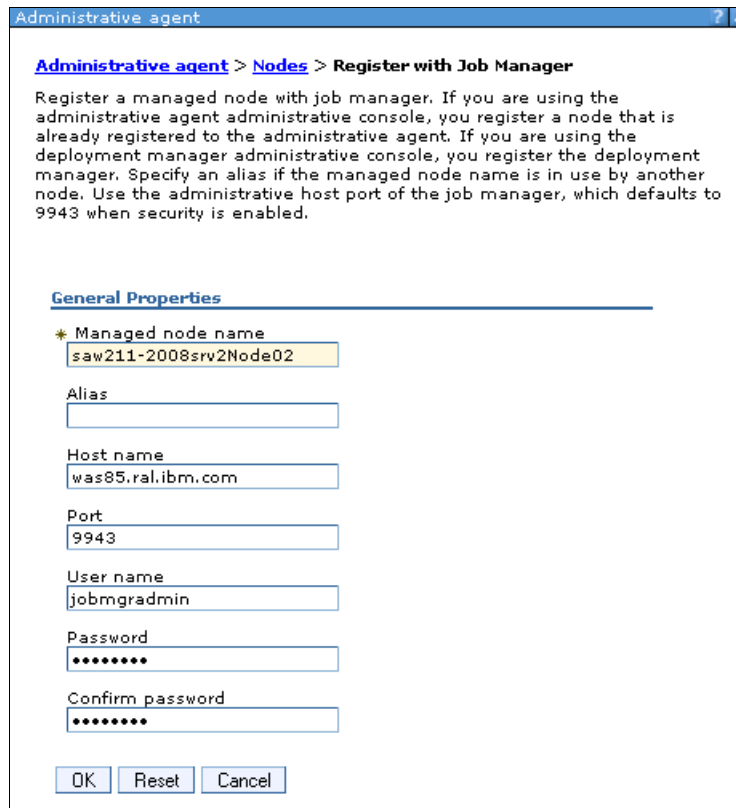


Figure 3-25 Registering a node with a job manager

Note: If the node name you are registering is already in use by the job manager, you can enter an alias for the node.

To view the newly registered node, log in to the job manager console, and click **Jobs** → **Targets**. This action lists the nodes and deployment managers that are registered with the job manager, as illustrated in Figure 3-26.

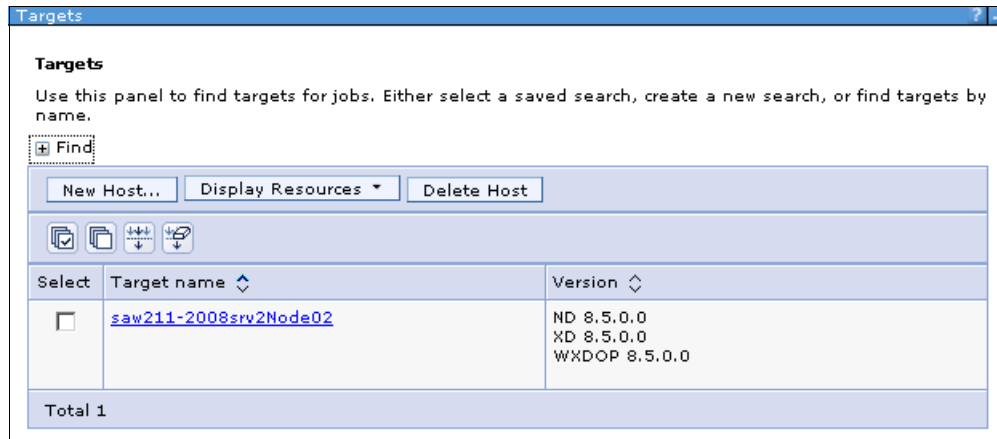


Figure 3-26 Listing targets in the job manager console

Refer to the following information center website for more details about registering administrative agents with job manager:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/tagt_adminagent_setup.html

Registering deployment managers

To register a deployment manager node with a job manager:

1. Log in to the deployment manager administrative console, and click **System administration** → **Deployment manager**.
2. Under Additional Properties, click **Job managers**.
3. Click **Register with Job Manager**.
4. Enter the information required to connect to the job manager, including the host name, port, user ID, and password. Click **OK** (see Figure 3-25 on page 93).

Note: If the node name you are registering is already in use by the job manager, you can enter an alias for the node.

To view the newly registered deployment manager, log in to the job manager console, and click **Jobs** → **Targets**. This action lists the nodes and deployment managers that are registered with the job manager.

3.4 Managing profiles with the command line

You already saw how profiles are created with the Profile Management Tool and administrative consoles. At the heart of these tool lays the **manageprofiles** command, which can also be used to manage profiles directly. Using the **manageprofiles** command, you can create, list, augment, or delete the profiles.

The **manageprofiles** command is in the *install_root/bin* directory. To get more information about using this command, type:

```
manageprofiles -help
```

To explore all of the functions of this command, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_manageprofiles.html

3.4.1 Listing profiles

To list all created profiles, run the command shown in Example 3-4.

Example 3-4 Listing profiles

```
C:\IBM\WebSphere\AppServer\bin>manageprofiles.bat -listProfiles  
[Dmgr01, AppSrv01, Custom01, AdminAgent01, JobMgr01, AppSrv02]
```

3.4.2 Creating profiles from templates

When you create a profile using the **manageprofiles** command, you must specify a *profile template*, which is supplied with the product. WebSphere Application Server uses these templates as a base for creating a new profile. These templates are located in the *install_root/profile templates* directory. Each template consists of a set of files that provide the initial settings for the profile and a list of actions to perform after the profile is created.

The following profiles are defined by default for the WebSphere Application Server V8.5:

- ▶ Default (for application server profiles)
- ▶ Management (for deployment manager, job manager, and administrative agent profiles)
- ▶ Managed (for custom profiles)
- ▶ Cell (for cell profiles)

To create a deployment manager named Dmgr02 with enabled administrative security enabled, see the command used in Example 3-5.

Example 3-5 Creating deployment manager profile with the manageprofiles command

```
manageprofiles.bat -create -templatePath  
c:\IBM\WebSphere\AppServer\profileTemplates\management -serverType  
DEPLOYMENT_MANAGER -profileName Dmgr02 -profilePath  
c:\IBM\WebSphere\AppServer\profiles\Dmgr02 -enableAdminSecurity true  
-adminUserName wasadmin -adminPassword passwOrd -cellName myHostCell01 -nodeName  
myHostCellManager01
```

The log files that are created when you run the `manageprofiles` command are located in:

`install_root/logs/manageprofile/profilename_action.log`

For example:

`C:\IBM\WebSphere\AppServer\logs\manageprofiles\Dmgr02_create.log`

Additional log files are created in the following directory:

`install_root/logs/manageprofile/profile_name/`

For example:

`C:\IBM\WebSphere\AppServer\logs\manageprofiles\Dmgr02`

Important: Do not manually modify the files that are located in the `install_root/profileTemplates` directory.

3.4.3 Creating profiles with non-default ports

During profile creation using the `manageprofiles` command, you can accept the default port values, or you can specify your own port settings. If you want to specify ports, you can do so in any of the following ways:

- ▶ Specify the ports by pointing to a file that contains the port values
- ▶ Specify the starting port value
- ▶ Specify the default port values

During profile creation, the `manageprofiles` command uses an automatically generated set of recommended ports. You can modify the port values using the following parameters on the `manageprofiles` command:

Note: When you create a managed profile you should not use any of these parameters.

- ▶ `defaultPorts`: Assigns default or base port value for the profile.
- ▶ `startingPort`: Specifies the starting port number for generating and assigning all ports for the profile. If a port value in the sequence conflicts with an existing port assignment, the next available port value is used.
- ▶ `portsFile`: Specifies a path to a file that defines port settings for the profile. Example 3-6 shows sample content of such a file. You can use the `portdef.props` file as a template.

Example 3-6 Example contents of portdef.props file

```
IPC_CONNECTOR_ADDRESS=9636
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS=9416
XDAGENT_PORT=7063
OVERLAY_UDP_LISTENER_ADDRESS=11011
WC_adminhost=9064
DataPowerMgr_inbound_secure=5556
DCS_UNICAST_ADDRESS=9357
BOOTSTRAP_ADDRESS=9812
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS=9417
SOAP_CONNECTOR_ADDRESS=8884
CELL_DISCOVERY_ADDRESS=7279
ORB_LISTENER_ADDRESS=9103
STATUS_LISTENER_ADDRESS=9421
```



```
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS=9418
OVERLAY_TCP_LISTENER_ADDRESS=11012
WC_adminhost_secure=9047
```

You can also use the `validatePorts` parameter, which specifies that ports must be validated to ensure that they are not reserved or in use. This parameter helps identify ports that are not being used.

Example 3-7 shows how to create an AppSrv05 stand-alone application server profile using the `startingPort` parameter that generates ports greater than 22222.

Example 3-7 Creating a stand-alone profile using the `startingPort` parameter

```
C:\IBM\WebSphere\AppServer\bin>manageprofiles.bat -create -templatePath
c:\IBM\WebSphere\AppServer\profileTemplates\default -profileName AppSrv05
-profilePath c:\IBM\WebSphere\AppServer\profiles\AppSrv05 -startingPort 22222
-cellName test5Cell01 -nodeName test5Node01
```

Note: To change the ports after the profile creation, use the `updatePorts` tool. For more information, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/tins_updatePorts.html

For more examples of creating profiles with non-default ports, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rxml_manageprofiles.html

3.4.4 Deleting profiles

To properly delete a profile:

- ▶ If you are removing an application server profile that has not been federated to a cell:
 - a. Stop the application server.
 - b. Delete the profile using the following command:

```
manageprofiles -delete -profileName profile_name
```
 - c. Clean the profile registry using the following command:

```
manageprofiles -validateAndUpdateRegistry
```
 - d. Delete the `profile_root` directory.
- ▶ If you are removing a custom profile or application server profile that is federated to a cell:
 - a. Stop the profile server instance on this node.
 - b. Remove the node from the cell using the administrative console or the `removeNode` command. This will not delete the node but only restore it to its pre-federated configuration.
 - c. Delete the profile using the following command:

```
manageprofiles -delete -profileName profile_name.
```
 - d. Clean the profile registry using the following command:

```
manageprofiles -validateAndUpdateRegistry
```

- e. Delete the *profile_root* directory.
- ▶ If you are removing a deployment manager profile:
 - a. Remove any nodes federated to the cell using the administrative console or the **removeNode** command. This will not delete the node but only restore it to its pre-federated configuration.
 - b. Stop the deployment manager.
 - c. Delete the deployment manager profile using the following command:


```
manageprofiles -delete -profileName profile_name
```
 - d. Clean the profile registry using the following command:


```
manageprofiles -validateAndUpdateRegistry
```
 - e. Delete the *profile_root* directory.

In case of problems or errors while deleting the profiles, check the logs under:

install_root/logs/manageprofile/profilename_delete.log

Example 3-8 shows the deletion of the AppSrv05 stand-alone profile and cleaning of the profile registry.

Example 3-8 Deleting a profile using manageprofiles

```
C:\IBM\WebSphere\AppServer\bin>manageprofiles.bat -delete -profileName AppSrv05
INSTCONFSUCCESS: Success: The profile no longer exists.
```

```
C:\IBM\WebSphere\AppServer\bin>manageprofiles -validateAndUpdateRegistry
[]
```

For more uses of the **manageprofile** command, see Chapter 30, “System recovery” on page 1055, where the following topics are discussed:

- ▶ Backing up a profile
- ▶ Restoring a profile
- ▶ Exporting and importing profiles

3.4.5 Using the manageprofiles interactive utility

The **manageprofile** command takes many parameters and for complex WebSphere environments it can be difficult to use. There is an interactive tool called Manage Profiles Interactive that guides you through the important **manageprofile** use cases.

This command is not shipped with the WebSphere package, but it is available to download at no cost. You will find the tool, documentation, and a sample video on its usage at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg21442487>

After you download the tool, you must unpack its content in the *install_root/bin* directory. After you unpack it, you can use two scripts to run it:

- ▶ **run_manageprofilesInteractive.bat** for Windows operating systems
- ▶ **run_manageprofilesInteractive.sh** for UNIX operating systems

Example 3-9 illustrates the tool usage of listing the profiles.

Example 3-9 Listing profiles using the interactive manageprofiles tool

```
C:\IBM\WebSphere\AppServer\bin>run_manageprofilesInteractive.bat
C:\IBM\WebSphere\AppServer\bin>CALL
"C:\IBM\WebSphere\AppServer\bin\setupCmdLine.bat"
manageprofilesInteractive-v70 V0.6.6 ~ 2011.05.10/Windows Server 2008 R2

-----
MANAGEPROFILES - Command Menu
-----
1  create
2  augment
3  delete
4  unaugment
5  unaugmentAll
6  deleteAll
7  listProfiles
8  listAugments
9  backupProfile
10 restoreProfile
11 getName
12 getPath
13 validateRegistry
14 validateAndUpdateRegistry
15 getDefaultName
16 setDefaultName
17 response
18 help
Select number [press "q" to quit]: 7
listProfiles

-----
LISTPROFILES command summary:
-----
Press "b" to go back and make changes or "c" to continue: c


Press "q" to quit, "r" add to response file, or "c" to run the command: c
-----
manageprofiles.bat -listProfiles
Added command to C:/IBM/WebSphere/AppServer/logs/manageprofilesInteractive.log
You may check C:/IBM/WebSphere/AppServer/logs/manageprofiles/listProfiles.log for
command status.

[Dmgr01, AppSrv01, Custom01, AdminAgent01, JobMgr01, AppSrv02, Dmgr02]

Elapse time: 4.954 seconds
Done!
```

After the tool is launched, it prints the available operations. To select the operation, provide the operation number and follow any instructions to execute the command. Notice that this tool can also generate response files for the **manageprofile** command. The tool also records all invoked commands in a special `manageprofilesInteractive.log` file. In Example 3-9, the following command was logged:

```
[6/25/12 6:55 PM] manageprofiles.bat -listProfiles
```

Installing WebSphere Application Server on z/OS systems

In this chapter, we provide an overview of IBM Installation Manager and explain how to install WebSphere Application Server on z/OS systems.

This chapter includes the following topics:

- ▶ IBM Installation Manager overview
- ▶ Installing Installation Manager
- ▶ Working with Installation Manager
- ▶ Using Installation Manager
- ▶ Installing WebSphere Application Server
- ▶ WebSphere Customization Toolbox
- ▶ Troubleshooting

4.1 IBM Installation Manager overview

IBM Installation Manager is a general-purpose software installation and update tool that can be used to install and maintain software packages. It provides the following features:

- ▶ Consistency across all platforms
- ▶ Lifecycle management for the products you install
- ▶ Ability to run as a command-line UNIX System Services application with the same look and feel
- ▶ Common packaging
- ▶ Greater efficiency for delivering new fixes (no need for a ++APAR to install an interim fix for WebSphere Application Server on z/OS)
- ▶ Invocable through a command-line interface, console mode, or response files.
- ▶ Modifiable through the addition or removal of optional features or language packs.

Figure 4-1 provides a high-level view of Installation Manager and the products it is used to install.

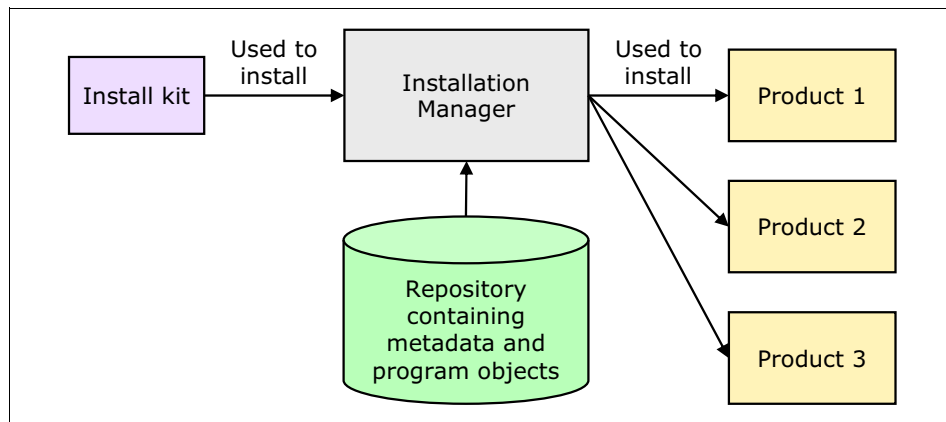


Figure 4-1 Installation Manager overview

We use the following concepts and terminology in this chapter:

- ▶ *Package*: is a software product that can be installed by Installation Manager. It is a separately installable unit that can operate independently from other packages of that software. A package can include a product, a group of components, or a single component. Each package has a name, version, and an identifier, for example, in case of WebSphere Application Server V8.5 for ZOS:

- Package name: `com.ibm.websphere.ZOS.V85`
- Package version: `8.5.0.20120501_1118`
- Package identifier: `com.ibm.websphere.zOS.v85_8.5.0.20120501_1118`

The packages are installed at a defined location in a UNIX System Services file system. Installation Manager allows you to control where products are installed and at which level.

- ▶ *Package group*: Packages installed to the same location that share UI elements. When more than one product is installed at the same location, the package group names are set automatically by Installation Manager.
- ▶ *Repository*: A place where the packages to be installed can be found. It has a list of files organized in a tree structure and includes metadata that describes the software version

and how it must be installed. A repository can reside on a local directory or on a remote, reachable server.

Installation Manager installs, uninstalls, modifies, and maintains the software using a software repository. In doing this, it uses three software locations:

- *Binary location*: The directory where Installation Manager is installed.
- *Agent data location* (also known as *appDataLocation*): The directory where Installation Manager stores data associated with an application, including the application state and operations history.
- *Object Cache location*: Used by Installation Manager to reduce the time when an operation is performed, which avoids spending time going online to use objects.

Figure 4-2 shows the components of Installation Manager.

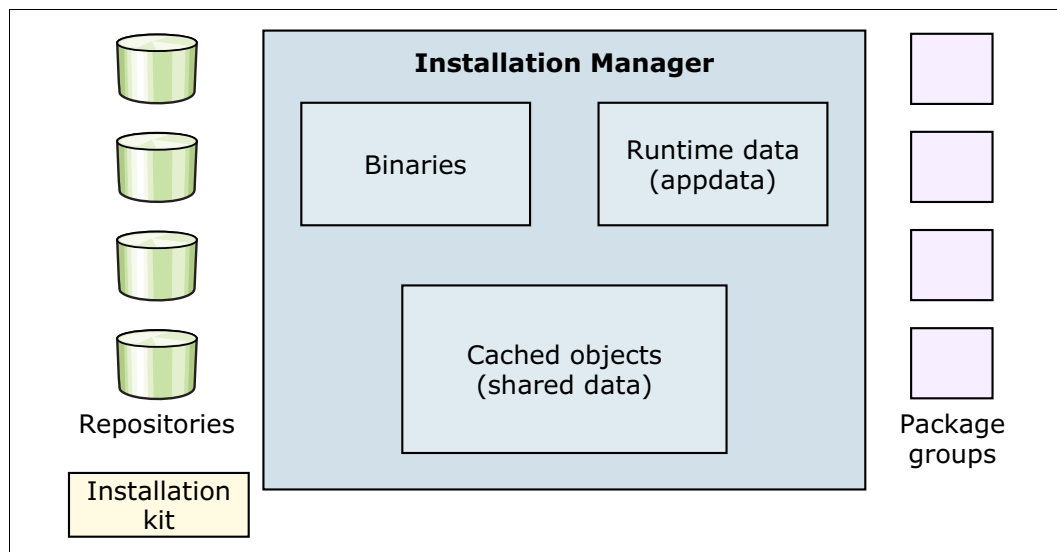


Figure 4-2 Installation Manager components

4.2 Installing Installation Manager

You must run Installation Manager only on systems on which you install or update product code. Typically, you need only one Installation Manager on a system because one Installation Manager can track any number of product installations.

You install Installation Manager through SMP/E, and then use the z/OS Profile Management Tool (zPMT) to configure your environment, as shown in Figure 4-3 on page 104.

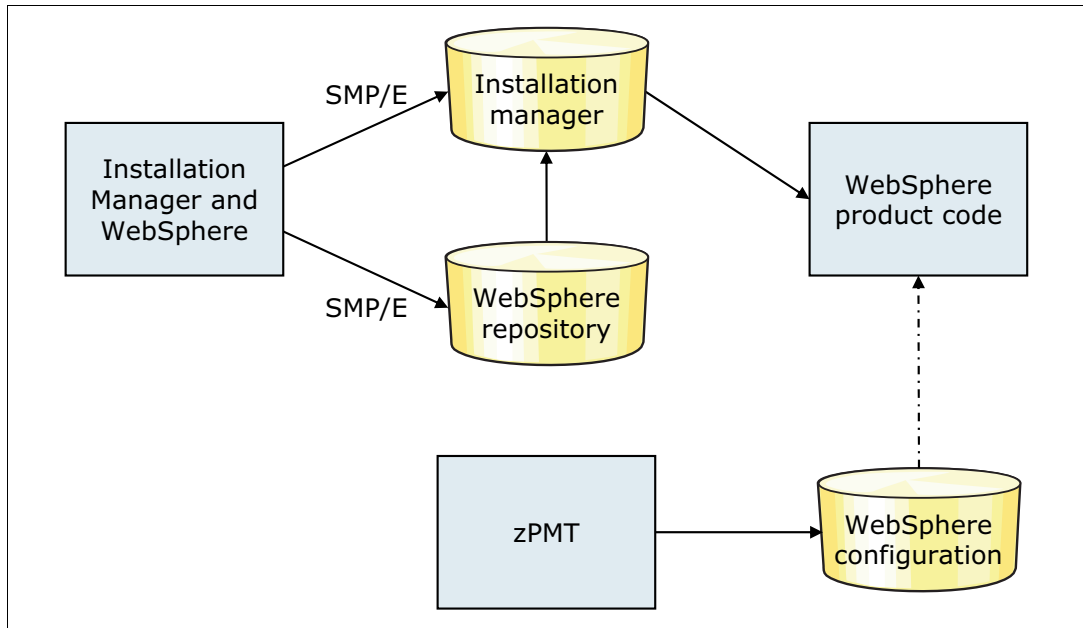


Figure 4-3 Installation process on z/OS

For further information about Installation Manager, visit the product information center at this website:

<http://pic.dhe.ibm.com/infocenter/install/v1r5/index.jsp>

4.2.1 Checking prerequisites

Before you install WebSphere Application Server for z/OS Version 8.5, be sure to check the hardware and software prerequisites at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg27024798>

4.2.2 Obtaining an Installation Manager installation kit

The installation kit for Installation Manager has function modification identifier (FMID) HGIN140, provided within WebSphere Application Server for z/OS Version 8.5. You can also obtain the stand-alone installation kit in one of the following ways:

- ▶ Through a ServerPac or IBM SystemPac® in a ready-to-use format.

Under this method, PTFs must be installed to bring the kit up to Installation Manager Version 1.5.2, which is the minimum that is required to install WebSphere Application Server for z/OS V8.5.

- ▶ Through a Custom-Built Product Delivery Offering (CBPDO).

In this scenario, you must use the following jobs to install CBPDO:

- GINRECEV to receive the deliverables
- GINALLOC to allocate target and distribution data sets
- GINDDEF to create SMP/E DDDEFs
- GINISMKD to mount the installation file system and create directories
- GINAPPLY to apply the installation kit FMID
- GINACCEPT to accept the installation kit FMID

- ▶ By downloading the kit from the Internet, transferring the compressed file to the z/OS system, and extracting it. You can order Installation Manager through Shopz or find the most current version at the following website:

http://pic.dhe.ibm.com/infocenter/install/v1r5/index.jsp?topic=%2Fcom.ibm.cic.aigent.ui.doc%2Ftopics%2Fr_links.html

Important: If you download Installation Manager from the Internet, SMP/E cannot maintain and track its level.

4.2.3 Installing Installation Manager on the system

Prepare for installation by determining the mode in which Installation Manager will run:

- | | |
|-------------------|---|
| Admin mode | Installation Manager can be invoked by any superuser. There can be only one admin mode Installation Manager on a system. |
| User mode | Installation Manager can be invoked by a non-superuser. There can be only one user mode Installation Manager per user. |
| Group mode | Installation Manager can be invoked by any user who is connected to the group that owns Installation Manager. There is no limit for the number of group mode installations. |

Installation Manager includes the following sets of files:

- ▶ Binaries: A set of executable files.
- ▶ Runtime: A set of files that describe the installed products.
- ▶ Shared data: A set of files that store artifacts from a repository when installing packages. To install WebSphere Application Server, the shared data must have at least 30,000 tracks of free space.

The data stored in the shared data directory is used only for a package rollback if a product repository is not available. The amount of shared data can become large if you have several products installed. If you have access to the product repositories, you can delete the shared data contents after using Installation Manager. As an alternative, you can ask Installation Manager to discard the cache objects by adding the preference value shown in Example 4-1 to the `imcl install` command.

Example 4-1 The preference parameter to discard cached objects

```
-preferences com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts=false
```

This value is used in the sample jobs in library SBBOJCL.

Note: The shared resources directory is set at the time a package is first installed. If all cached objects are removed and the directory becomes empty, Installation Manager can unset the shared resources location.

Important: All sets of files must be writable by an Installation Manager user or group. The permission settings for the admin and user modes are 755, and for group mode, 775.

The user who will run Installation Manager must have the following permissions on IBM Resource Access Control Facility (RACF®):

- ▶ Read access to FACILITY profile BPX.FILEATTR.APF
- ▶ Read access to FACILITY profile BPX.FILEATTR.PROGCTL
- ▶ Read access to FACILITY profile BPX.FILEATTR.SHARELIB
- ▶ Read access to UNIXPRIV profile SUPERUSER.FILESYS.CHOWN
- ▶ Read access to UNIXPRIV profile SUPERUSER.FILESYS.CHANGEPERMS

Table 4-1 lists the file systems that are needed to hold Installation Manager information.

Table 4-1 Installation Manager default locations

| Information type | Default location |
|------------------|------------------------------|
| Binaries | /InstallationManager/bin |
| Application data | /InstallationManager/appdata |

To install Installation Manager:

1. Go to the directory where Installation Manager is installed, and run the following command:


```
./set-ext-attr.sh
```
2. Choose a user ID to be the owner of this Installation Manager. If you need to create a user ID, use the GIN2ADMN job.
3. Create and mount two file systems to configure Installation Manager. You can use the GIN2CFS sample job.
4. Issue one of the following commands to begin installing Installation Manager in your preferred mode:

| | |
|-------------------|--|
| installc | Installs Installation Manager in administrator mode. |
| userinstc | Installs Installation Manager in user mode. |
| groupinstc | Installs Installation Manager in group mode. |

Important: The user who runs the command is the Installation Manager owner.

Example 4-2 demonstrates how to install Installation Manager in admin mode using the following parameters:

| | |
|-------------------------------|---|
| -acceptLicense | Accepts the software license agreement |
| -installationDirectory | Specifies the directory where Installation Manager binaries are installed |
| -dataLocation | Specifies the directory where runtime data is installed |

Example 4-2 Installing Installation Manager in admin mode

```
/usr/lpp/InstallationManager/V1R5/installc -acceptLicense
-installationDirectory /InstallationManager/bin -dataLocation
/InstallationManager/appdata
```

As an alternative to these step-by-step instructions, you also can install Installation Manager using the GIN2INST sample job.

Verification: To verify that Installation Manager was installed, run the following command:
`/InstallationManager/bin/eclipse/tools/imcl -version`

Additional information about installing Installation Manager can be found at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg24031300>

4.3 Working with Installation Manager

When you work with Installation Manager, you can influence how it operates by setting certain preferences. You also need one or more repositories. In this section, we explain how to work with Installation Manager.

4.3.1 Installation Manager preferences

You can set the Installation Manager preferences using one of the following methods:

- ▶ When using command line, specify the **-preferences** parameter followed by the parameter name and its value, as shown in Example 4-2 on page 106.
- ▶ When using a response file, specify the preference in XML format, as shown in Example 4-3.

Example 4-3 Setting a preference in a response file

```
<preference name='com.ibm.cic.common.core.preferences.eclipseCache'  
value='/InstallationManager/sharedResources' />
```

4.3.2 Repository overview

Installation Manager uses a repository to identify the packages or updates to install. A repository is a location that stores data for installing, modifying, rolling back, updating, or uninstalling packages. Each installed package has an embedded location for its default update repository. You can add, edit, or remove repositories as needed.

Based on the configured repositories, Installation Manager determines the specific packages to install, including products, fix packs, interim fixes, and so on. It checks prerequisites and interdependencies and installs the selected packages.

An Installation Manager repository contains one or more product offerings, each with both metadata and the actual offering payload. The offering metadata describes the following aspects of the offering:

- ▶ Name, version, supported platforms, and so on
- ▶ Required and optional features
- ▶ Relationships and dependencies between offerings and features of offerings

Normally, an Installation Manager repository contains the full content that is required to install the product on various platforms, operating systems, and so on.

Repository topologies can be generalized into these categories:

- ▶ A public repository that is accessible to the general public at an IBM-hosted site, such as IBM Passport Advantage
- ▶ A local repository that is used by a single user and not shared with others
- ▶ An enterprise repository that is located behind the firewall and is accessed by multiple machines within the enterprise

4.3.3 Updating Installation Manager

To update Installation Manager, you must update the installation kit. Refer to 4.2.2, “Obtaining an Installation Manager installation kit” on page 104 for more information.

After you obtain the updated installation kit, refresh the Installation Manager installation using the same commands that you used for the initial installation (**installc**, **userinstc**, or **groupinstc**). For details about these commands, refer to 4.2.3, “Installing Installation Manager on the system” on page 105.

4.3.4 Installing the WebSphere Application Server initial repository

WebSphere Application Server V8.5 has FMID HBBO850 and can be obtained in one of the following ways:

- ▶ Through a ServerPac or SystemPac: Follow the instructions in the *Installing Your Order* guide to create the repository file system <high level qualifier (HLQ)>.SBB0IMR and the <HLQ>.SBB0JCL library.
- ▶ Through a Custom-Built Product Delivery Offering (CBPDO): Run the jobs to install the WebSphere Application Server repository following the instructions that are available at Program Directory to create the repository file system <high level qualifier (HLQ)>.SBB0IMR and the <HLQ>.SBB0JCL library.

Run the following jobs to create the WebSphere Application Server repository:

- ▶ BBORECEV to receive the deliverables
- ▶ BBOISMKD to allocate the system paths
- ▶ BBODDDEF to define de SMP/E DDDEFs
- ▶ BBOAPPLY to apply product repository
- ▶ BBOACCEP to accept product repository

4.4 Using Installation Manager

Before you begin using Installation Manager, consider in which mode it is going to be used and what actions are going to be performed in the software that is going to be maintained. The following options are available:

Command-line mode Use the Installation Manager command line (**imc1**) to manage installations from the `/eclipse/tools` subdirectory.

Silent mode Use this mode to install software easily to multiple systems. Silent mode uses the **imc1** command in conjunction with response files.

You can use all of the commands that we demonstrate in this chapter in z/OS jobs using BPXBATCH.

4.4.1 Key features of Installation Manager

In addition to installing packages, Installation Manager can be used to perform maintenance operations, such as updating, modifying, or rolling back packages. These operations differ slightly from how they are performed on a distributed platform.

Installing packages

When the specific version of a package is not specified in the installation command, Installation Manager checks the designated repository and picks the highest version that is available there. For example, the WebSphere package name for z/OS is `com.ibm.websphere.zOS.v85`.

Due to this approach, if a repository shows more than one WebSphere version and the installation needs to be done at a level that is *not* the highest, you must specify the complete package name, such as `com.ibm.websphere.zOS.v85_8.5.0.20120501_1118`, in your command.

Updating packages

You can update a package as soon as the updates are available in an Installation Manager repository. You can apply the following types of updates:

- ▶ A *fix pack* is a new product level. Each fix pack repository is a delta on top of the previous fix pack level. Fix packs have the same package name but a different version level. A fix pack can be delivered as an SMP/E program temporary fix (PTF) to the initial product repository.
- ▶ An *interim fix* is also known as a patch. Interim fixes use the package name, and they are not available as SMP/E PTF.

You can obtain fix packs and interim fixes by downloading them from the IBM Fix Central website:

<http://www.ibm.com/support/fixcentral/>

When you need to verify the fixes that are available for maintenance, use the `imcl listAvailableFixes` command, as shown in Example 4-4, with the following parameters:

| | |
|-----------------------------------|--|
| <code><package name></code> | The package to be installed with the version |
| <code>-repositories</code> | The repository location |

Example 4-4 Listing available fixes in a repository

```
/InstallationManager/bin/eclipse/tools $ imcl listAvailableFixes
com.ibm.websphere.zOS.v85_8.5.0.20120501_1118 -repositories
/usr/lpp/InstallationManagerRepository/HBB0850
```

Modifying packages

A package can have features, languages, and functions added or removed by Installation Manager. To modify a package, run `imcl install`. To add sample applications to WebSphere Application Server, as shown in Example 4-5 on page 110, run `imcl install` with the following parameters:

| | |
|---|--|
| <code><package name>, <feature name></code> | The package and feature name to be installed |
| <code>-installationDirectory</code> | The directory where the package is installed |
| <code>-repositories</code> | The repository location |

Example 4-5 Adding sample applications to WebSphere Application Server

```
/InstallationManager/bin/eclipse/tools $ imcl install  
com.ibm.websphere.zOS.v85,samples -installationDirectory /usr/lpp/zWebSphere/V8R5  
-repositories /usr/lpp/InstallationManagerRepository/HBB0850
```

Modified com.ibm.websphere.zOS.v85_8.5.0.20120501_1118 in the
/usr/lpp/zWebSphere/V8R5 directory.

To uninstall the sample applications from WebSphere Application Server, as shown in Example 4-6, run **imcl install** with the following parameters:

| | |
|-------------------------------|--|
| <package name> | The package name to be installed |
| -installationDirectory | The directory where the package is installed |

Example 4-6 Uninstalling Application samples from WebSphere Application Server

```
/InstallationManager/bin/eclipse/tools $ imcl uninstall  
com.ibm.websphere.zOS.v85,samples -installationDirectory /usr/lpp/zWebSphere/V8R5
```

To add a language pack to WebSphere Application Server, as shown in Example 4-7, run **imcl install** with the following parameters:

| | |
|------------------------------------|--|
| <package name> | The package name to be installed |
| -properties cic.selector.n1 | The language pack to be installed |
| -installationDirectory | The directory where the package is installed |
| -repositories | The repository location |

Example 4-7 Adding a language pack to WebSphere Application Server

```
/InstallationManager/bin/eclipse/tools $ imcl install com.ibm.websphere.zOS.v85  
-installationDirectory /usr/lpp/zWebSphere/V8R5 -properties cic.selector.n1=de  
-repositories /usr/lpp/InstallationManagerRepository/HBB0850
```

Modified com.ibm.websphere.zOS.v85_8.5.0.20120501_1118 in the
/usr/lpp/zWebSphere/V8R5 directory.

Rolling back packages

To roll back a package, run **imcl install**, and specify the previous product level. For example, if you are running WebSphere at Fix Pack 14 and want to roll it back to Fix Pack 13, issue an installation command that specifies Fix Pack 13.

If there were interim fixes installed at the previous level, you must reinstall them. You can do this using a single command, or you can just install the interim fixes after the rollback is completed.

Important: Take care when interim fixes are involved in building the product's previous level. If interim fixes are available only online, you must specify all of the repositories (local and online) in the installation command and separate their names by commas without any blank spaces in between.

Creating a keyring

Credentials are required for authentication when you access certain URLs, such as to download fixes from IBM Fix Central. To use these credentials, you need a keyring file.

The `imcl` command does not have keyring capability, so use the `imutilsc` command, as shown in Example 4-8, with the following parameters:

| | |
|-----------------------------|---|
| <code>saveCredential</code> | Instructs <code>imutilsc</code> to save credentials |
| <code>-url</code> | The URL that is accessed |
| <code>-userName</code> | The user name that authenticates to the URL |
| <code>-userPassword</code> | The password for the user |
| <code>-keyring</code> | The file where information is stored |

Example 4-8 Creating a keyring

```
/InstallationManager/bin/eclipse/tools $ imutilsc saveCredential -url
http://www.mycorporation.com/repository -userName jsmith -userPassword secret
-keyring /u/jsmith/corporate.keyring
```

Listing installed products

You can list information about the installed products by running `imcl listInstalledPackages`, as shown in Example 4-9.

Example 4-9 Listing the installed packages

```
/InstallationManager/bin/eclipse/tools $ imcl listInstalledPackages
com.ibm.websphere.zOS.v85_8.5.0.20120501_1118
```

When you need to check the features that are installed with a package, use the `imcl listInstalledPackages -feature` command, as shown in Example 4-10.

Example 4-10 Listing the features installed by packages

```
/InstallationManager/bin/eclipse/tools $ imcl listInstalledPackages -features
com.ibm.websphere.zOS.v85_8.5.0.20120501_1118 :
ejbdeploy,embeddablecontainer,thinclient
```

Optionally, you can use the `versionInfo.sh` command to show the same information as the `-feature` option.

For more `imcl` command line arguments, refer to this website:

http://pic.dhe.ibm.com/infocenter/install/v1r5/index.jsp?topic=/com.ibm.cic.commandline.doc/topics/r_tools_imcl.html

4.4.2 Uninstalling Installation Manager

Before you uninstall Installation Manager, you must uninstall all of the software packages that were previously installed using it.

To uninstall Installation Manager:

1. Log in as the Installation Manager owner's user ID.
2. Uninstall all software packages.
3. To uninstall Installation Manager, run `uninstallc`, which is in the agent data directory under the `uninstall` subdirectory, as shown in Example 4-11.

Example 4-11 Uninstall Installation Manager

```
/InstallationManager/appdata/uninstall $ ./uninstallc
```

As an alternative to these step-by-step instructions, you also can uninstall Installation Manager using the GIN2UNIN sample job.

4.5 Installing WebSphere Application Server

You can install WebSphere Application Server V8 using the command line or the supplied jobs. In this section, we describe how to install WebSphere Application Server using the command line.

4.5.1 Installing using the command line

Complete the following steps to install WebSphere Application Server using the command line:

1. Create an empty file system to hold the WebSphere Application Server installation.

You can create the file system manually, or you can use the `zCreateFileSystem.sh` script. The file system needs at least 35,000 tracks (3390) or 1,800 MB. Example 4-12 shows a sample execution using the following parameters:

| | |
|--------------------|---|
| -name | Data set name |
| -type | Type of file system |
| -megabytes | Primary and secondary allocation |
| -volume | Volume where the data set will reside |
| -mountpoint | Directory where the file system will be mounted |
| -owner | Directory owner |
| -group | Group owner |

Example 4-12 Creating an empty file system

```
/InstallationManager/bin/eclipse/tools $ zCreateFileSystem.sh -name OMVS.
BB08558.SBBOHFS -type ZFS -megabytes 1800 200 -volume TARHF1 -mountpoint
/usr/lpp/zWebSphere/V8R5 -owner STC -group TSO
CWLCS9023I Defining file system OMVS.BB08558.SBBOHFS .
IOEZ00248I VSAM linear dataset OMVS.BB08558.SBBOHFS successfully created.
CWLCS9024I File system OMVS.BB08558.SBBOHFS successfully defined.
CWLCS9022I Formatting ZFS file system OMVS.BB08558.SBBOHFS.
IOEZ00077I HFS-compatibility aggregate OMVS.BB08558.SBBOHFS has been
successfully created
CWLCS9012I Creating mount point directory /usr/lpp/zWebSphere/V8R5.
CWLCS9013I Mount point directory /usr/lpp/zWebSphere/V8R5 successfully created.
CWLCS9006I Mounting data set OMVS.BB08558.SBBOHFS at mount point
/usr/lpp/zWebSphere/V8R5.
CWLCS9007I OMVS.BB08558.SBBOHFS successfully mounted at mount point
/usr/lpp/zWebSphere/V8R5.
CWLCS9017I Setting owner and group for directory /usr/lpp/zWebSphere/V8R5.
CWLCS9018I Owner and group successfully set for directory
/usr/lpp/zWebSphere/V8R5.
CWLCS9019I Setting permissions for directory /usr/lpp/zWebSphere/V8R5.
CWLCS9020I Permissions successfully set for directory /usr/lpp/zWebSphere/V8R5.
```

As an alternative, you can use the BBO1CFS job to create the file system instead of using the command line.

2. Confirm that the repository has the needed packages by running `imcl listAvailablePackages` from the `/InstallationManager/bin/eclipse/tools` directory, as shown in Example 4-13. Use the `repositories` parameter to set the repository location.

Example 4-13 Listing available packages on a repository

```
/InstallationManager/bin/eclipse/tools $ imcl listAvailablePackages
-repositories /usr/lpp/InstallationManagerRepository/HBB0850
com.ibm.websphere.IHS.zOS.v85_8.5.0.20120501_1121
com.ibm.websphere.NDDMZ.zOS.v85_8.5.0.20120501_1118
com.ibm.websphere.PLG.zOS.v85_8.5.0.20120501_1122
com.ibm.websphere.zOS.v85_8.5.0.20120501_1118
```

Important: You can find the license for a package in the `lafiles` subdirectory of the Installation Manager repository. Be sure to check it before installing products.

3. Install WebSphere Application Server by running `imcl install`, as shown in Example 4-14, using the following parameters:

| | |
|--|--|
| <code><package name></code> | The package to be installed |
| <code>-installationDirectory</code> | The directory where the package is installed |
| <code>-repositories</code> | The repository location |
| <code>-sharedResourcesDirectory</code> | The directory where the artifacts from repository are stored |

Important: The `-sharedResourcesDirectory` parameter can be omitted in subsequent commands after the shared resources directory is set for the first time.

| | |
|-----------------------------|--|
| <code>-acceptLicense</code> | To accept the software license agreement |
|-----------------------------|--|

Example 4-14 WebSphere Application Server installation

```
/InstallationManager/bin/eclipse/tools $ imcl install com.ibm.websphere.zOS.v85
-installationDirectory /usr/lpp/zWebSphere/V8R5 -repositories
/usr/lpp/InstallationManagerRepository/HBB0850 -sharedResourcesDirectory
/InstallationManager/sharedResources -acceptLicense
```

Installed `com.ibm.websphere.zOS.v85_8.5.0.20120501_1118` to the `/usr/lpp/zWebSphere/V8R5` directory.

As an alternative, you can use the `BBO1INST` job to install WebSphere Application Server instead of using the command line.

4. After installing WebSphere Application Server, mount and remount the product file system in read-only mode for use by WebSphere Application Server nodes and servers.

4.5.2 Installing additional packages

WebSphere Application Server includes the following additional packages:

- ▶ DMZ secure proxy
- ▶ IBM HTTP server
- ▶ Web server plug-ins

You can install these packages using the following jobs:

- ▶ `BBO2CFS` to allocate a file system for the DMZ secure proxy

- ▶ BBO2INST to install the DMZ secure proxy
- ▶ BBO3CFS to allocate a file system for the web server plug-ins
- ▶ BBO3INST to install the web server plug-ins
- ▶ BBO4CFS to allocate a file system for the IBM HTTP Server
- ▶ BBO4INST to install the IBM HTTP Server

4.5.3 Creating response files

You can create a response file that will contain all of the installation commands needed to install software with Installation Manager. This method allows you to reuse the commands to perform the installation on several machines.

A response file can be created using sample files or by recording one during an Installation Manager operation.

Using sample files

Some products deliver a sample file to use during the installation process. If a sample file is not available, as is the case with WebSphere Application Server, you can create one using the samples that are available at the product information center at this website:

http://pic.dhe.ibm.com/infocenter/install/v1r5/index.jsp?topic=%2Fcom.ibm.silentinstall112.doc%2Ftopics%2Fc_sample_response_files.html

Choose one of the scenarios and adapt the sample file to your installation. The commands that you can use in response files are available at this website:

http://pic.dhe.ibm.com/infocenter/install/v1r5/index.jsp?topic=%2Fcom.ibm.silentinstall112.doc%2Ftopics%2Fr_silent_inst_commands12.html

Recording a response file

During an Installation Manager operation, a response file can be recorded for later reuse. Example 4-15 shows how to generate a response file during the WebSphere Application Server V8.5 package installation. It uses the following parameters:

| | |
|-------------------------------|---|
| <command> | Specifies the action that must be taken |
| <package name> | Specifies the package that is handled by the <command> action |
| -repositories | Specifies the repository to use |
| -record | Specifies the file name that will be generated |
| -acceptLicense | Specifies that you agree to the license agreement |
| -installationDirectory | Specifies in which directory the package is installed |

Example 4-15 Record a response file when installing a package

```
/InstallationManager/bin/eclipse/tools $ imcl install com.ibm.websphere.zOS.v85
-installationDirectory /usr/lpp/zWebSphere/V8R5 -repositories
/usr/lpp/InstallationManagerRepository/HBB0850 -record /tmp/WAS_inst.xml
-sharedResourcesDirectory /InstallationManager/sharedResources -acceptLicense
```

The resulting file has all of the instructions needed to install the web server plug-in package, as shown in Example 4-16.

Example 4-16 Response file for WebSphere Application Server V8.5 installation

```
<xml version="1.0" encoding="UTF-8"?>
<agent-input>
```

```

<server>
<repository location='/usr/lpp/InstallationManagerRepository/HBB0850' />
</server>
<profile id='IBM WebSphere Application Server V8.5'
installLocation='/usr/lpp/zWebSphere/V8R5'>
<data key='eclipseLocation' value='/usr/lpp/zWebSphere/V8R5' />
<data key='user.import.profile' value='false' />
<data key='cic.selector.os' value='zos' />
<data key='cic.selector.ws' value='motif' />
<data key='cic.selector.arch' value='s390' />
<data key='cic.selector.nl' value='de' />
</profile>
<install modify='true'>
<offering id='com.ibm.websphere.zOS.v85' version='8.5.0.20120501_1118'
profile='IBM WebSphere Application Server V8.5'
features='core.feature,ejbdeploy,thinclient,embeddablecontainer' />
</install>
<preference name='com.ibm.cic.common.core.preferences.eclipseCache'
value='/InstallationManager/sharedResources' />
<preference name='com.ibm.cic.common.core.preferences.connectTimeout' value='30' />
<preference name='com.ibm.cic.common.core.preferences.readTimeout' value='45' />
<preference name='com.ibm.cic.common.core.preferences.downloadAutoRetryCount'
value='0' />
<preference name='offering.service.repositories.areUsed' value='true' />
<preference name='com.ibm.cic.common.core.preferences.ssl.nonsecureMode'
value='false' />
<preference
name='com.ibm.cic.common.core.preferences.http.disablePreemptiveAuthentication'
value='false' />
<preference name='http.ntlm.auth.kind' value='NTLM' />
<preference name='http.ntlm.auth.enableIntegrated.win32' value='true' />
<preference name='com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts'
value='true' />
<preference name='com.ibm.cic.common.core.preferences.keepFetchedFiles'
value='false' />
<preference name='PassportAdvantageIsEnabled' value='false' />
<preference name='com.ibm.cic.common.core.preferences.searchForUpdates'
value='false' />
<preference name='com.ibm.cic.agent.ui.displayInternalVersion' value='false' />
<preference name='com.ibm.cic.common.sharedUI.showErrorLog' value='true' />
<preference name='com.ibm.cic.common.sharedUI.showWarningLog' value='true' />
<preference name='com.ibm.cic.common.sharedUI.showNoteLog' value='true' />
</agent-input>

```

4.5.4 Installing silently

You can use silent installation mode to perform software deployment to multiple systems. To do this, complete the following steps:

1. Install Installation Manager.
2. Generate a response file.
3. Manage your packages silently.

After Installation Manager is installed and a response file is recorded, you can use the response file to make new installations, as shown in Example 4-17. Run `imcl` with the following parameters:

| | |
|-----------------------|---|
| input | Specifies which response file to be used |
| -log | Specifies the log file to be generated |
| -acceptLicense | Specifies that you agree to the license agreement |

Example 4-17 Package installation in silent mode

```
/InstallationManager/bin/eclipse/tools $ imcl input /tmp/WAS_inst.xml -log
/tmp/silent.log -acceptLicense
```

4.5.5 The post-installer

For z/OS systems, WebSphere Application Server sometimes requires that service-applied changes be made to configuration files. The post-installer can update the configuration files automatically or manually. You can use the post-installer to complete the following tasks:

- ▶ Run configuration actions by Installation Manager during installation or uninstallation
- ▶ Automatically detect the application or removal of fix packs, and run any necessary configuration actions
- ▶ Automatically detect the addition or removal of an offering, optional feature, or interim fix, and then create or remove any associated symbolic links

Installation Manager considers the post-installation step as a nonfatal step. Thus, if the post-installer returns a FAIL or PARTIAL SUCCESS return code, Installation Manager displays the following message:

The packages are installed with warnings

4.5.6 Service information

The Preventive Service Planning (PSP) database can be searched for specific installation tips, high-impact or pervasive problems, and service recommendations. Information about both software and hardware for the System z family of servers is in the database at the following website:

<http://www14.software.ibm.com/webapp/set2/psearch/search?domain=psp>

To find the proper information, complete the Upgrade Name and Subset name fields with the values for WebSphere Application Server and Installation Manager listed in Table 4-2.

Table 4-2 PSP information for WebSphere Application Server and Installation Manager

| Product | Upgrade name | Subset name |
|------------------------------|--------------|-------------|
| WebSphere Application Server | WASAS850 | HBBO850 |
| Installation Manager | IIMZOSV1 | HGIN140 |

Figure 4-4 on page 117 shows a PSP search result for WebSphere Application Server 8.5.

Upgrade WASAS850, Subset HBBO850

Preventive Service Planning

Upgrade WASAS850, Subset HBBO850

[Service Recommendation Summary](#)
[Installation Information](#)
[Documentation Changes](#)
[General Information](#)
[Service Recommendations](#)
[Cross Product Dependencies](#)
[Informational/Documentation](#)
[PTF/APAR Reference Lists](#)

Upgrade WASAS850, Subset HBBO850:

This subset contains installation information for WebSphere [Application Server](#) for z/OS DMZ Proxy Server Version 8, Release 5, Modification 0.

Figure 4-4 Installation tips from Preventive Service Planning

4.5.7 Uninstalling packages

You can uninstall packages using the command line or using supplied jobs. Here, we describe how to uninstall WebSphere Application Server using the command line.

To uninstall WebSphere Application Server:

1. From the /InstallationManager/bin/eclipse/tools directory, run **imcl listInstalledPackages** to verify that the package is installed, as shown in Example 4-18.

Example 4-18 Listing installed packages

```
/InstallationManager/bin/eclipse/tools $ imcl listInstalledPackages
com.ibm.websphere.zOS.v85_8.5.0.20120501_1118
```

2. Initiate the uninstallation by running **imcl uninstall**, as shown in Example 4-19, using the following parameters:

<package name> The package to be uninstalled
-installationDirectory The directory where the package is installed

Example 4-19 WebSphere Application Server uninstallation

```
/InstallationManager/bin/eclipse/tools $ imcl uninstall
com.ibm.websphere.zOS.v85 -installationDirectory /usr/lpp/zWebSphere/V8R5
Uninstalled com.ibm.websphere.zOS.v85_8.5.0.20120501_1118 from the
/usr/lpp/zWebSphere/V8R5 directory.
```

As an alternative, you can use the BBO1UNIN job to uninstall WebSphere Application Server instead of using the command line.

The uninstallation jobs for the other WebSphere components are:

- ▶ BBO2UNIN to uninstall the DMZ secure proxy
- ▶ BBO3UNIN to uninstall the web server plug-ins
- ▶ BBO4UNIN to uninstall the IBM HTTP Server

4.5.8 Preparing the base z/OS operating system

After installing WebSphere Application Server, you must prepare the z/OS system. Because of extensive use of underlying z/OS services for security, reliability, and performance, consider performing the following tasks:

- ▶ Prepare z/OS operating system settings
- ▶ Prepare z/OS sysplex settings
- ▶ Prepare the z/OS job entry subsystem (JES)
- ▶ Set up Resource Recovery Services (RRS)
- ▶ Set up Resource Access Control Facility (RACF)
- ▶ Prepare TCP/IP
- ▶ For IBM DB2® database, set up DB2 for concurrency control management

For the complete list of tasks to prepare z/OS target systems, visit the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.installation.zseries.doc%2Fae%2Fwelc_howdoi_tins.html

4.6 WebSphere Customization Toolbox

The WebSphere Customization Toolbox for WebSphere Application Server V8.5 includes tools for managing, configuring, and migrating various parts of your WebSphere Application Server environment. The WebSphere Customization Toolbox includes the following features:

- ▶ Web Server Plug-ins Configuration Tool to configure web server plug-ins
- ▶ Profile Management Tool (z/OS only) to generate jobs and instructions for creating profiles (Intel-based Windows or Linux operating systems)
- ▶ z/OS Migration Management Tool to generate definitions for migrating WebSphere Application Server for z/OS profiles (Intel-based Windows or Linux operating systems)

For details about using the WebSphere Customization Toolbox, refer to Chapter 5, “Working with profiles on z/OS systems” on page 121.

4.7 Troubleshooting

To diagnose an Installation Manager problem, go to the Installation Manager binaries location and look into the config.ini file in the configuration directory. The following line in config.ini gives the location of the runtime data (appdata):

```
cic.appDataLocation=/InstallationManager/appdata
```

The Installation Manager logs are in <appdata>/logs and are named <date>_<time>.xml. These files are in ASCII. They can be viewed on a z/OS system, or you can download them in binary, together with the log.xsl file, to view from a web browser on your workstation.

Be sure to verify that any target (product) file systems required for product installation are present, mounted read/write, and have the correct ownership and permissions.

If Installation Manager reports insufficient space in a file system that you recognize as belonging to Installation Manager or a product file system, you can add space as necessary. If Installation Manager reports that more space is needed in the root file system (/) or some other unexpected location, this probably means that a needed file system is not mounted.

4.7.1 Error message overview

Error message IDs are composed of a prefix, a number, and a message type.

Examples of prefixes include:

- ▶ CRIMA: IBM Installation Manager
- ▶ CRIMC: Common messages shared by Installation Manager and IBM Packaging Utility
- ▶ CRIMD: Installation Manager integration with IBM WebSphere Application Server
- ▶ CRIMG: Packaging Utility

Example 4-20 shows a sample error message

Example 4-20 Prefix, number, and message type in an error message

```
CRIMA5096821AE ERROR: Error installing
```

4.7.2 Collecting Installation Manager information

The `imutilsc exportInstallData` command can be used to create a condensed file containing a variety of critical Installation Manager information. Provide this file to the IBM Support Center whenever you report an Installation Manager problem or defect.



Working with profiles on z/OS systems

The information in this chapter can help you build your initial WebSphere Application Server environment using the new WebSphere Customization Toolbox.

This chapter includes the following topics:

- ▶ Creating WebSphere environments for z/OS
- ▶ Getting started with the Profile Management tool
- ▶ Creating a sample z/OS Network Deployment cell
- ▶ Creating a deployment manager definition
- ▶ Creating the base application server definition
- ▶ Federating an application server
- ▶ Creating a job manager profile
- ▶ Creating an administrative agent profile

5.1 Creating WebSphere environments

Configuring a WebSphere Application Server for z/OS environment consists of setting up the configuration directory for the environment and making changes to the z/OS target system that pertain to the particular application serving environment. Configuring these application serving environments after product installation requires a fair amount of planning and coordination. For example, when defining multiple deployment managers or application servers on a single machine or LPAR, you must ensure that the ports and names you select for each one are unique and that the z/OS environment variables, generated jobs, and so on, are set up properly. Spend time planning the installation and, if possible, practice first by configuring a stand-alone application server using the default options.

You use the WebSphere Application Server for z/OS Profile Management tool available with the WebSphere Customization Toolbox to configure WebSphere Application Server environments for the z/OS platform.

The Profile Management tool is a dialog-driven tool that runs in the WebSphere Customization Toolbox. It is an Eclipse plug-in that allows you to perform the initial set up of WebSphere Application Server for z/OS cells and nodes. It provides the same functionality as the former ISPF dialog boxes with additional features.

The WebSphere Customization Toolbox itself does not create the cells and nodes; however, it creates batch jobs, scripts, and data files that you can use to perform WebSphere Application Server for z/OS customization tasks. These jobs, scripts, and data files form a *customization definition* on your workstation, which is then uploaded to z/OS where you submit the jobs, as shown in Figure 5-1 on page 123.

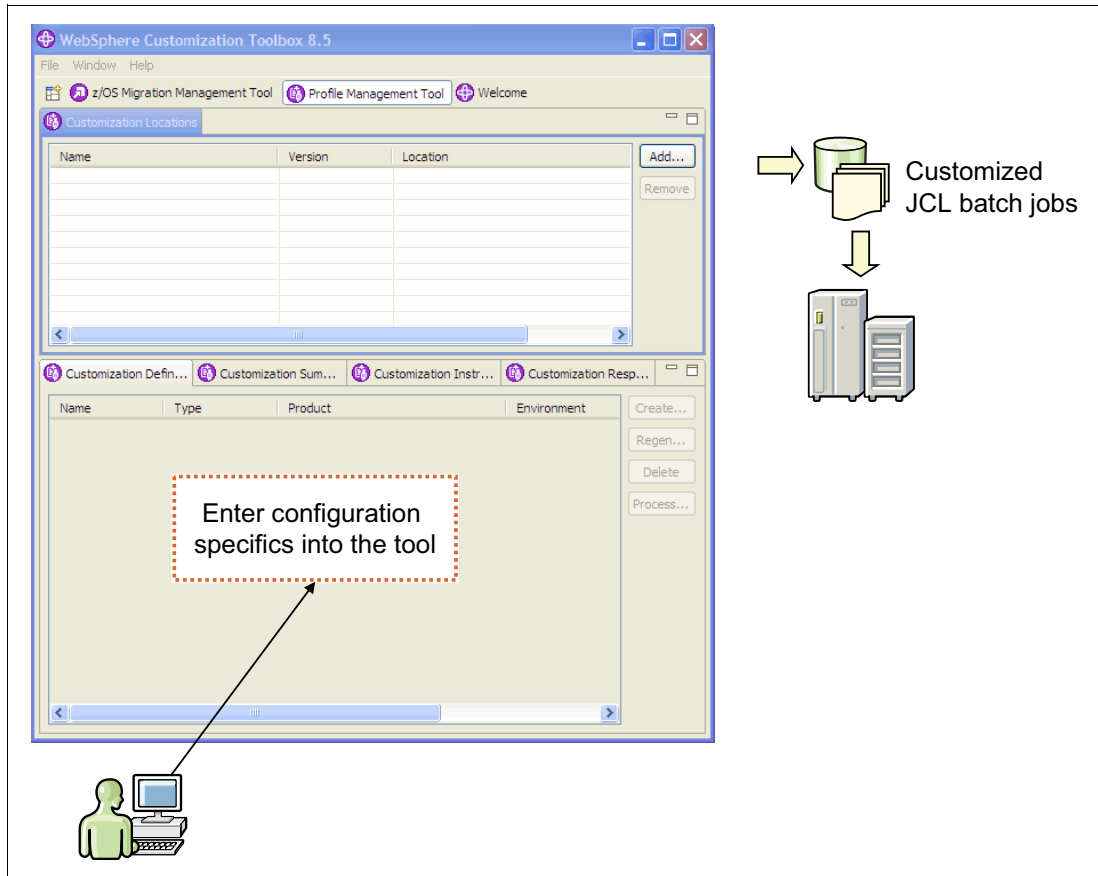


Figure 5-1 The WebSphere Customization Toolbox configuration flow

Review the documentation: The WebSphere Application Server information center contains planning topics for each WebSphere Application Server package that is tailored to each platform. This section gives you a high-level look at the planning tasks that you must perform.

If you are planning a WebSphere Application Server for z/OS environment, review the installation planning material for z/OS platforms that is available at the following website:

<http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.installation.zseries.doc%2Finfo%2Fzseries%2Fae%2Fwlc6topinstalling.html>

You can use a spreadsheet to create and record configuration variables and plan many of the values that you need to specify to the Profile Management tool. You can download a template spreadsheet from the following website:

<http://www-03.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS4944>

Using the Profile Management tool, you can create environments for the following products:

- ▶ WebSphere Application Server for z/OS
- ▶ WebSphere DMZ secure proxy server for z/OS

5.1.1 WebSphere Application Server for z/OS

The Profile Management tool provides support to generate jobs to create the following WebSphere Application Server for z/OS environments:

- ▶ Create a cell environment consisting of a deployment manager and a federated application server.
- ▶ Create a management environment, which can be either:
 - A deployment manager
 - A job manager
 - An administrative agent
- ▶ Create a stand-alone application server environment.
- ▶ Create a managed (custom) node and federate the node into an existing cell.
- ▶ Federate an application server.

5.1.2 WebSphere DMZ secure proxy server for z/OS

Using the DMZ secure proxy server for an IBM WebSphere Application Server installation, you can install your proxy server in the DMZ, while reducing the security risk that might occur if you choose to install an application server in the DMZ to host a proxy server. The risk is reduced by removing any functionality from the application server that is not required to host the proxy servers, but this reduction in security can also pose a security risk. Installing the secure proxy server in the DMZ rather than the secured zone presents new security challenges. However, the secure proxy server is equipped with capabilities to provide protection from these challenges.

The Profile Management tool for WebSphere Application Server V8.5 provides support for the following WebSphere DMZ secure proxy server for z/OS environments:

- ▶ Management
 - Generates the customization jobs to create an administrative agent for the secure proxy server.
- ▶ Secure proxy
 - Generates the customization jobs to create a secure proxy server.

5.2 Getting started with the Profile Management tool

This section explains how to prepare and start the Profile Management tool. These steps are common for any type of profile.

To start the Profile Management tool:

1. Start the WebSphere Customization Toolbox.

On Windows systems, click **Start** → **All Programs** → **IBM WebSphere** → **WebSphere Customization Toolbox V8.5** → **WebSphere Customization Toolbox**.

On Linux, click **operating_system_menus_to_access_programs** → **IBM WebSphere** → **WebSphere Customization Toolbox V8.5** → **WebSphere Customization Toolbox**.

2. Click the **Welcome** tab. Click **Profile Management Tool (z/OS only)**, as shown in Figure 5-2. Click **Launch Selected Tool**.

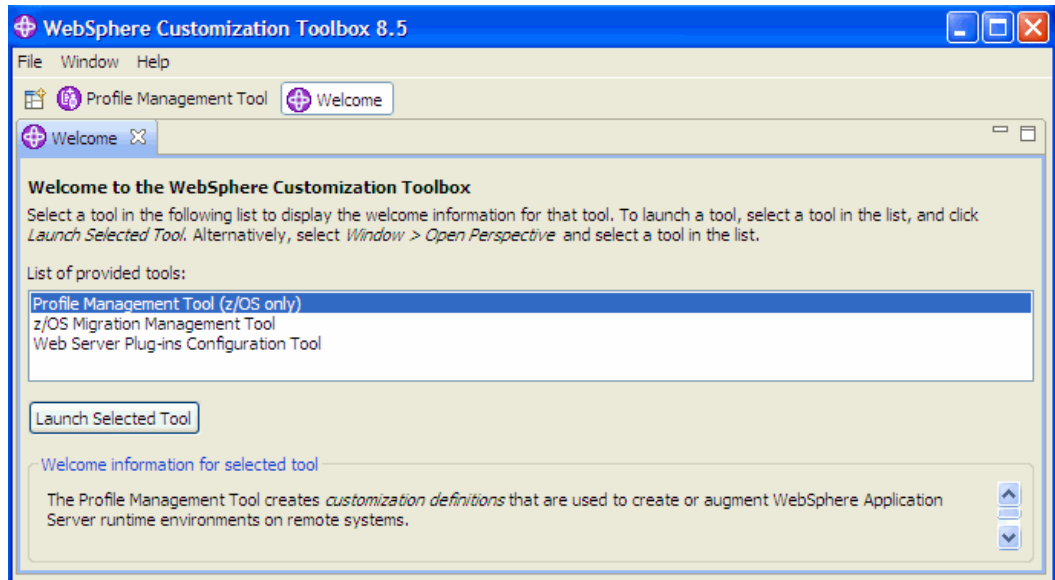


Figure 5-2 WebSphere Customization Toolbox welcome

3. On the Customization Locations tab, create a new location by clicking **Add**, as shown in Figure 5-3.

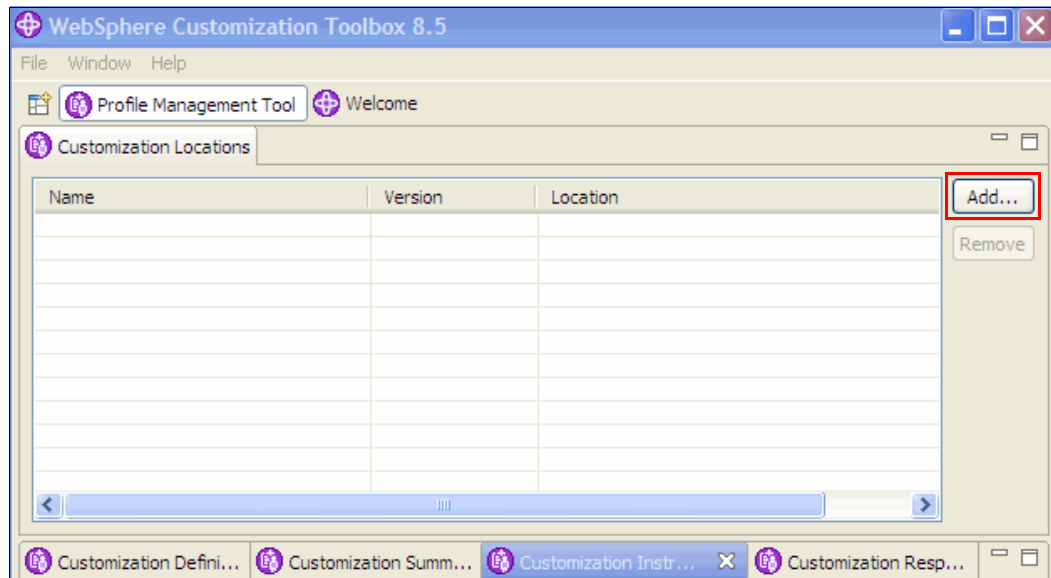


Figure 5-3 Add a new location

The definitions that contain the generated customization jobs are stored at the location that you define in the Add Customization dialog box. Enter the following information, as shown in Figure 5-4:

- a. Select **Create a new customization location**, and enter a name for the new location.
- b. Click **8.5** from the Version drop-down menu, and click **Browse** to define the corresponding folder for this location.
- c. Click **Finish** to create this location.

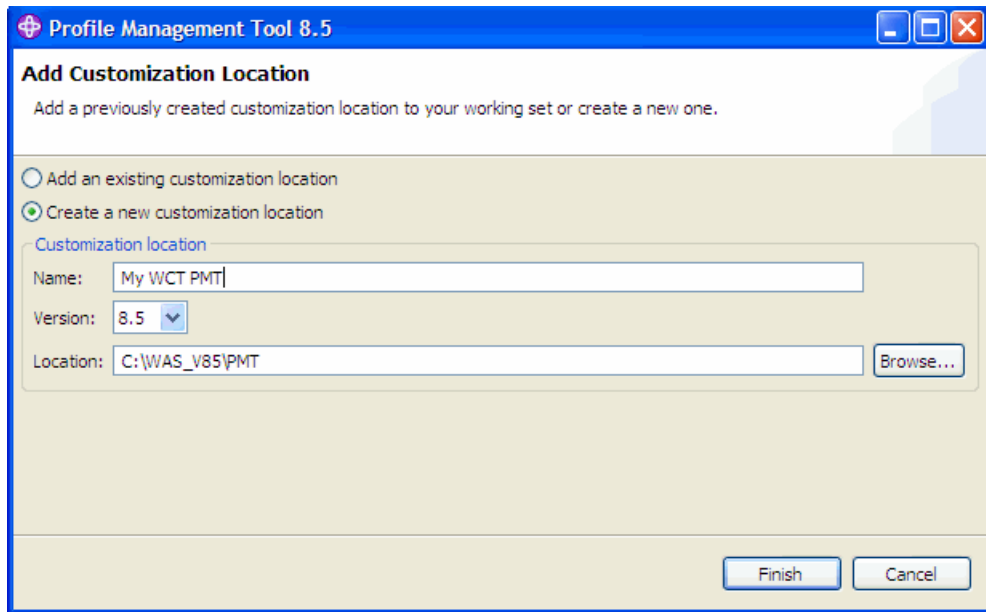


Figure 5-4 Create the customization location

Tip: You can create a single location to hold all of the definitions. Depending on the size of your environment, using a single location for all definitions can become confusing. Consider creating one location per cell definitions as a method of organizing your definitions.

The Profile Management Tool main windows shows the new Customization Location, as shown in Figure 5-5.

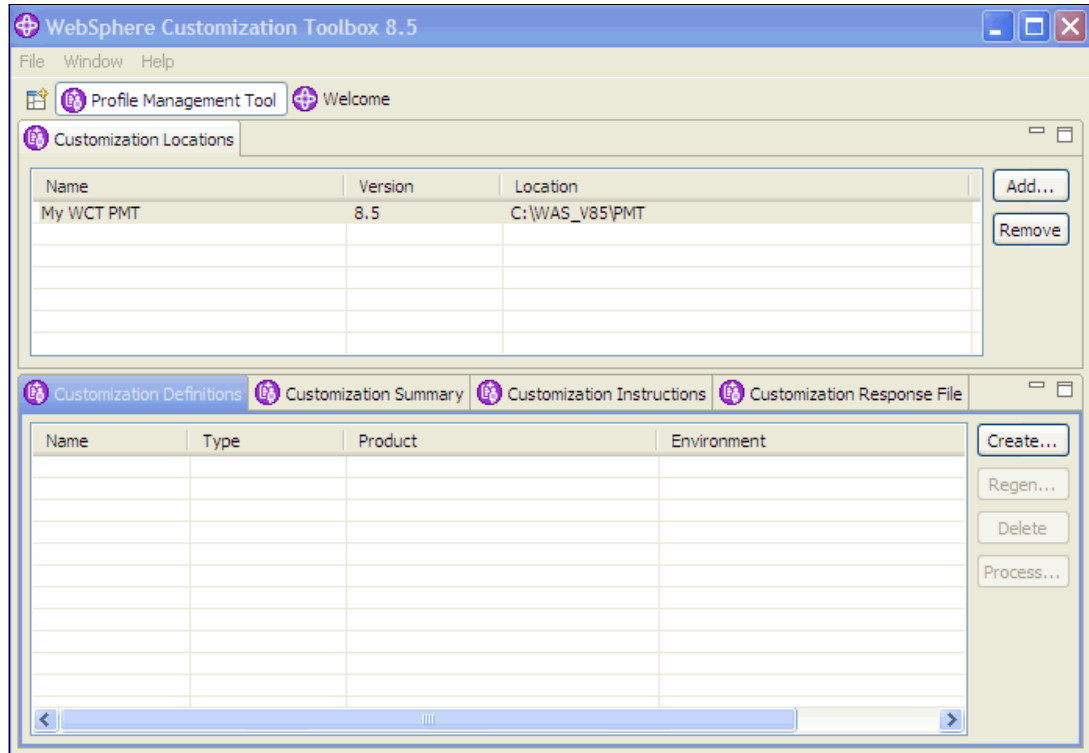


Figure 5-5 Profile Management Tool main window

Using response files: Each time you create a customization definition, it is stored in a directory in the selected customization location. A corresponding response file is generated in the same directory.

You can display this response file by switching to the Customization Response File tab shown in Figure 5-5, which is the last tab in the bottom portion of the window. You can use this response file when creating future profiles to populate the input fields with values contained in the response file.

Configuring additional users: If your daemon and control region adjunct processes must run using different user IDs from the associated control region process, click **Window** → **Preferences** → **Profile Management Tool** in the WebSphere Customization Toolbox. Next, select **Enable unique user IDs for daemon and adjunct**, and click **Apply**.

With this setting, you get an additional window where you build a customization definition that allows you to specify additional user IDs for processes that are relevant to the profile (that is, for the controller adjunct and daemon processes).

5.3 Creating a sample z/OS Network Deployment cell

This section demonstrates how to use the Profile Management tool to create a z/OS Network Deployment cell (deployment manager and application server). This configuration is

representative and inclusive of the windows and steps that you will encounter with the other environments.

Figure 5-6 shows the flow to generate the jobs for this sample configuration.

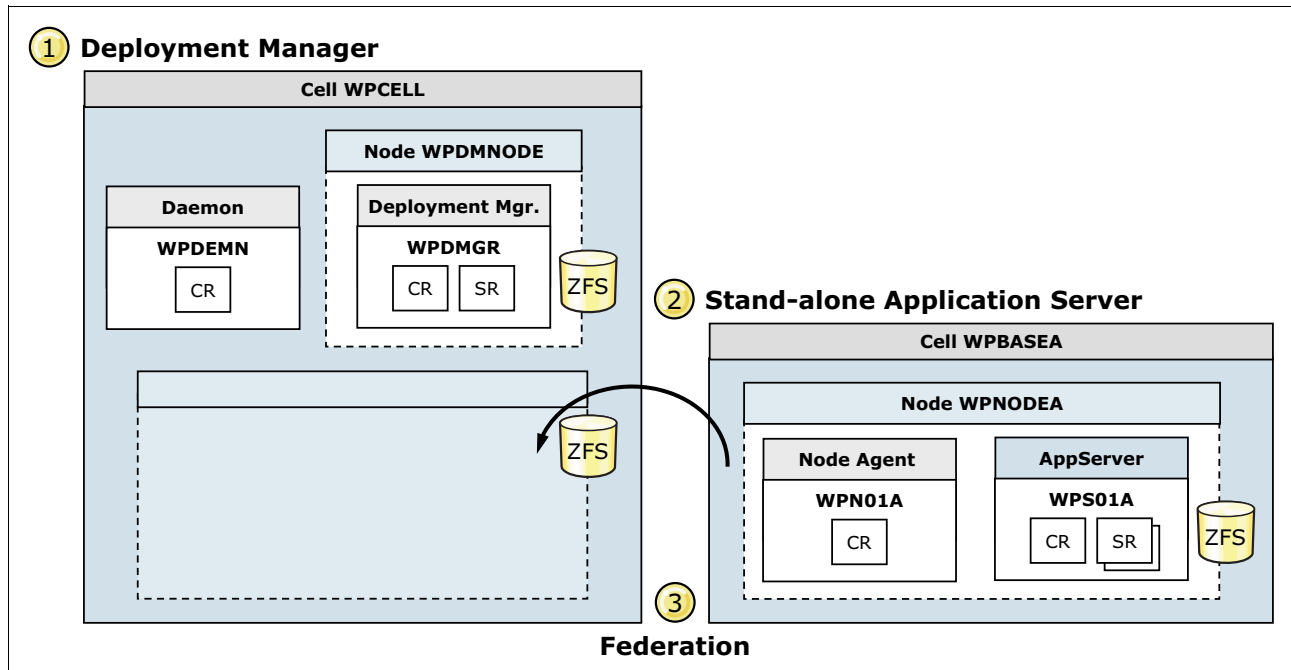


Figure 5-6 Configuration objectives

Building the sample environment involves the following steps:

1. Creating a deployment manager definition
2. Creating the base application server definition
3. Federating an application server

5.3.1 Creating a deployment manager definition

When building the sample environment, you first create the deployment manager definition.

Creating the customization definition

To create the customization definition:

1. Run the Profile Management tool to create the custom definition.
2. On the Profile Management Tool main window, click **Create**.

3. Click **Management** (as shown in Figure 5-7), and then click **Next**.

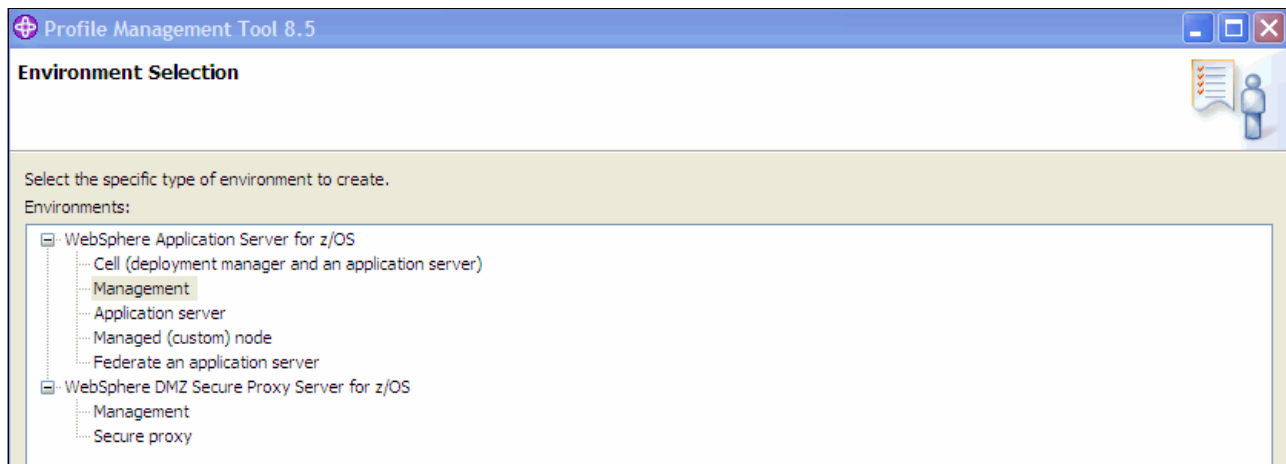


Figure 5-7 Environment Selection window

4. Select **Deployment manager**, and then click **Next**.

5. In the Customization Definition Name window, shown in Figure 5-8, complete the following fields:

- Customization definition name

Specify the customization profile that you are about to create. This name is not transported to your host system.

- Response file path name

Specify a saved file with values from a previously created configuration. Using a previously created configuration populates the fields throughout the windows that follow with the values that are contained in the response file. This field is optional.

Because you are creating a profile for the first time, you might not have this file. A response file is written each time a z/OS customization definition is created, and its name is the customization definition name itself with the extension *.responseFile*. The file is created in the root directory for the customization definition. Normally, you specify a response file from a customization definition of the same type as the definition that you are about to define. However, you can use a response file from a similar customization type to pre-load most of the default values.

After you complete the fields, click **Next**.

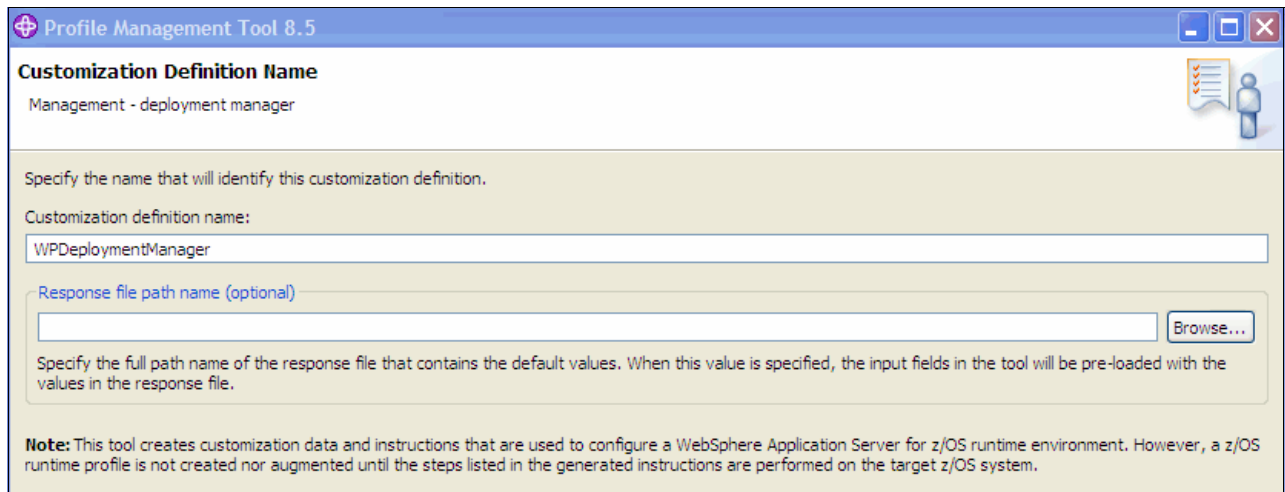


Figure 5-8 Customization Definition Name

6. In the Default values window (shown in Figure 5-9), specify defaults for GID and UID values, name, and user ID defaults based on a two-character prefix that identifies the cell, and specify a default range for ports that are assigned to the process. Click **Next**.

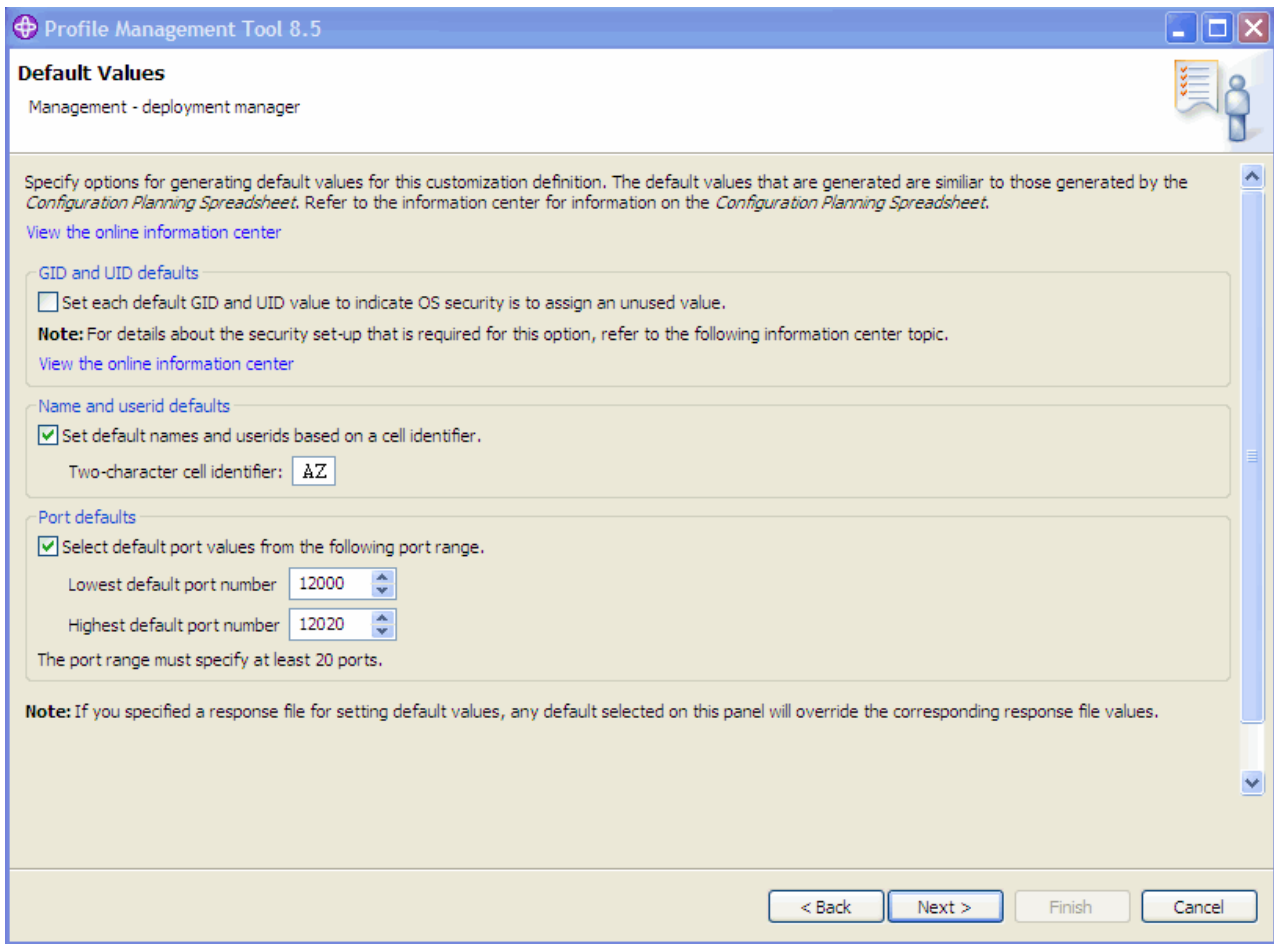


Figure 5-9 Specify default values

7. In the Target Data Sets window (shown in Figure 5-10), specify a high-level qualifier for the target z/OS data sets that will contain the generated jobs and instructions.

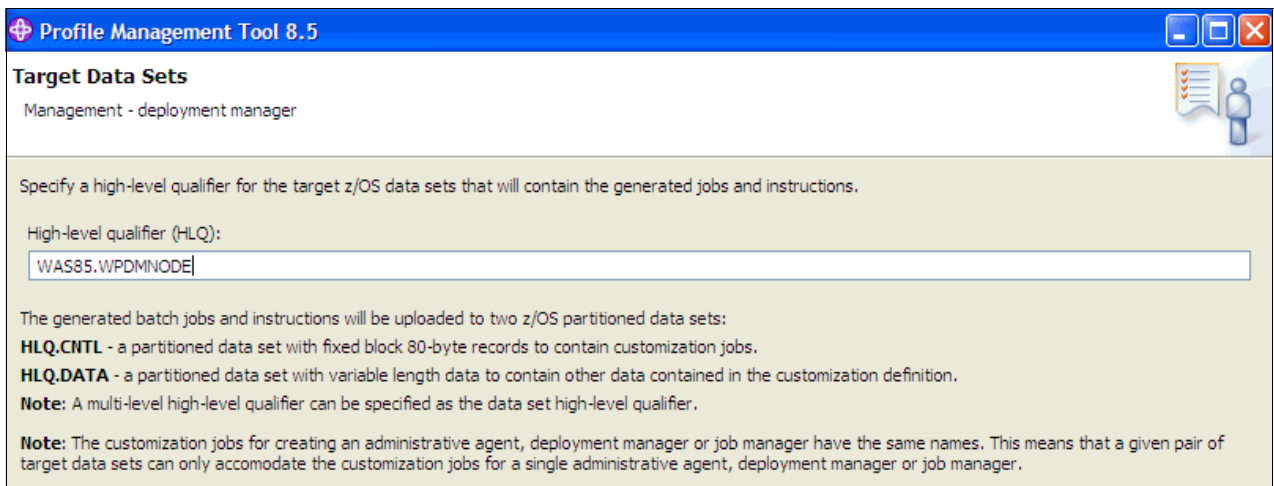


Figure 5-10 Target Data Sets

The high-level qualifier can be composed of multiple qualifiers, up to 39 characters. When a customization profile is uploaded on the target z/OS system, the generated jobs and files are written on a pair of data sets. The same data sets can be reused for a future installation; however, as a best practice, create a new pair of data sets for every new profile installation.

A good planning and naming convention is crucial when defining this type of information. As a best practice, try to set the high-level qualifier according to the version and release of WebSphere Application Server for z/OS, the task you are performing, and the cell (and, in some cases, the node name) you are configuring.

In this case, the following data sets are created when the customization profile is uploaded to the target z/OS system:

- WAS85.WPDMNODE.CNTL
- WAS85.WPDMNODE.DATA

The CNTL data set is a partitioned data set with a fixed block 80-byte records that keeps the customization jobs. The DATA data set is a partitioned data set as well but with variable length data to contain the other customization data.

Click **Next**.

Data set names: After you create the customization profile, you cannot change the data set names because all jobs are based on these data set names.

8. The Configure Common Groups window (shown in Figure 5-11) contains the fields to configure common groups.

Profile Management Tool 8.5

Configure Common Groups

Management - deployment manager

WebSphere Application Server configuration group information

Group: WPCFG

GID

Allow OS security to assign GID

Assign user-specified GID: 2500

WebSphere Application Server servant group information

Group: WPSRVG

GID

Allow OS security to assign GID

Assign user-specified GID: 2501

WebSphere Application Server local user group information

Group: WPGUESTG

GID

Allow OS security to assign GID

Assign user-specified GID: 2502

< Back Next > Finish Cancel

Figure 5-11 Configure Common Groups

Provide the following information for this window:

- WebSphere Application Server Configuration Group Information
Specifies the group name for the WebSphere Application Server administrator user ID and all server user IDs.
- WebSphere Application Server Servant Group Information
Connects all servant user IDs to this group. You can use it to assign subsystem permissions, such as DB2 authorizations, to all servants in the security domain.
- WebSphere Application Server Local User Group Information
Specifies the local client group. This group provides minimal access to the cell.

Click **Next**.

9. The Configure Common Users window (shown in Figure 5-12) contains configuration information about the common users.

Profile Management Tool 8.5

Configure Common Users

Management - deployment manager

WebSphere Application Server user ID home directory:
/var/WebSphere/home

Common controller user ID

User ID: WPACRU

UID

Allow OS security to assign UID

Assign user-specified UID: 2431

Common servant user ID

User ID: WPASRU

UID

Allow OS security to assign UID

Assign user-specified UID: 2432

WebSphere Application Server administrator

User ID: WPADMIN

UID

Allow OS security to assign UID

Assign user-specified UID: 2403

< Back Next > Finish Cancel

Figure 5-12 Configure Common Users

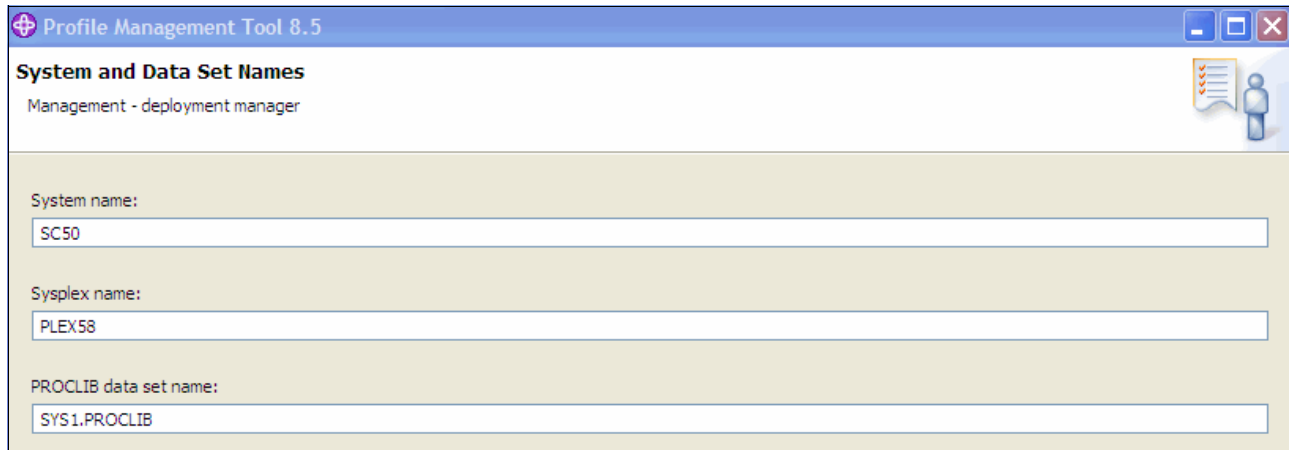
In this window, enter the following information:

- WebSphere Application Server user ID home directory
Specify a new or existing z/OS file system directory in which home directories for WebSphere Application Server for z/OS user IDs are created by the customization process. Note that this directory does not need to be shared among z/OS systems in a WebSphere Application Server cell.
- Common controller user ID
Specifies the user ID that is associated with all the control regions and the daemon. This user ID also owns all of the configuration file systems.
Common servant user ID
Specifies the user ID that is associated with the servant regions.
- WebSphere Application Server administrator user ID
Defines the initial WebSphere Application Server administrator. This ID must have the WebSphere Application Server configuration group as its default UNIX System

Services group. The UNIX System Services UID number for the administrator user ID is specified here, and must be a unique numeric value between 1 and 2,147,483,647.

Click **Next**.

10. The System and Data Set Names window (shown in Figure 5-13) requests system and data set names.



The screenshot shows a window titled "Profile Management Tool 8.5" with a subtitle "System and Data Set Names". Below the subtitle is the text "Management - deployment manager". There are three input fields: "System name" with the value "SC50", "Sysplex name" with the value "PLEX58", and "PROCLIB data set name" with the value "SYS1.PROCLIB".

Figure 5-13 System and Data Set Names

In this panel, complete the following information:

- System name: The system name of the target z/OS system
- Sysplex name: The sysplex name of the target z/OS system

System and sysplex names: If you are not sure of the system and sysplex names for your target z/OS system, use the D SYMBOLS console command on the target z/OS system to display them.

- PROCLIB data set name: The PROCLIB data set where the WebSphere Application Server for z/OS cataloged procedures are to be added.

Click **Next**.

11. The Cell, Node and Server Names window (shown in Figure 5-14) requests the cell, node, and server names.

Profile Management Tool 8.5

Cell, Node and Server Names
Management - deployment manager

Cell names
Short name: WPCCELL
Long name: wpcell

Note: Each management server (administrative agent, deployment manager or job manager) should be assigned its own cell name, different from that of any other WebSphere Application Server cell(s) on the same z/OS sysplex.

Node names
Short name: WPDMMODE
Long name: wpdmmode

Server names
Short name: WPDMMGR
Long name: dmgr

Cluster transition name:
WPDMMGR

Figure 5-14 Cell, Node and Server Names

In this window, complete the following information:

- Cell short name
Identifies the cell to z/OS facilities, such as SAF.
- Cell long name
Defines the primary external identification of this WebSphere Application Server for this z/OS cell. This name identifies the cell as displayed through the administrative console.
- Deployment manager short name
Specifies the name that identifies the node to z/OS facilities, such as SAF.
- Deployment manager long name
Identifies the primary external identification of this WebSphere Application Server for the z/OS node. This name identifies the node as displayed through the administrative console.
- Deployment manager server short name
Identifies the server to z/OS facilities, such as SAF. The server short name is also used as the server JOBNAME.
- Deployment manager server long name
Specifies the name of the application server and the primary external identification of this WebSphere Application Server for the z/OS server. This name identifies the server as displayed through the administrative console.
- Deployment manager cluster transition name
Defines the WLM application environment (WLM APPLENV) name for the deployment manager. If this is a server that is converted into a clustered server, this name

becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are of the same cluster.

Click **Next**.

12. The Configuration File System window (shown in Figure 5-15) requests configuration file system information for your z/OS system. The file system can be either HFS or zFS. It is used to hold WebSphere Application Server configuration information.

Profile Management Tool 8.5
Configuration File System
Management - deployment manager

Mount point:
/wasv85config/wpccell/wpdmnode

Directory path name relative to mount point:
DeploymentManager

Data set name:
OMVS.WAS85.WPCELL.WPDMNODE.ZFS

File system type
 Hierarchical File System (HFS)
 zSeries File System (ZFS)

Volume, or * for SMS:
TARHF1

Primary allocation in cylinders:
420

Secondary allocation in cylinders:
100

Figure 5-15 Configuration File System

In this window, complete the following information:

– Mount point

Specifies the read/write HFS directory where application data and environment files are written. The customization process creates this mount point if it does not already exist.

– The directory path name relative to the mount point

Defines the directory that is used for the deployment manager home directory.

– Data set name

Specifies the file system data set that you will create and mount at the specified mount point.

- File system type
Select the files system type to allocate and mount the configuration file system data set using as either HFS or zFS.
- Volume or “*” for SMS
Specify either the DASD volume serial number to contain the above data set or “*” to let SMS select a volume. Using “*” requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.
- Primary allocation in cylinders
Set the initial size allocation for the configuration file system data set. In the application server, the total space needed for this data set increases with the size and number of the installed applications. The minimum suggested size is 250 cylinders (3390).
- Secondary allocation in cylinders
Specifies the size of each secondary extent. The minimum suggested size is 100 cylinders.

Click **Next**.

13. Complete the information on the WebSphere Application Server Product File System window (shown in Figure 5-16), which defines the product file system directory and allows you to set up an intermediate symbolic link.

Best practice: Use intermediate symbolic links for flexibility when applying maintenance and running with different versions of the code.

After you complete this information, click **Next**.

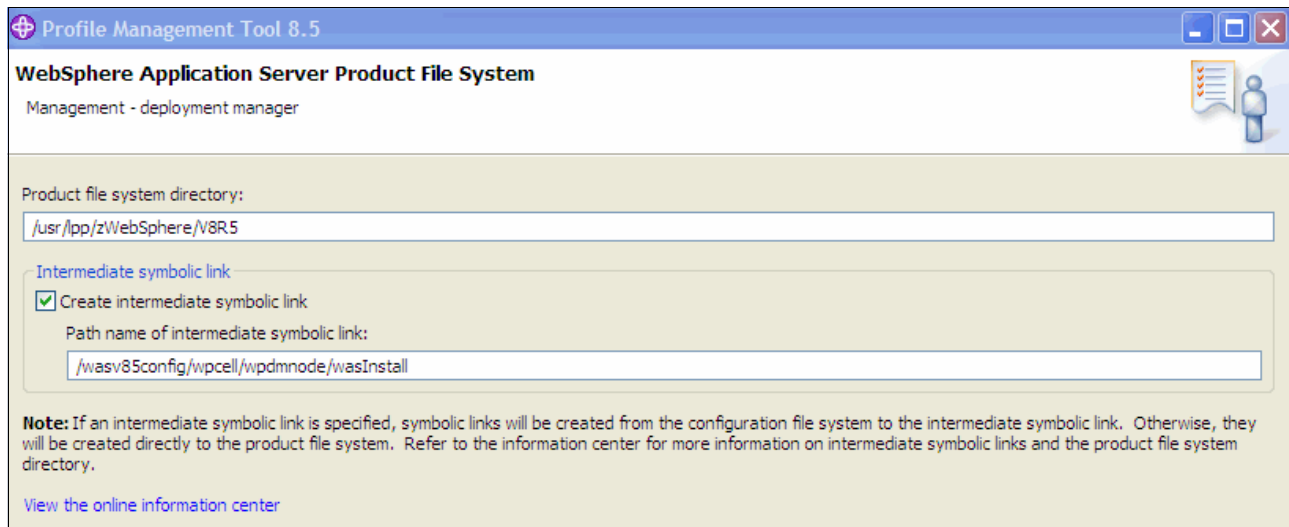


Figure 5-16 WebSphere Application Server Product File System

14. In the Process Definitions window (shown in Figure 5-17), enter the job names, procedure names, and user IDs to use for each process.

The screenshot shows a window titled "Profile Management Tool 8.5" with a subtitle "Management - deployment manager". The main heading is "Process Definitions". There are two sections for defining processes:

- Controller process:**
 - Job name: WPDMGR
 - Procedure name: WPDCR
- Servant process:**
 - Job name: WPDMGRS
 - Procedure name: WPDSR

Figure 5-17 Process Definitions

Enter the following information:

- Deploy manager controller process

The job name is specified in the IBM MVS™ START command JOBNAME parameter that is associated with the control region. This job name is the same job name as the server short name, and it cannot be changed during customization. The procedure name is the member name in your procedure library to start the control region. The user ID is the user ID that is associated with the control region.

- Deploy manager servant process

Specify the job name used by WLM to start the servant regions. This job name is set to the server short name followed by the letter S, and it cannot be changed during customization. The procedure name is the member name in your procedure library to start the servant regions. The user ID is the user ID that is associated with the servant regions.

After you complete this information, click **Next**.

15. The Port Values Assignment window (Figure 5-18) requires you to specify the ports to use for each process. Planning is important to avoid port conflicts, so be sure that you have all the values you need to complete this window.

| | |
|---|---------------------|
| Node host name or IP address (3): | wtsc58.itso.ibm.com |
| JMX SOAP connector port: | 12002 |
| Cell discovery port (6): | 12012 |
| ORB listener IP address (4): | * |
| ORB listener port: | 12003 |
| ORB SSL listener port (Z): | 12004 |
| HTTP transport IP address (5): | * |
| Administrative console port: | 12005 |
| Administrative console secure port: | 12006 |
| Administrative interprocess communication port (X): | 12009 |
| High availability manager communication port (DCS): | 12010 |
| DataPower appliance manager secure inbound port: | 12011 |
| Middleware agent RPC port: | 12013 |
| Administrative overlay UDP port (J): | 12014 |
| Administrative overlay TCP port (K): | 12015 |
| Status update listener port: | 12016 |

Figure 5-18 Port Values Assignment

After entering the required ports, click **Next**.

16. In the Location Service Daemon Definitions window (shown in Figure 5-19), enter the location service daemon settings. The location daemon service is the initial point of client contact in WebSphere Application Server for z/OS. The server contains the CORBA-based location service agent that places sessions in a cell. All RMI/IIOP IORs (for example, enterprise beans) establish connections to the location service daemon first, and then forward them to the target application server.

Profile Management Tool 8.5

Location Service Daemon Definitions

Management - deployment manager

Specify the location service daemon settings. The location service daemon is the initial point of client contact in WebSphere Application Server for z/OS. The server contains the CORBA-based location service agent, which places sessions in a cell. All RMI/IIOP IORs (for example, for enterprise beans) establish connections to the location service daemon first, then forward them to the target application server.

Daemon home directory:

Daemon job name:

Procedure name:

IP name:

Listen IP address:

Port:

SSL port:

Register daemon with WLM DNS

Figure 5-19 Location Service Daemon Definitions

In this window, enter the following information:

- Daemon home directory

The directory in which the location service daemon resides. This setting is set to the configuration file system mount point / daemon and cannot be changed.

- Daemon job name

Specifies the job name of the location service daemon, which is specified in the JOBNAME parameter of the MVS start command used to start the location service daemon. When configuring a new cell, be sure to choose a new daemon job name value. A server automatically starts the location service daemon if it is not already running.

- Procedure name

The member name in your procedure library to start the location service daemon.

- IP name

The fully-qualified IP name, registered with the Domain Name Server (DNS), that the location service daemon uses. The default is your node host name. In a sysplex, consider using a virtual IP address (VIPA) for the location service daemon IP name. Select the IP name for the location service daemon carefully. You can choose any name that you want, but, after being chosen, it is difficult to change, even in the middle of customization.

- Listen IP
The address at which the daemon listens. Select either * or a dotted IP address for this value.
- Port
Specify the port number on which the location service daemon listens.
- SSL port
The port number on which the location service daemon listens for SSL connections.

Important: Choose the IP name and port number carefully, because these names are difficult to change.

- Register daemon with WLM DNS
If you use the WLM DNS (connection optimization), you must register your location service daemon. Otherwise, do not register your location service daemon. Only one location service daemon per LPAR can register its domain name with WLM DNS. If you have multiple cells in the same LPAR and register more than one location service, it will fail to start.

Click **Next**.

17. In the SSL Customization window (shown in Figure 5-20), enter the SSL customization values.

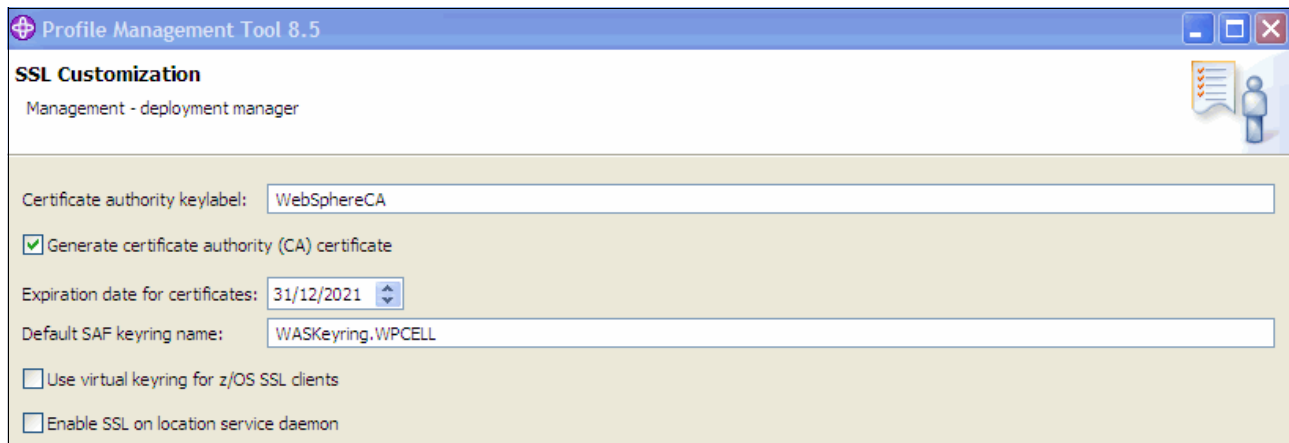


Figure 5-20 SSL Customization

Enter the following information:

- Certificate authority keylabel
The name that identifies the certificate authority (CA) to be used in generating server certificates.
- Generate certificate authority (CA) certificate
Select to generate a new CA certificate. Do not select this option to have an existing CA certificate generate server certificates.

- Expiration date for certificates
Used for any X509 Certificate Authority certificates created during customization and the expiration date for the personal certificates generated for WebSphere Application Server for z/OS servers. You must specify this even if you did not select **Generate Certificate Authority (CA) certificate**.
- Default SAF keyring name
The default name given to the RACF keyring used by WebSphere Application Server for z/OS. The keyring names created for repertoires are all the same within a cell.
- Use virtual keyring for z/OS SSL clients check box
Select this option if you want to enable the z/OS SSL client using SAF Virtual KeyRing to connect to this WebSphere Application Server node without requiring each user to have the WebSphere Application Server keyring or the WebSphere Application Server CA certificate connected to it.
- Enable SSL on the location service daemon check box
Select this option if you want to support secure communications using Inter-ORB Request Protocol (IIOP) to the location service daemon using SSL. If selected, a RACF keyring is generated for the location service daemon to use.

After completing the required SSL information, click **Next**.

18. On the next window, select the user registry to be used for administrative security. Choose from the following options:

- z/OS security product
This option uses the z/OS system's SAF-compliant security product, such as IBM RACF or equivalent, to manage WebSphere Application Server identities and authorization according to the following rules:
 - The SAF security database will be used as the WebSphere user repository.
 - SAF EJBROLE profiles will be used to control role-based authorization, including administrative authority.
 - Digital certificates will be stored in the SAF security database.

z/OS security note: Select the z/OS security product option if you plan to use the SAF security database as your WebSphere Application Server registry or if you plan to set up an LDAP or custom user registry whose identities will be mapped to SAF user IDs for authorization checking. For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

- WebSphere Application Server security
The WebSphere Application Server administrative security option is used to manage the Application Server identities and authorization according to these rules:
 - A simple file-based user registry will be built as part of the customization process.
 - Application-specific role binds will be used to control role-based authorization.
 - The WebSphere Application Server console users and groups list will control administrative authority.
 - Digital certificates will be stored in the configuration file system as keystores.

Digital certificates note: Choose this option if you plan to use an LDAP or custom user registry without mapping to SAF user IDs. (The file-based user registry is not a best practice for production use.)

- No security

Although it is not a best practice, you can disable administrative security. If you choose this security option, there are no other choices to make. Your WebSphere Application Server environment will not be secured until you configure and enable security manually. You can enable security manually later using the administrative console or using Jython scripts.

For our scenario, we used the “Use z/OS security product” option.

Click **Next**.

19. Enter the parameters shown in Figure 5-21 after you select the z/OS security product option.

The screenshot shows the 'Profile Management Tool 8.5' window. The title bar reads 'Profile Management Tool 8.5'. The main content area is titled 'Security Managed by the z/OS Product' and has a subtitle 'Management - deployment manager'. Below the title, there is a text input field for 'SAF profile prefix (optional):' containing the text 'WPCELL'. Underneath, it says 'WebSphere Application Server unauthenticated user' followed by a 'User ID:' label and a text input field containing 'WPGUEST'. A section titled 'UID' contains two radio button options: 'Allow OS security to assign UID' (which is selected) and 'Assign user-specified UID:' followed by a dropdown menu showing the value '2402'. At the bottom left, there is a checked checkbox labeled 'Enable writable SAF keyring support'.

Figure 5-21 Security Managed by the z/OS Product

Complete the following information:

- (Optional) SAF profile prefix (formally known as the *Security domain identifier*)

This optional parameter is used to distinguish between APPL or EJBROLE profiles based on the security domain name. It provides an alphanumeric security domain name of one to eight characters. Internally, this sets SecurityDomainType to the string cellQualified.

All servers in the cell prepend the security domain name that you specify to the application-specific J2EE role name to create the SAF EJBROLE profile for checking. The security domain name is not used, however, if role checking is performed using WebSphere Application Server for z/OS bindings.

The security domain name is also used as the APPL profile name and inserted into the profile name used for CBIND checks. The RACF jobs that the Customization Dialog generates create and authorize the appropriate RACF profiles for the created nodes and servers.

If you do not want to use a security domain identifier, leave this field blank.

- WebSphere Application Server unauthenticated user ID
Associated with unauthenticated client requests. It is sometimes referred to as the “guest” user ID. Give it the RESTRICTED attribute in RACF to prevent it from inheriting UACC-based access privileges. The UNIX System Services UID number for the user ID is specified here and is associated with unauthenticated client requests. The UID value must be unique numeric values between 1 and 2,147,483,647.
- Enable Writable SAF Keyring support
This feature allows the administrative console to create and sign certificates by a CA, connect to keyrings, remove from keyrings, and import, export, and renew.
All certificates created with the writable keyring support are generated and signed by Java code and not by SAF. In this case, the writable keyring support only uses SAF to store the generated certificates.
The writable keyring support is completely optional. New keystores and truststores marked as read-only can be created independently from the writable keyring support. When using the read-only JCERACFKS and JCECCARACFS keystores, the certificates in the appropriate SAF keyring can still be viewed in the administrative console.

Click **Next**.

20. On the next window, you tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process updates the job name for you in all the generated jobs, so do not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.
21. The last window shows a short summary of the customization, including profile type and where the generated jobs will be stored. To change the characteristics of this profile, click **Back**. Otherwise, click **Create** to generate your z/OS customization jobs.
22. The Profile Management tool displays a summary window that indicates whether the jobs were created successfully or not. If the jobs were not created, a log file containing failure information is identified. Click **Finish** to return to the Profile Management tool main window. The new deployment manager definition is listed in the Customization Definitions tab.

Uploading the jobs to the z/OS system

Next, upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile, and click **Process**. To upload the generated jobs to the target z/OS system, select the desired option. The available options are:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file system

Click **Next**.

2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port.

Select the **Allocate target z/OS data sets** option to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish**. A progress bar displays while the upload occurs.

Executing the jobs

After the customization profile is uploaded, the next step is to execute the jobs. The instructions for preparing for and executing the jobs are in the Profile Management Tool. Select the customization definition, and click the **Customization Instructions** tab (Figure 5-22). These instructions are also contained in a job that was loaded to the host.

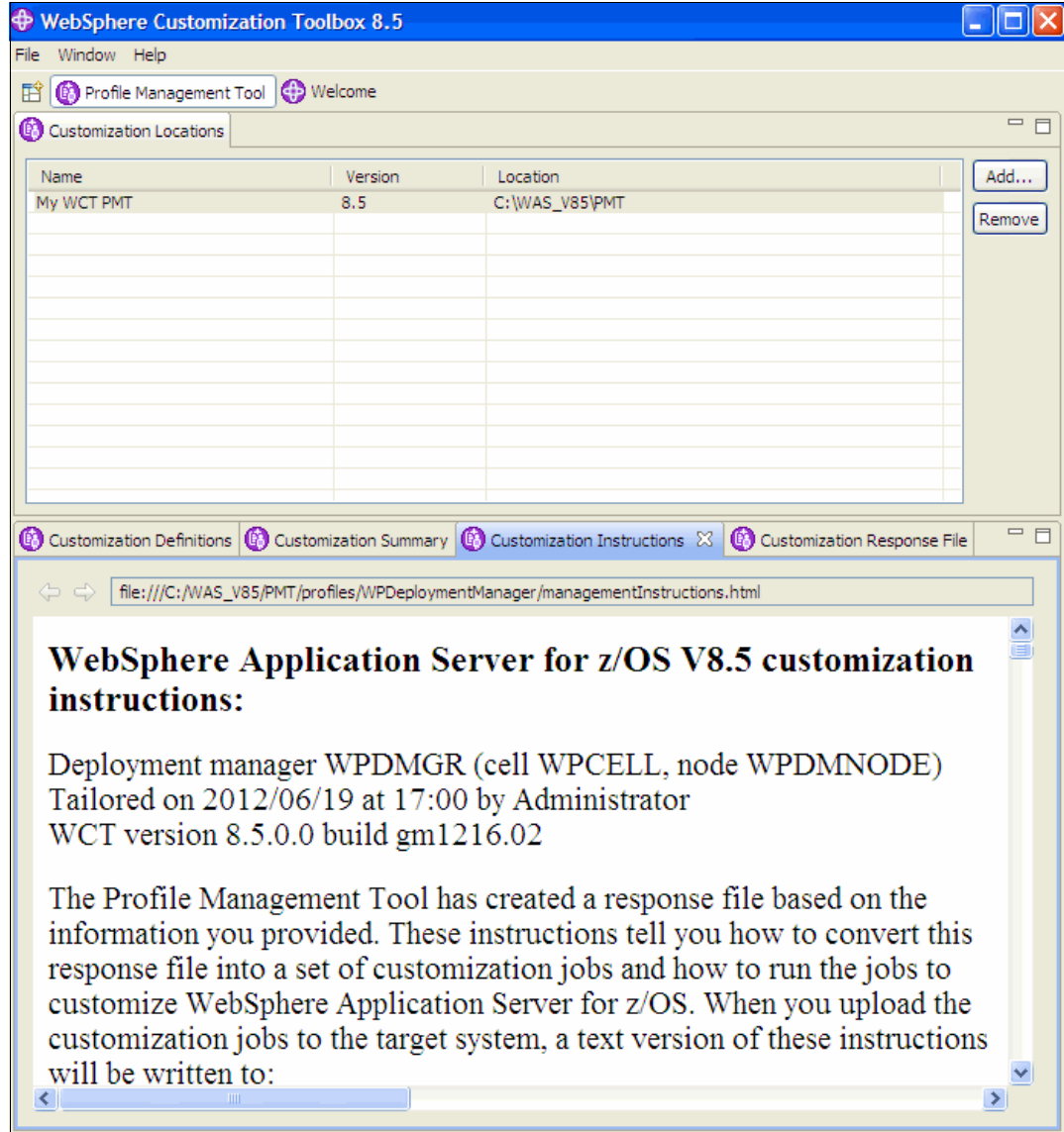


Figure 5-22 Customization definition instructions

The instructions help you determine what jobs to run, the order to run them in, and the expected results. They also tell you how to start the environment after you are done.

After the jobs run successfully, the deployment manager profile is complete.

Run the following jobs:

- BBOSBRAK** Creates common groups and user IDs.
- BBOSBRAM** Creates home directories to user IDs.
- BBODBRAK** Creates users and profiles required by WebSphere node.
- BBODCFCS** Creates the mount point directory, allocates the file system, and mounts it.
- BBODHFSA** Populates the configuration file system and prepares it for profile creation.
- BBOWWPFDF** Creates the profile in the configuration file system.
- BBODPROC** Creates the procedures and copies them to the proper library.

A configuration has three basic components: a file system with configuration XML, JCL start procedures, and SAF profiles. Figure 5-23 shows how the jobs are mapped to the components of the configuration.

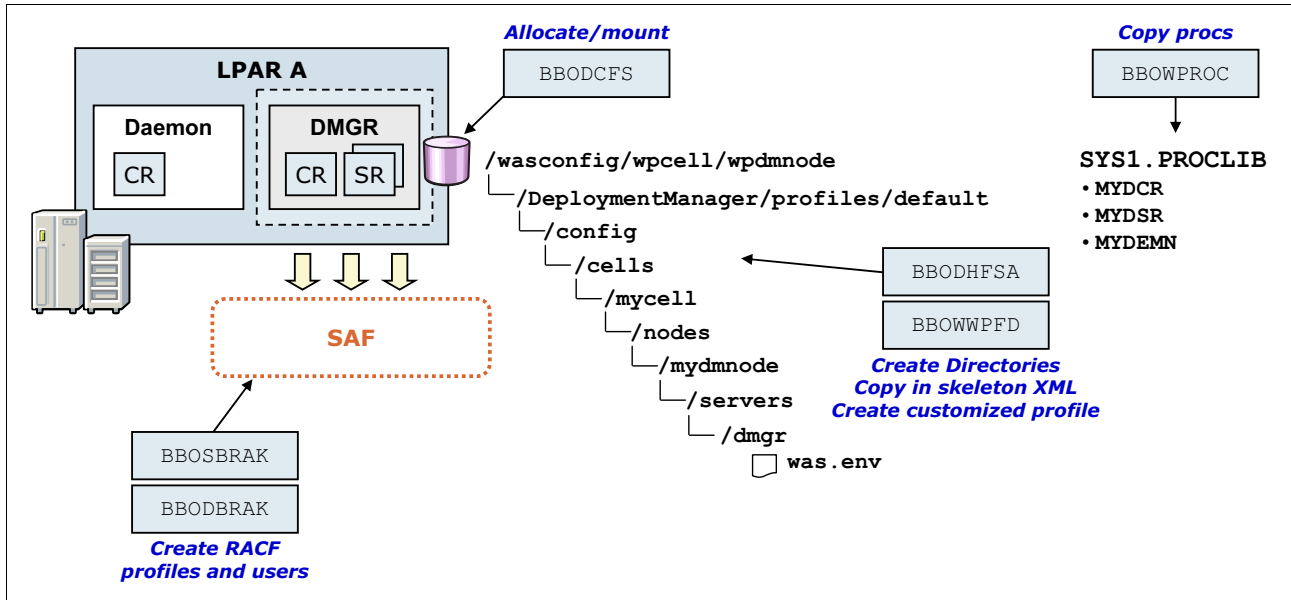


Figure 5-23 Jobs mapped to configuration components

5.3.2 Creating the base application server definition

Next, create the base application server definition. Using the Profile Management tool, complete the following steps:

1. Click **Create**.
2. In the Environment Selection window (shown in Figure 5-24), click **Application server**, and click **Next**.

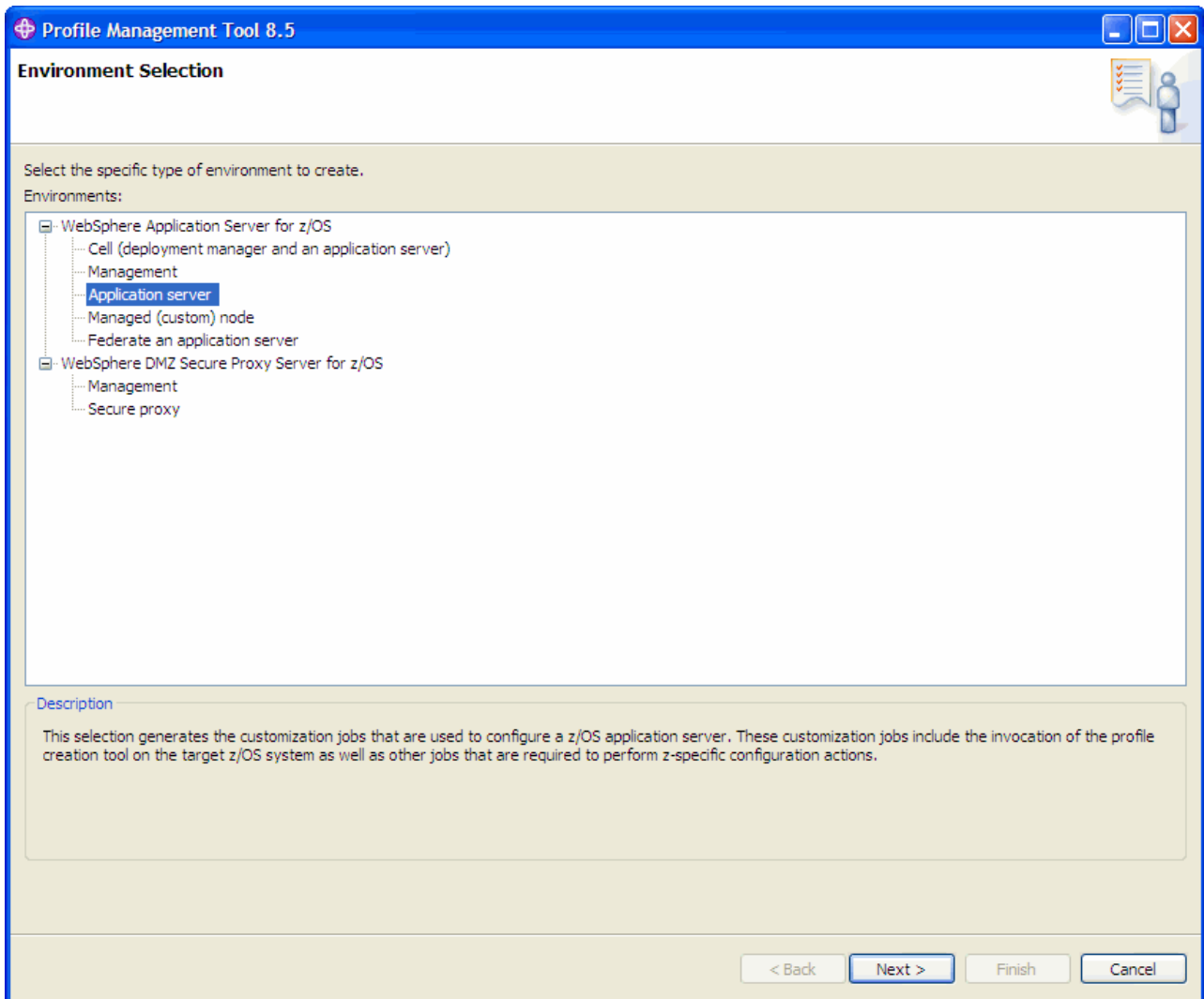


Figure 5-24 Environment Selection

3. Specify a name for the customization definition and, optionally, a response file path. The server runtime performance tuning setup contains the following selectable options:
 - Standard: Performance options set to general purpose server
 - Peak: Performance options set to environments where application updates are not often

We used the following values:

- Customization definition name: ZAppSrv01
- Server runtime performance running setting: Standard

Click **Next**.

4. In the Default Values window (shown in Figure 5-25), set the configuration default values.

Profile Management Tool 8.5

Default Values

Application server

Specify options for generating default values for this customization definition. The default values that are generated are similar to those generated by the *Configuration Planning Spreadsheet*. Refer to the information center for information on the *Configuration Planning Spreadsheet*.
[View the online information center](#)

GID and UID defaults

Set each default GID and UID value to indicate OS security is to assign an unused value.
Note: For details about the security set-up that is required for this option, refer to the following information center topic.
[View the online information center](#)

Name and userid defaults

Set default names and userids based on cell, cluster and system identifiers.

Application server will be federated into a Network Deployment cell

Two-character cell identifier: (for the Network Deployment cell into which this node will be federated)

Two-character cluster identifier:

Single-character system identifier:

Port defaults

Select default port values from the following port range.

Lowest default port number:

Highest default port number:

The port range must specify at least 20 ports.

< Back Next > Finish Cancel

Figure 5-25 Application server default values

Important: If you specified a response file for setting default values, any default that you select here overrides the corresponding response file values.

The GID and UID defaults section contains the following selectable option:

- Set each default GID and UID value to indicate that operating-system security is to assign an unused value:

When this option is selected, each GID and UID value defaults to allow operating system security to assign an unused value. When this option is not selected, each GID and UID value defaults to an IBM-provided number.

The Name and user ID defaults contain the following selectable options:

- Set default names and user IDs based on cell, cluster, and system identifiers
When this option is selected, default cell, node, server, cluster, procedure names, group names, and user IDs are based on cell, cluster, and system identifiers.
- Application server will be federated into a Network Deployment cell
Select this option to indicate that the application server will be federated into a Network Deployment cell at some point in time.
- Two-character cell identifier
Enter a two-character cell identifier to be used to create default names and user IDs.
If you have selected the option to federate to a cell, specify the two-character cell identifier of the target Network Deployment cell.
The first character must be an alphabetic character and the second character must be an alphanumeric character. Alphabetic characters can be entered in lowercase or uppercase. The case of alphabetic characters are adjusted as appropriate for each generated default value.
- Two-character cluster identifier
The two-character cluster identifier to be used to create default names and user IDs. The characters will be appended to the cluster transition name.
The characters must be alphabetic characters. The alphabetic characters can be entered in lowercase or uppercase. The case of alphabetic characters are adjusted as appropriate for each generated default value.
- Single-character system identifier
The single-character system identifier to be used to create default names and user IDs. It will be appended to the short and long names for the cell, node, and server and to the appropriate process names.
The character must be an alphanumeric character. An alphabetic character can be entered in lowercase or uppercase. The case of the alphabetic character will be adjusted as appropriate for each generated default value.
- Port defaults
Select default port values from the following port range.
When this option is not selected, each port value defaults to an IBM-provided number. When this option is selected, each port default value is selected from the following port number range.
The port range must contain at least 20 ports.
- Lowest default port number
The lowest number that can be assigned as a default port number.
- Highest default port number
The highest number that can be assigned as a default port number.

Click **Next**.

5. Next, specify the high-level qualifier for the target z/OS data sets that will contain the generated jobs and instructions that are created (refer to Figure 5-10 on page 131).

Note: You can specify a multilevel high-level qualifier as the data set high-level qualifier.

The generated batch jobs and instructions are uploaded to the following z/OS partitioned data sets:

- *HLQ.CNTL*

A partitioned data set with fixed block 80-byte records to contain customization jobs

- *HLQ.DATA*

A partitioned data set with variable-length data to contain other data contained in the customization definition

We used the WAS85.WPASND high-level qualifier (refer to Figure 5-10 on page 131).

Click **Next**.

6. Complete the information in the Configure Common Groups window, as shown in Figure 5-26.

The screenshot shows the 'Configure Common Groups' dialog box in the Profile Management Tool 8.5. The dialog is titled 'Configure Common Groups' and has a subtitle 'Application server'. It is divided into three sections for configuring different groups:

- WebSphere Application Server configuration group information:** The 'Group' field contains 'WPCFG'. Under the 'GID' section, the radio button 'Allow OS security to assign GID' is selected.
- WebSphere Application Server servant group information:** The 'Group' field contains 'WPSRVG'. Under the 'GID' section, the radio button 'Allow OS security to assign GID' is selected.
- WebSphere Application Server local user group information:** The 'Group' field contains 'WPGUESTG'. Under the 'GID' section, the radio button 'Allow OS security to assign GID' is selected.

At the bottom of the dialog, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

Figure 5-26 Configure Common Groups

Complete the following information:

- WebSphere Application Server configuration group information

Specify the default group name for the WebSphere Application Server administrator user ID and all server user IDs.

Select whether to allow the OS security system (RACF) to assign an unused GID value or to assign a specific GID.

– WebSphere Application Server servant group information

Specify the group name for all servant user IDs. You can use this group to assign subsystem permissions, such as DB2 authorizations, to all servants in the security domain.

Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.

– WebSphere Application Server local user group information

Specify the group name for local clients and unauthorized user IDs (provides minimal access to the cell).

Select whether to allow the OS security system (RACF) to assign an unused GID value or assign a specific GID.

GID values: The specified GID is the UNIX System Services GID number for the WebSphere Application Server configuration group. GID values must be unique numeric values between 1 and 2,147,483,647.

Click **Next**.

7. In the Configure Common Users window, provide information about the asynchronous administration user ID, as shown in Figure 5-27 on page 153.

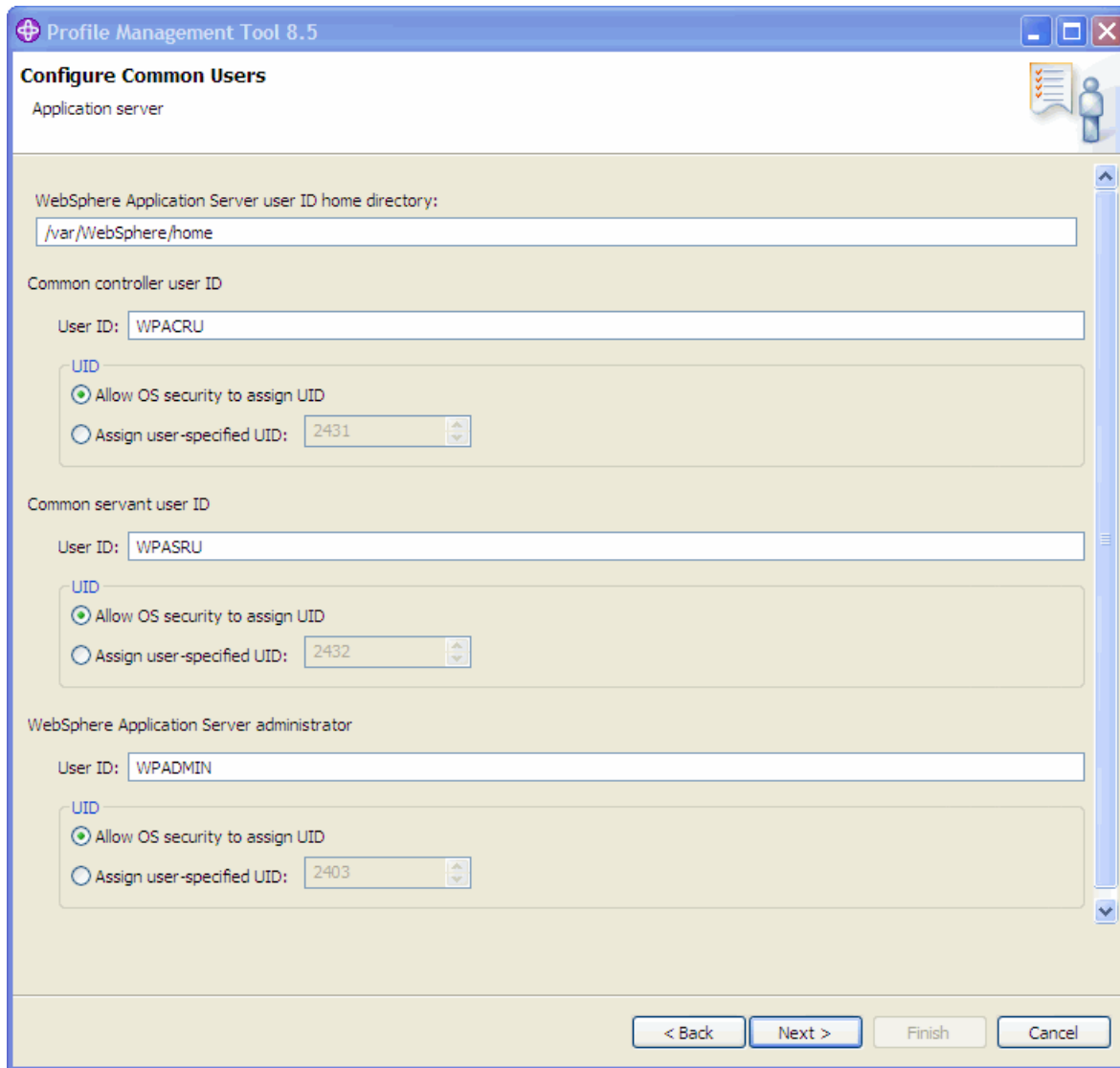


Figure 5-27 Configure Common Users - Asynchronous administration user ID

Provide the following information:

- WebSphere Application Server user ID home directory

This field identifies a new or existing file system directory in which home directories for WebSphere Application Server for z/OS user IDs are created by the customization process. This directory does not need to be shared among z/OS systems in a WebSphere Application Server cell.

- Common controller user ID

UIDs: UIDs must be unique numbers between 1 and 2,147,483,647 within the system.

Enter the user ID to be associated with all of the control regions and the daemon. This user ID will also own all of the configuration file systems. If you are using a non-IBM security system, the user ID might have to match the procedure name. Refer to your security system's documentation.

Select whether to allow the OS security system (RACF) to assign an unused UID value, or assign a specific UID to be associated with the control region user ID.

– Common servant user ID

Enter the user ID to be associated with the servant and control adjunct regions.

Select whether to allow the OS security system (RACF) to assign an unused UID value, or assign a specific UID to be associated with the servant region user ID.

– WebSphere Application Server administrator

Enter the user ID for the initial WebSphere Application Server administrator. The user ID must have the WebSphere Application Server configuration group as its default UNIX System Services group.

Select whether to allow the OS security system (RACF) to assign an unused UID value, or assign a specific UID to be associated with the administrator user ID.

We used the following values:

- Common controller user ID: WPACRU
- UID: Allow OS security to assign UID
- Common servant user ID: WPASRU
- UID: Allow OS security to assign UID
- WebSphere Application Server administrator: WPADMIN
- UID: Allow OS security to assign UID
- Asynchronous administration user ID: WPADMSH
- UID: Allow OS security to assign UID
- WebSphere Application Server user ID home directory: /var/WebSphere/home

Click **Next**.

8. Provide the system and data set names to be used (See Figure 5-13 on page 135):

- Specify the system and sysplex name for the target z/OS system on which you will configure WebSphere Application Server for z/OS.
- Enter the name of an existing procedure library where the WebSphere Application Server for z/OS cataloged procedures are added.

We used the following values:

- System name: SC58
- Sysplex name: PLEX58
- PROCLIB data set name: SYS1.PROCLIB

Click **Next**.

9. In the Cell, Node and Server names window, provide the necessary information, as shown in Figure 5-28.

Profile Management Tool 8.5

Cell, Node and Server Names

Application server

Cell names

Short name: WPBASEA

Long name: wpbasea

Node names

Short name: WPNODEA

Long name: wpnodea

Server names

Short name: WPSR00A

Long name: wpsr00a

Cluster transition name:

WPSR00

< Back Next > Finish Cancel

Figure 5-28 Cell, Node and Server Names

Provide the following information:

- Specify the long and short names for the cell, node, and servers. Short names identify the process to z/OS facilities, such as SAF. Long names are used as the primary external identification for the process. This is the name you will see in the administrative console.
- Cluster transition name

If this server is converted into a clustered server, this name becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are part of the same cluster.

Click **Next**.

10. Enter the Configuration File System values, as shown in Figure 5-16 on page 138.

- Mount point

Application server configuration file system mount point: Specifies the Read/write file system directory where the application data and environment files are written. This field is not writable here, but was specified earlier on the System Environment: Configuration file system information window.

- Directory path name relative to mount point

The relative path name of the directory within the configuration file system in which the application server configuration resides.

- Data set name
The file system data set that you will create and mount at the specified mount point. File system type
Select to allocate and mount your configuration file system data set using HFS or zFS.
- Volume, or “*” for SMS
Specify either the DASD volume serial number to contain the above data set or “*” to let SMS select a volume. Using “*” requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.
- Primary allocation in cylinders
The initial size allocation for the configuration file system data set. In the application server, the total space needed for this data set increases with the size and number of the installed applications. The minimum suggested size is 250 cylinders (3390).
- Secondary allocation in cylinders
The size of each secondary extent. The minimum suggested size is 100 cylinders.

We used the following values:

- Mount point: /wasv85config/wpcell/wpdmnodea
- Directory path name relative to mount point: AppServer
- Data set name: OMVS.WAS85.WPCELL.WPNODEA.ZFS
- File system type: IBM zSeries® File System (zFS)
- Volume, or “*” for SMS: *
- Primary allocation in cylinders: 420
- Secondary allocation in cylinders: 100

Click **Next**.

11. Specify the information for the product file system (see Figure 5-16 on page 138).

- Specify the name of the directory where the product files for WebSphere Application Server for z/OS were stored during installation.
- Select the option to allow to set up an intermediate symbolic link and specify the path name.

We used the following values:

- Product file system directory: /usr/lpp/zWebSphere/V8R5
- Intermediate symbolic link: Create intermediate symbolic link
- Path name of intermediate symbolic link: /wasv85config/wpcell/wpnodea/wasInstall

Click **Next**.

12. In the next window, select the applications to deploy to the environment that you are creating:

- Administrative console (best practice)
- Default application

We enabled both options. Click **Next**.

13. Enter the process information into the window shown in Figure 5-29. The job names for the processes are provided in the window and cannot be changed. Specify the procedure name for each process.

Profile Management Tool 8.5

Process Definitions

Application server

Controller process

Job name: WPSR00A

Procedure name: WPACRA

Controller adjunct process

Job name: WPSR00AA

Procedure name: WPAARA

Servant process

Job name: WPSR00AS

Procedure name: WPASRA

< Back Next > Finish Cancel

Figure 5-29 Application server names

Enter the following information:

- Controller process job and procedure name
The job name for the control region is the same as the server short name. This is the name used in the MVS START command to start the region.
- Controller adjunct process job and procedure name
The job name is used by WLM to start the control region adjunct. This is set to the server short name followed by the letter “A.”
- Servant process job and procedure name
The job name is used by WLM to start the servant regions. This is set to the server short name followed by the letter “S.”

Click **Next**.

14. Specify the application server port values in the window shown in Figure 5-30.

Good planning is important to avoid port conflicts. Ensure that you have all of the values that you need to complete the information in this window.

Click **Next**.

The screenshot shows the 'Port Values Assignment' window in the Profile Management Tool 8.5. The window title is 'Profile Management Tool 8.5' and the subtitle is 'Port Values Assignment'. Below the subtitle, it says 'Application server'. The window contains a list of configuration items, each with a text input field and a spin button. The values are as follows:

| Configuration Item | Value |
|--|---------------------|
| Node host name or IP address (3): | wtsc58.itso.ibm.com |
| JMX SOAP connector port: | 12002 |
| ORB listener IP address (4): | * |
| ORB listener port: | 12003 |
| ORB SSL listener port (Z): | 12004 |
| HTTP transport IP address (5): | * |
| Administrative console port: | 12005 |
| Administrative console secure port: | 12006 |
| HTTP transport port: | 12007 |
| HTTPS transport port: | 12008 |
| Administrative interprocess communication port (X): | 12009 |
| High availability manager communication port (DCS): | 12010 |
| Service integration port: | 12011 |
| Service integration secure port: | 12012 |
| Service integration MQ interoperability port: | 12013 |
| Service integration MQ interoperability secure port: | 12014 |
| Session Initiation Protocol (SIP) port: | 12015 |

At the bottom of the window, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

Figure 5-30 Application server port values

15. Enter the following information about the Location Service Daemon Definitions, as shown in Figure 5-19 on page 141:

- Daemon home directory

Directory in which the location service daemon resides. This is set to the configuration file system mount point / daemon and cannot be changed.

- Daemon job name

Specifies the job name of the location service daemon, specified in the JOBNAME parameter of the MVS start command used to start the location service daemon.

- Procedure name

Name of the member in your procedure library to start the location service daemon.

- IP Name
The fully qualified IP name, registered with the Domain Name Server (DNS), that the location service daemon uses.
- Listen IP
Address at which the daemon listens.
- Port
Port number on which the location service daemon listens.
- SSL port
Port number on which the location service daemon listens for SSL connections.
- Register daemon with WLM DNS
If you use the WLM DNS (connection optimization), you must select this option to register your location service daemon with it; otherwise, do not select it.

We used the following values:

- Daemon home directory: /wasv85config/wpce11/wpnodea/Daemon
- Daemon job name: WPDEMNA
- Procedure name: WPDEMNA
- IP name: wtsc58.itso.ibm.com
- Listen IP address: *
- Port: 12060
- SSL Port: 12061
- Register daemon with WLM DNS: not enabled

Click **Next**.

16. Enter the information required for SSL connections.

We used the following values:

- Certificate authority keylabel: WebSphereCA
- Generate certificate authority (CA) certificate: enabled
- Expiration date for certificates: 2021/12/31
- Default SAF keyring name: WASKeyring.WPCELL
- Use virtual keyring for z/OS SSL clients: Not enabled
- Enable SSL on location service daemon: Not enabled

Click **Next**.

17. Select the user registry that will be used to manage user identities and authorization policy.

Tip: If you plan to federate this application server, set the application server's SAF profile prefix to be the same as that of the Network Deployment Cell.

Select from one of the following choices:

- z/OS security product
This option uses the z/OS system's SAF-compliant security product, such as IBM RACF or equivalent, to manage WebSphere Application Server identities and authorization according to the following rules:
 - The SAF security database is used as the WebSphere user repository.
 - SAF EJBROLE profiles will be used to control role-based authorization, including administrative authority.

- Digital certificates are stored in the SAF security database.

Important: Select the z/OS security product option if you plan to use the SAF security database as your WebSphere Application Server registry or if you plan to set up an LDAP or custom user registry whose identities will be mapped to SAF user IDs for authorization checking. For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

– WebSphere Application Server security

The WebSphere Application Server administrative security option is used to manage the Application Server identities and authorization as follows:

- A simple file-based user registry is built as part of the customization process.
- Application-specific role binds will be used to control role-based authorization.
- The WebSphere Application Server console users and groups list will control administrative authority.
- Digital certificates are stored in the configuration file system as keystores.

Tip: Choose this option if you plan to use an LDAP or custom user registry without mapping to SAF user IDs. (The file-based user registry is not recommended for production use.)

– No security

Although it is not a best practice, you can disable administrative security. If you choose this security option, there are no other choices to make. Your WebSphere Application Server environment will not be secured until you configure and enable security manually. You can enable security manually later using the administrative console or using Jython scripts.

We select the **Use z/OS security product** option.

Click **Next**.

18. In the next window, choose one of the following options:

– SAF profile prefix (formally known as *Security domain identifier*)

This optional parameter is used to distinguish between APPL or EJBROLE profiles based on security domain name. It provides an alphanumeric security domain name of one to eight characters. Internally, this sets SecurityDomainType to the string cellQualified.

All servers in the cell prepend the security domain name you specify to the application-specific J2EE role name to create the SAF EJBROLE profile for checking. The security domain name is not used, however, if role checking is performed using WebSphere Application Server for z/OS bindings.

The security domain name is also used as the APPL profile name and inserted into the profile name used for CBIND checks. The RACF jobs that the Customization Dialog generates create and authorize the appropriate RACF profiles for the created nodes and servers.

If you do not want to use a security domain identifier, leave this field blank.

- WebSphere Application Server unauthenticated user ID
Associated with unauthenticated client requests. It is sometimes referred to as the “guest” user ID. Give it the RESTRICTED attribute in RACF to prevent it from inheriting UACC-based access privileges. The UNIX System Services UID number for the user ID is specified here and is associated with unauthenticated client requests. The UID value must be unique numeric values between 1 and 2,147,483,647.
- Enable Writable SAF Keyring support
This feature allows the administrative console to create and sign certificates by a CA, connect to keyrings, remove from keyrings, and import, export, and renew.
All certificates created with the writable keyring support are generated and signed by Java code and not by SAF. In this case, the writable keyring support only uses SAF to store the generated certificates.
The writable keyring support is completely optional. New keystores and truststores marked as read-only can be created independently from the writable keyring support. When using the read-only JCERACFKS and JCECCARACFS keystores, the certificates in the appropriate SAF keyring can still be viewed in the administrative console.

We used the following values:

- SAF profile prefix (optional): WPCELL
- WebSphere Application Server unauthenticated user ID: WPGUEST
- UID: Allow OS security to assign UID
- Enable Writable SAF Keyring support: enabled

Click **Next**.

19. In the next window, you create web server definitions; however, they are not needed at this time. Click **Next**.
20. In the next window, tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process updates the job name for you in all of the generated jobs, so do not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.
21. The Customization Summary is the final window. Click **Create** to store this application server environment definition for later transfer to the intended z/OS host system.
22. Click **Finish** when the jobs are complete.

5.3.3 Uploading jobs and associated instructions

If successful, the next step in the z/OS customization process is to upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile, and click **Process**. To upload the generated jobs to the target z/OS system, select from the following options:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file system
Click **Next**.
2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path name must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port.

Select the **Allocate target z/OS data sets** option to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish**, and a progress bar displays while the upload is occurring.

3. After the customization profile is uploaded, select the customization definition in the Profile Management Tool, and click the **Customization Instructions** tab. This tab provides complete instructions about how to build the profile using the jobs.

These instructions can help you determine the jobs to run, the order to run them in, and the expected results. It also explains how to start the environment after you are finished.

After the jobs run successfully, the application server profile is complete.

5.3.4 Federating an application server

This definition is used to federate the base application server into the previously created deployment manager node. To begin, run the Profile Management tool to create the customization definition:

1. On the Profile Management Tool main window, click **Create**. In the Environment Selection window, click **Federate an application server**, as shown in Figure 5-31. Click **Next**.

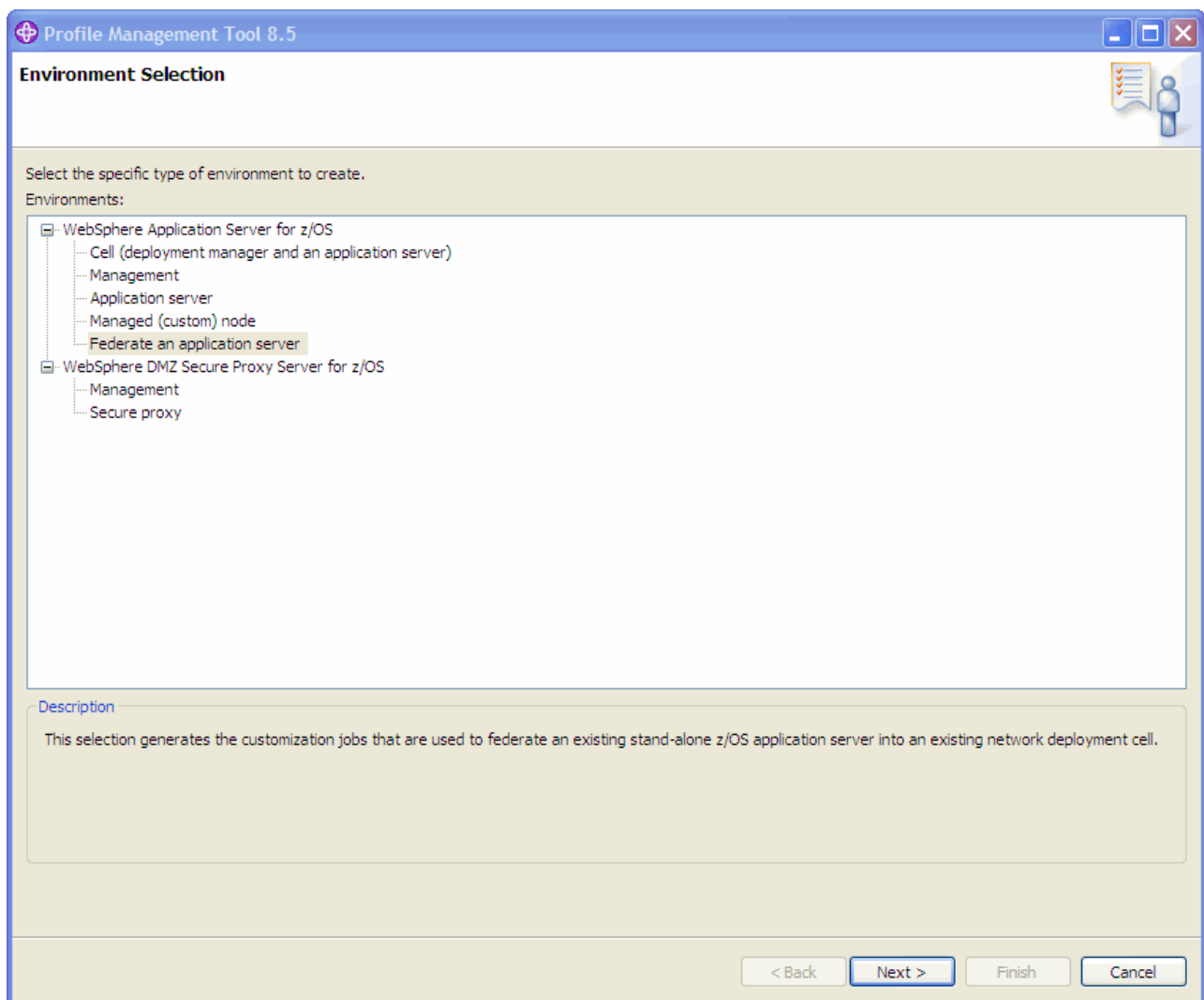


Figure 5-31 Federate an application server selection

2. Specify a name for the customization definition and, optionally, a response file path. We used ZFederate01 as the name. Click **Next**.
3. In the Default Values window, specify a default range of ports to be used for the node agent, as shown in Figure 5-32. When this option is not selected, each port value defaults to an IBM-provided number. Click **Next**.

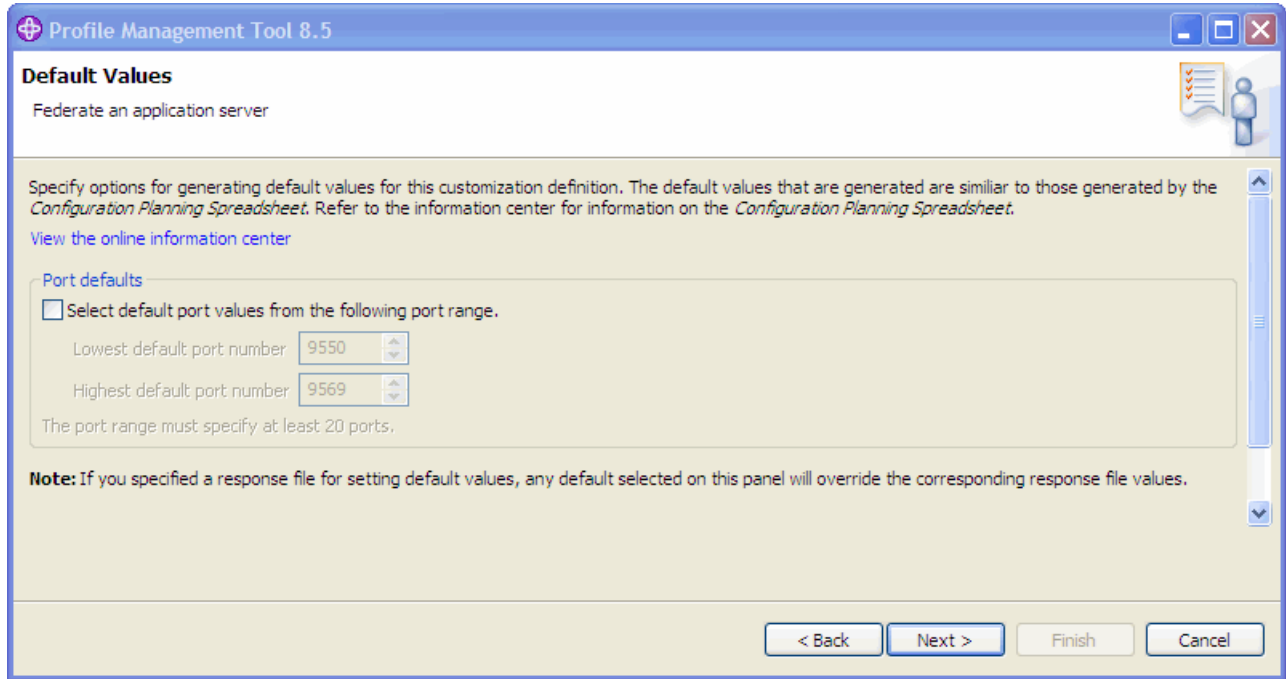


Figure 5-32 Default port settings

4. Specify the high-level qualifier for the target z/OS data sets that will contain the generated jobs and instructions that are created (see Figure 5-10 on page 131).

The generated batch jobs and instructions are uploaded to the following z/OS partitioned data sets:

- *HLQ.CNTL*

Partitioned data set with fixed block 80-byte records to contain customization jobs

- *HLQ.DATA*

Partitioned data set with variable-length data to contain other data contained in the customization definition (Figure 5-33 on page 164)

Tip: You can specify a multi-level high-level qualifier as the data set high-level qualifier.

- We used the *WAS85.WPFED* high-level qualifier.

Click **Next**.

5. In the Federate Application Server (Part 1) window, specify the information that is needed to access the application server and the deployment manager, as shown in Figure 5-33.

Figure 5-33 Specify Application server settings (Part 1)

Enter the following information:

- Application server access information

This section contains the information needed to find and access the application server node configuration file system. This information includes the file system mount point, the directory path name, and, if administrative security is enabled on the application server, the administrator user ID and password.

- Deployment manager access information

Federating the application server requires that the deployment manager be running and accessible by the federation process. This section provides the information required to connect to the deployment manager, including the host name or IP address, JMX connection type and port, and the administrator user ID and password.

The node host name must always resolve to an IP stack on the system where the deployment manager runs. The node host name cannot be a DVIPA or a DNS name that, in any other way, causes the direction of requests to more than one system.

The user ID and password are required when global security is enabled on the Network Deployment cell unless an RMI connector is being used. If an RMI connector is being used, the identity information is extracted from the thread of execution of the addNode job if the user ID and password are not specified.

Click **Next**.

6. Configure the node agent that is required for the application server to participate in the distributed environment cell, as shown in Figure 5-34.

Profile Management Tool 8.5

Federate Application Server (Part 2)

Federate an application server

New ORB port for application server: 12063

Node group name: DefaultNodeGroup

Configuration group name: WPCFG

Configuration user ID: WSADMIN

Include applications

Launch the node agent after node federation

Federate service integration buses that exist on this node

Node agent definitions

Server name (short): BBON001

Server name (long): nodeagent

JMX SOAP connector port: 8878

ORB listener IP address (4): *

ORB port: 2810

ORB SSL port (Z): 0

Node discovery port: 7272

Node multicast discovery port: 5000

Node IPv6 multicast discovery port: 5001

Node agent interprocess communication port (X): 9626

High availability manager communication port (DCS): 9354

Middleware agent RPC port: 7061

< Back Next > Finish Cancel

Figure 5-34 Specify Application server settings (Part 2)

Enter the following information:

- Specify the short name for the node agent process.
The short name is the server's job name, as specified in the MVS START command JOBNAME parameter. (The node agent server long name is set to the fixed value of nodeagent.)
- Specify the IP addresses and ports to be used by the node agent and a new ORB port to be used by the application server.
- The node group, configuration group, and configuration user ID.
- Select the relevant federation options for your environment. If you installed applications on the application server or defined service integration buses, you can choose to have those included in the newly federated application server.

Click **Next**.

7. On the next window, tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process updates the job name for you in all the generated jobs, so do not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.
8. The Customization Summary window is the final window. Click **Create**, and then click **Finish**.

5.3.5 Uploading jobs and associated instructions

If successful, the next step in the z/OS customization process is to upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile, and click **Process**. To upload the generated jobs to the target z/OS system, select from the following options:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file system

Click **Next**.

2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port.

Select the **Allocate target z/OS data sets** option to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish**, and a progress bar displays while the upload occurs.

3. After the customization profile is uploaded, select the customization definition in the Profile Management Tool, and click the **Customization Instructions** tab. This tab provides complete instructions about how to build the profile using the jobs.

These instructions help you to determine the jobs to run, the order in which to run them, and the expected results. It also explains how to start the environment after you are finished.

After the jobs run successfully, the application server profile is complete.

Attention: The user that runs job BBOWADDN must have READ access on profiles IRR.DIGTCERT.LIST and IRR.DIGTCERT.LISTRING.

5.4 Creating a job manager profile

In a flexible management environment, the job manager allows you to asynchronously submit and administer jobs for large numbers of unfederated application servers and deployment managers over a geographically dispersed area. Many of the management tasks that you can perform with the job manager are tasks that you can already perform with the product, such as application management, server management, and node management. However, with the job manager, you can aggregate the tasks and perform the tasks across multiple application servers or deployment managers.

5.4.1 Creating the customization definition

To begin, run the profile management tool to create the customization definition:

1. Click **Create** on the Profile Management Tool main window.
2. To generate the definitions for the job manager, select **Management** in the Profile Management tool environment selection window, and click **Next**.
3. In the next window, select **Job manager** as the type, and click **Next**.
Specify a name for the customization definition, and click **Next**. We specified a customization definition name of *ZJManager01*.
4. On the next window, specify defaults for GID and UID values, name, and user ID defaults based on a two-character prefix that identifies the cell and specify a default range for ports assigned to the process. Click **Next**.
5. You are now prompted to specify the high-level qualifier for the target z/OS data sets that contain the generated jobs and instructions that are created (refer to Figure 5-10 on page 131).

The generated batch jobs and instructions are uploaded to the following z/OS partitioned data sets:

- *HLQ.CNTL*
Partitioned data set with fixed block 80-byte records to contain customization jobs
- *HLQ.DATA*
Partitioned data set with variable-length data to contain other data contained in the customization definition

Tip: You can specify a multi-level high-level qualifier as the data set high-level qualifier.

We used WAS85.WPJMGR as the high-level qualifier.

Click **Next**.

6. In the Configure Common Groups window, enter the following information:
 - WebSphere Application Server configuration group information:
 - Specify the default group name for the WebSphere Application Server administrator user ID and all server user IDs.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.
 - WebSphere Application Server servant group information:
 - Specify the group name for all servant user IDs. You can use this group to assign subsystem permissions, such as DB2 authorizations, to all servants in the security domain.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value or assign a specific GID.
 - WebSphere Application Server local user group information:
 - Specify the group name for local clients and unauthorized user IDs (provides minimal access to the cell).
 - Select whether to allow the OS security system (RACF) to assign an unused GID value or assign a specific GID.

GID values: The specified GID is the UNIX System Services GID number for the WebSphere Application Server configuration group. GID values must be unique numeric values between 1 and 2,147,483,647.

We used the following values:

- WebSphere Application Server configuration group information: WSCFG1
- GID: Allow OS security to assign GID
- WebSphere Application Server servant group information: WSSR1
- GID: Allow OS security to assign GID
- WebSphere Application Server local user group information: WSCLGP
- GID: Allow OS security to assign GID

Click **Next**.

7. Configure the user ID settings:

- Common controller user ID
Defines the user ID that is associated with all of the control regions and the daemon. This user ID also owns the configuration file systems.
- Common servant user ID
Defines the user ID that is associated with the servant regions.
- WebSphere Application Server administrator user ID
Specifies the initial WebSphere Application Server administrator. The ID must have the WebSphere Application Server configuration group as its default UNIX System Services group. The UNIX System Services UID number for the administrator user ID is specified here and must be a unique numeric value between 1 and 2,147,483,647.

We used the following values:

- Common controller user ID: WJACR
- UID: Allow OS security to assign UID
- Common servant user ID: WJASR
- UID: Allow OS security to assign UID
- WebSphere Application Server administrator: WJADMIN
- UID: Allow OS security to assign UID
- WebSphere Application Server user ID home directory: /var/WebSphere/home

Click **Next**.

8. Provide the system and data set names to be used:

- Specify the system and sysplex name for the target z/OS system on which you will configure WebSphere Application Server for z/OS.
- Enter the name of an existing procedure library where the WebSphere Application Server for z/OS cataloged procedures are added.

We used the following values:

- System name: SC58
- Sysplex name: PLEX58
- PROCLIB data set name: SYS1.PROCLIB

Click **Next**.

9. In the Cell, Node and Server names window, enter the following information:
- Specify the long and short names for the cell, node, and servers. Short names identify the process to z/OS facilities, such as SAF. Long names are used as the primary external identification for the process. This is the name you will see in the administrative console. (The job manager server long name is set to the fixed value of jobmgr.)

Tip: Assign each management server (administrative agent, deployment manager, or job manager) its own cell name that is different from that of any other WebSphere Application Server cell on the same z/OS sysplex.

- Cluster transition name
If this server is converted into a clustered server, this name becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are part of the same cluster.

We used the following values:

- Cell short name: WJCELL
- Cell long name: wjcell
- Node short name: WJNODEA
- Node long name: wjnodea
- Server short name: WJS01A
- Server long name: jobmgr
- Cluster transition name: WJC01

Click **Next**.

10. Enter the Configuration File System values:

- Mount point
Application server configuration file system mount point: Specifies the read/write file system directory where the application data and environment files are written. This field is not writable here but was specified earlier on the System Environment: Configuration file system information window.
- Directory path name relative to mount point
The relative path name of the directory within the configuration file system in which the application server configuration resides.
- Data set name
The file system data set you will create and mount at the specified mount point above.
- File system type
Select to allocate and mount your configuration file system data set using HFS or zFS.
- Volume, or '*' for SMS:
Specify either the DASD volume serial number to contain the above data set or "*" to let SMS select a volume. Using "*" requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.
- Primary allocation in cylinders
The initial size allocation for the configuration file system data set. In the application server, the total space needed for this data set increases with the size and number of the installed applications. The minimum suggested size is 250 cylinders (3390).

- Secondary allocation in cylinders

The size of each secondary extent. The minimum suggested size is 100 cylinders.

We entered the following values:

- Mount point: /wasv85config/wjcell/wjnodea
- Directory path name relative to mount point: JobManager
- Data set name: OMVS.WAS85.WJCELL.WJNODEA.ZFS
- File system type: zSeries File System (zFS)
- Volume, or '*' for SMS: *
- Primary allocation in cylinders: 420
- Secondary allocation in cylinders: 100

Click **Next**.

11. Specify the information for the product file system:

- Specify the name of the directory where the product files for WebSphere Application Server for z/OS were stored during installation.
- Select the option to allow to set up an intermediate symbolic link and specify the path name.

We used the following values:

- Product file system directory: /usr/lpp/zWebSphere/V8R5
- Intermediate symbolic link: Create intermediate symbolic link
- Path name of intermediate symbolic link: /wasv85config/wjcell/wjnodea/wasInstall

Click **Next**.

12. Enter the process information. The job names for the processes are provided in the window and cannot be changed. Specify the procedure name for each process.

- Controller process job and procedure name

The job name for the control region is the same as the server short name. This is the name used in the MVS START command to start the region.

- Servant process job and procedure name

The job name is used by WLM to start the servant regions. This is set to the server short name followed by the letter "S."

We used the following values:

- Controller process job name: WJS01A
- Controller process procedure name: WJACRA
- Servant process job name: WJS01AS
- Servant process procedure name: WJASRA

Click **Next**.

13. Specify the application server port values shown in Figure 5-35. Good planning is important to avoid port conflicts. Ensure that you have all values that you need to complete the information in this window.

The screenshot shows the 'Port Values Assignment' window in Profile Management Tool 8.5. The window title is 'Profile Management Tool 8.5' and the subtitle is 'Management - job manager'. The window contains the following fields and values:

| Field | Value |
|---|---------------------|
| Node host name or IP address (3): | wtsc58.itso.ibm.com |
| JMX SOAP connector port: | 8876 |
| ORB listener IP address (4): | * |
| ORB listener port: | 9808 |
| ORB SSL listener port (Z): | 0 |
| HTTP transport IP address (5): | * |
| Administrative console port: | 9960 |
| Administrative console secure port: | 9943 |
| Administrative interprocess communication port (X): | 9631 |
| Status update listener port: | 9425 |

At the bottom of the window, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

Figure 5-35 Job manager port values

14. Enter the Location Service Daemon Definitions:

- Daemon home directory
Specifies the directory in which the location service daemon resides. This directory is set to the configuration file system mount point / daemon and cannot be changed.
- Daemon job name
Specifies the job name of the location service daemon, specified in the JOBNAME parameter of the MVS start command used to start the location service daemon.
- Procedure name
Defines the name of the member in your procedure library to start the location service daemon.
- IP Name
Specifies the fully-qualified IP name that is registered with the Domain Name Server (DNS), which the location service daemon uses.
- Listen IP
Defines that address at which the daemon listens.
- Port
Specifies the port number on which the location service daemon listens.
- SSL port
Specifies the port number on which the location service daemon listens for SSL connections.

- Register daemon with WLM DNS

If you use the WLM DNS (connection optimization), select this option to register your location service daemon with it; otherwise, do not select it.

We used the following values:

- Daemon home directory: /wasv85config/wjcell/wjnodea/Daemon
- Daemon job name: WJDEMNA
- Procedure name: WJDEMNA
- IP name: wtsc58.itso.ibm.com
- Listen IP address: *
- Port: 13060
- SSL Port: 13061
- Register daemon with WLM DNS: not enabled

Click **Next**.

15. Enter the information that is required for SSL connections.

We used the following values:

- Certificate authority keylabel: WebSphereCAJM
- Generate certificate authority (CA) certificate: Enabled
- Expiration date for certificates: 2021/12/31
- Default SAF keyring name: WASKeyring.WJCELL
- Use virtual keyring for z/OS SSL clients: Not enabled
- Enable SSL on location service daemon: Not enabled

Click **Next**.

16. Select the user registry that will be used to manage user identities and the authorization policy from one of the following choices:

- z/OS security product

This option uses the z/OS system's SAF-compliant security product, such as IBM RACF or equivalent, to manage WebSphere Application Server identities and authorization according to the following rules:

- The SAF security database will be used as the WebSphere user repository.
- SAF EJBROLE profiles are used to control role-based authorization, including administrative authority.
- Digital certificates are stored in the SAF security database.

Important: Select the z/OS security product option if you plan to use the SAF security database as your WebSphere Application Server registry or if you plan to set up an LDAP or custom user registry whose identities will be mapped to SAF user IDs for authorization checking. For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

- WebSphere Application Server security

The WebSphere Application Server administrative security option is used to manage the Application Server identities and authorization as follows:

- A simple file-based user registry will be built as part of the customization process.
- Application-specific role binds will be used to control role-based authorization.
- The WebSphere Application Server console users and groups list will control administrative authority.

- Digital certificates are stored in the configuration file system as keystores.

Tip: Choose this option if you plan to use an LDAP or custom user registry without mapping to SAF user IDs. (The file-based user registry is not a best practice for production use.)

- No security

Although it is not a best practice, you can disable administrative security. If you choose this security option, there are no other choices to make. Your WebSphere Application Server environment is not secured until you configure and enable security manually. You can enable security manually later using the administrative console or using Jython scripts.

We used the “Use z/OS security product” option.

Click **Next**.

17. In the next window, choose one of the following options:

- SAF profile prefix (formally known as *Security domain identifier*)

This optional parameter is used to distinguish between APPL or EJBROLE profiles based on the security domain name. It provides an alphanumeric security domain name of one to eight characters. Internally, this sets SecurityDomainType to the string cellQualified.

All servers in the cell prepend the security domain name that you specify to the application-specific J2EE role name to create the SAF EJBROLE profile for checking. The security domain name is not used, however, if role checking is performed using WebSphere Application Server for z/OS bindings.

The security domain name is also used as the APPL profile name and inserted into the profile name used for CBIND checks. The RACF jobs that the Customization Dialog generates create and authorize the appropriate RACF profiles for the created nodes and servers.

If you do not want to use a security domain identifier, leave this field blank.

- WebSphere Application Server unauthenticated user ID

Associated with unauthenticated client requests. It is sometimes referred to as the “guest” user ID. Give it the RESTRICTED attribute in RACF to prevent it from inheriting UACC-based access privileges. The UNIX System Services UID number for the user ID is specified here and is associated with unauthenticated client requests. The UID value must be unique, numeric values between 1 and 2,147,483,647.

- Enable Writable SAF Keyring support

This feature allows the administrative console to create and sign certificates by a CA, connect to keyrings, remove from keyrings, and import, export, and renew.

All certificates created with the writable keyring support are generated and signed by Java code and not by SAF. In this case, the writable keyring support only uses SAF to store the generated certificates.

The writable keyring support is completely optional. New keystores and truststores that are marked as read-only can be created independently from the writable keyring support. When using the read-only JCERACFKS and JCECCARACFS keystores, the certificates in the appropriate SAF keyring can still be viewed in the administrative console.

We used the following values:

- SAF profile prefix (optional): WJ
- WebSphere Application Server unauthenticated user ID: WJGUEST
- UID: Allow OS security to assign UID
- Enable Writable SAF Keyring support: Enabled

Click **Next**.

18. Tailor the JCL for the customization jobs. Enter a valid job statement for your installation. The profile creation process will update the job name for you in all the generated jobs, so do not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.
19. In the Customization Summary, click **Create** to store this application server environment definition for later transfer to the intended z/OS host system.
20. Click **Finish** when the jobs are complete.

5.4.2 Uploading the jobs and the associated instructions

Upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile, and click **Process**. To upload the generated jobs to the target z/OS system, select from the following options:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file system

Click **Next**.

2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path name must be fully qualified or the upload will fail. You must also specify the user ID, password, and FTP server port.

Select **Allocate target z/OS data sets** to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish**, and a progress bar displays while the upload occurs.

3. After the customization profile is uploaded, select the customization definition in the Profile Management Tool, and click the **Customization Instructions** tab. This tab provides complete instructions about how to build the profile using the jobs.

These instructions help you determine the jobs to run, the order in which to run them, and the expected results. It also explains how to start the environment after you are finished.

After the jobs run successfully, the application server profile is complete.

5.5 Creating an administrative agent profile

The administrative agent provides a single interface in which to administer multiple application server nodes, such as development, unit test, or server farm environments. Using a single interface to administer your application servers reduces the impact of running administrative services in every application server.

5.5.1 Creating the customization definition

To begin, run the profile management tool to create the customization definition:

1. On the Profile Management Tool main window, click **Create**.
2. To generate the definitions for the Job Manager, select **Management** in the Profile Management tool environment selection window, and click **Next**.
3. In the next window, select **Administrative agent** as the type, and click **Next**.
Specify a name for the customization definition, and click **Next**. We defined a customization definition name of ZAdminAg01.
4. Specify defaults for GID and UID values, name, and user ID defaults based on a two-character prefix that identifies the cell, and specify a default range for ports assigned to the process. Click **Next**.
5. Specify the high-level qualifier for the target z/OS data sets that will contain the generated jobs and instructions that are created.

The generated batch jobs and instructions are uploaded to the following z/OS partitioned data sets:

- *HLQ.CNTL*
Partitioned data set with fixed block 80-byte records to contain customization jobs
- *HLQ.DATA*
Partitioned data set with variable-length data to contain other data contained in the customization definition

Tip: You can specify a multi-level high-level qualifier as the data set high-level qualifier.

We used WAS85.WPADMAG as the high-level qualifier.

Click **Next**.

6. In the Configure Common Groups window, enter the following information:
 - WebSphere Application Server configuration group information:
 - Specify the default group name for the WebSphere Application Server administrator user ID and all server user IDs.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.
 - WebSphere Application Server servant group information:
 - Specify the group name for all servant user IDs. You can use this group to assign subsystem permissions, such as DB2 authorizations, to all servants in the security domain.
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.
 - WebSphere Application Server local user group information:
 - Specify the group name for local clients and unauthorized user IDs (provides minimal access to the cell).
 - Select whether to allow the OS security system (RACF) to assign an unused GID value, or assign a specific GID.

GID values: The specified GID is the UNIX System Services GID number for the WebSphere Application Server configuration group. GID values must be unique numeric values between 1 and 2,147,483,647.

We used the following values:

- WebSphere Application Server configuration group information: WACFG
- GID: Allow OS security to assign GID
- WebSphere Application Server servant group information: WASRG
- GID: Allow OS security to assign GID
- WebSphere Application Server local user group information: WAGUESTG
- GID: Allow OS security to assign GID

Click **Next**.

7. Configure the user ID settings:

- Common controller user ID
Specifies the user ID that is associated with all of the control regions and the daemon. This user ID also owns the configuration file systems.
- Common servant user ID
Specifies the user ID that is associated with the servant regions.
- WebSphere Application Server administrator user ID
Defines the initial WebSphere Application Server administrator. The ID must have the WebSphere Application Server configuration group as its default UNIX System Services group. The UNIX System Services UID number for the administrator user ID is specified here, and must be a unique numeric value between 1 and 2,147,483,647.

We used the following values:

- Common controller user ID: WAACR
- UID: Allow OS security to assign UID
- Common servant user ID: WAASR
- UID: Allow OS security to assign UID
- WebSphere Application Server administrator: WAADMIN
- UID: Allow OS security to assign UID
- WebSphere Application Server user ID home directory: /var/WebSphere/home

Click **Next**.

8. Provide the system and data set names to be used:

- Specify the system and sysplex name for the target z/OS system on which you will configure WebSphere Application Server for z/OS.
- Enter the name of an existing procedure library where the WebSphere Application Server for z/OS cataloged procedures are added.

We used the following values:

- System name: SC58
- Sysplex name: PLEX58
- PROCLIB data set name: SYS1.PROCLIB

Click **Next**.

9. In the Cell, Node and Server names window, enter the following information:
- Specify the long and short names for the cell, node, and servers. Short names identify the process to z/OS facilities, such as SAF. Long names are used as the primary external identification for the process. This is the name you see in the administrative console. (The job manager server long name is set to the fixed value of adminagent.)

Tip: Assign each management server (administrative agent, deployment manager, or job manager) its own cell name that is different from that of any other WebSphere Application Server cell on the same z/OS sysplex.

- Cluster transition name
If this server is converted into a clustered server, this name becomes the cluster short name. The cluster short name is the WLM APPLENV name for all servers that are part of the same cluster.

We used the following values:

- Cell short name: WAADMA
- Cell long name: waadma
- Node short name: WANODEA
- Node long name: wanodea
- Server short name: WAS01A
- Server long name: adminagent
- Cluster transition name: WAC01

Click **Next**.

10. In the Configuration File System window, enter the following information:

- Mount point
Application server configuration file system mount point: Specifies the read/write file system directory where the application data and environment files are written. This field is not writable here, but was specified earlier on the System Environment: Configuration file system information window.
- Directory path name relative to mount point
The relative path name of the directory within the configuration file system in which the application server configuration resides.
- Data set name
The file system data set that you create and mount at the specified mount point.
- File system type
Select to allocate and mount your configuration file system data set using HFS or zFS.
- Volume, or '*' for SMS
Specify either the DASD volume serial number to contain the above data set or "*" to let SMS select a volume. Using "*" requires that SMS automatic class selection (ACS) routines be in place to select the volume. If you do not have SMS set up to handle data set allocation automatically, list the volume explicitly.
- Primary allocation in cylinders
The initial size allocation for the configuration file system data set. In the application server, the total space needed for this data set increases with the size and number of the installed applications. The minimum suggested size is 250 cylinders (3390).

- Secondary allocation in cylinders

The size of each secondary extent. The minimum suggested size is 100 cylinders.

We used the following values:

- Mount point: /wasv85config/waadma/wanodea
- Directory path name relative to mount point: AdminAgent
- Data set name: OMVS.WAS85.WAADMA.WANODEA.ZFS
- File system type: zSeries File System (zFS)
- Volume, or '*' for SMS: *
- Primary allocation in cylinders: 420
- Secondary allocation in cylinders: 100

Click **Next**.

11. Specify the information for the product file system:

- Specify the name of the directory where the product files for WebSphere Application Server for z/OS were stored during installation.
- Select the option to allow to set up an intermediate symbolic link and specify the path name.

We used the following values:

- Product file system directory: /usr/lpp/zWebSphere/V8R5
- Intermediate symbolic link: Create intermediate symbolic link
- Path name of intermediate symbolic link: /wasv85config/waadma/wanodea/wasInstall

Click **Next**.

12. Enter the process information. The job names for the processes are provided in the window and cannot be changed. Specify the procedure name for each process:

- Controller process job and procedure name
The job name for the control region is the same as the server short name. This is the name used in the MVS START command to start the region.
- Servant process job and procedure name
The job name is used by WLM to start the servant regions. This is set to the server short name followed by the letter "S."

We used the following values:

- Controller process job name: WAS01A
- Controller process procedure name: WAACRA
- Servant process job name: WAS01AS
- Servant process procedure name: WAASRA

Click **Next**.

13. Specify the application server port values, as shown in Figure 5-36. Good planning is important to avoid port conflicts. Ensure that you have all values that you need to complete the information in this window.

Profile Management Tool 8.5

Port Values Assignment

Management - administrative agent

Node host name or IP address (3): wtsc58.itso.ibm.com

JMX SOAP connector port: 8877

ORB listener IP address (4): *

ORB listener port: 9807

ORB SSL listener port (Z): 0

HTTP transport IP address (5): *

Administrative console port: 9060

Administrative console secure port: 9043

Administrative interprocess communication port (X): 9630

< Back Next > Finish Cancel

Figure 5-36 Admin Agent port values

14. In the Location Service Daemon Definitions window, enter the following information:

- Daemon home directory
Specifies the directory in which the location service daemon resides. This is set to the configuration file system mount point / daemon and cannot be changed.
- Daemon job name
Specifies the job name of the location service daemon, specified in the JOBNAME parameter of the MVS start command used to start the location service daemon.
- Procedure name
Defines the name of the member in your procedure library to start the location service daemon.
- IP Name
Specifies the fully-qualified IP name that is registered with the Domain Name Server (DNS), which the location service daemon uses.
- Listen IP
Specifies the address at which the daemon listens.
- Port
Provides the port number on which the location service daemon listens.
- SSL port
Provides the port number on which the location service daemon listens for SSL connections.

- Register daemon with WLM DNS

If you use the WLM DNS (connection optimization), you must select this option to register your location service daemon with it; otherwise, do not select it.

We used the following values:

- Daemon home directory: /wasv85config/waadma/wanodea/Daemon
- Daemon job name: WADEMNA
- Procedure name: WADEMNA
- IP name: wtsc58.itso.ibm.com
- Listen IP address: *
- Port: 13080
- SSL Port: 13081
- Register daemon with WLM DNS: Not enabled

Click **Next**.

15. Enter the information that is required for SSL connections.

We entered the following values:

- Certificate authority keylabel: WebSphereCAAA
- Generate certificate authority (CA) certificate: Enabled
- Expiration date for certificates: 2021/12/31
- Default SAF keyring name: WASKeyring.WAADMA
- Use virtual keyring for z/OS SSL clients: Not enabled
- Enable SSL on location service daemon: Not enabled

Click **Next**.

16. Select the user registry that will be used to manage user identities and the authorization policy. Select one of the following options:

- z/OS security product

This option uses the z/OS system's SAF compliant security product, such as IBM RACF or equivalent, to manage WebSphere Application Server identities and authorization according to the following rules:

- The SAF security database will be used as the WebSphere user repository.
- SAF EJBROLE profiles will be used to control role-based authorization, including administrative authority.
- Digital certificates are stored in the SAF security database.

Important: Select the z/OS security product option if you plan to use the SAF security database as your WebSphere Application Server registry or if you plan to set up an LDAP or custom user registry whose identities will be mapped to SAF user IDs for authorization checking. For this security option, you must decide whether to set a security domain name, and choose an administrator user ID and an unauthenticated (guest) user ID.

- WebSphere Application Server security

The WebSphere Application Server administrative security option is used to manage the Application Server identities and authorization as follows:

- A simple file-based user registry will be built as part of the customization process.
- Application-specific role binds will be used to control role-based authorization.
- The WebSphere Application Server console users and groups list control administrative authority.

- Digital certificates are stored in the configuration file system as keystores.

Tip: Choose this option if you plan to use an LDAP or custom user registry without mapping to SAF user IDs. (The file-based user registry is not recommended for production use.)

- No security

Although it is not a best practice, you can disable administrative security. If you choose this security option, there are no other choices to make. Your WebSphere Application Server environment is not secured until you configure and enable security manually. You can enable security manually later using the administrative console or using Jython scripts.

We used the “Use z/OS security product” option.

Click **Next**.

17. Select one of the following options:

- SAF profile prefix (formally known as *Security domain identifier*)

This optional parameter is used to distinguish between APPL or EJBROLE profiles based on security domain name. It provides an alphanumeric security domain name of one to eight characters. Internally, this sets SecurityDomainType to the string cellQualified.

All servers in the cell prepend the security domain name you specify to the application-specific J2EE role name to create the SAF EJBROLE profile for checking. The security domain name is not used, however, if role checking is performed using WebSphere Application Server for z/OS bindings.

The security domain name is also used as the APPL profile name and inserted into the profile name used for CBIND checks. The RACF jobs that the Customization Dialog generates create and authorize the appropriate RACF profiles for the created nodes and servers.

If you do not want to use a security domain identifier, leave this field blank.

- WebSphere Application Server unauthenticated user ID

Associated with unauthenticated client requests. It is sometimes referred to as the “guest” user ID. Give it the RESTRICTED attribute in RACF to prevent it from inheriting UACC-based access privileges. The UNIX System Services UID number for the user ID is specified here and is associated with unauthenticated client requests. The UID value must be unique numeric values between 1 and 2,147,483,647.

- Enable Writable SAF Keyring support

This feature allows administrative console to create and sign certificates by a CA, connect to keyrings, remove from keyrings, and import, export, and renew.

All certificates created with the writable keyring support are generated and signed by Java code and not by SAF. In this case, the writable keyring support only uses SAF to store the generated certificates.

The writable keyring support is completely optional. New keystores and truststores marked as read-only can be created independently from the writable keyring support. When using the read-only JCERACFKS and JCECCARACFS keystores, the certificates in the appropriate SAF keyring can still be viewed in the administrative console.

We used the following values:

- SAF profile prefix (optional): WA
- WebSphere Application Server unauthenticated user ID: WAGUEST
- UID: Allow OS security to assign UID
- Enable Writable SAF Keyring support: Enabled

Click **Next**.

18. Tailor the JCL for the customization jobs. Enter a valid job statement for your installation on this window. The profile creation process updates the job name for you in all the generated jobs, so do not be concerned with that portion of the job statement. If continuation lines are needed, replace the comment lines with continuation lines. Click **Next**.
19. In the Customization Summary panel, click **Create** to store this application server environment definition for later transfer to the intended z/OS host system.
20. Click **Finish** when the jobs are complete.

5.5.2 Uploading jobs and the associated instructions

Upload these jobs and the associated instructions to a pair of z/OS partitioned data sets:

1. On the main window, select the customization definition for the profile, and click **Process**. To upload the generated jobs to the target z/OS system select from the following options:
 - Upload to target z/OS system using FTP
 - Upload to target z/OS system using FTP over SSL
 - Upload to target z/OS system using secure FTP
 - Export to local file system

Click **Next**.

2. If you choose to upload the customization using FTP, in the upload customization definition window, enter the target z/OS system. This path name must be fully qualified or the upload will fail. You must also specify the user ID and password and FTP server port.

Click **Allocate target z/OS data sets** to specify whether to allocate the data sets if they do not exist. If the data sets exist and are to be reused, clear this option.

Click **Finish**, and a progress bar displays while the upload occurs.

3. After the customization profile is uploaded, select the customization definition in the Profile Management Tool, and click the **Customization Instructions** tab. This tab provides complete instructions about how to build the profile using the jobs.

These instructions help you to determine the jobs to run, the order in which to run them, and the expected results. It also explains how to start the environment after you are finished.

After the jobs run successfully, the application server profile is complete.



Administration consoles and commands

WebSphere Application Server properties are stored in the configuration repository as XML files. Manually editing any of the configuration files bypasses the validation of any changes and leads to synchronization-related problems. WebSphere Application Server provides administrative tools that help you administer the environment and avoid the risks of manual editing. These tools manage modifications to the files in the repository.

In this chapter, we introduce the administrative consoles and command-line administration. Information about scripting is provided in Chapter 8, “Administration with scripting” on page 319.

In this chapter we cover the following topics:

- ▶ Introducing the WebSphere administrative consoles
- ▶ Securing the administrative console
- ▶ Job manager console
- ▶ Using command-line tools

6.1 Introducing the WebSphere administrative consoles

The WebSphere Integrated Solutions Console, referred to as the administrative console, is a graphical, web-based tool that is used to configure and manage the resources within your WebSphere environment.

The administrative console application name is *isclite*, and it is a *system* application. This means that the application is central to a WebSphere Application Server product, and it is installed when the product is installed. In this case, the administrative console application is installed during profile creation, if selected, or afterwards using the command line. You do not see system applications in the list of installed applications when using the administrative console. You cannot stop or start the application directly or uninstall the application directly.

With the introduction of the flexible management topologies, there are multiple administrative consoles available in a WebSphere solution:

- ▶ Administrative console hosted by an application server or deployment manager in case of a Network Deployment environment:

This administrative console is used to manage an entire WebSphere cell. It supports the full range of product administrative activities, such as creating and managing resources and applications, viewing product messages, and so on.

In a stand-alone server environment, the administrative console is located on the application server and can be used to configure and manage the resources of that server only.

In a Network Deployment environment, the administrative console is located in the deployment manager server, *dmgr*. In this case, the administrative console provides centralized administration of multiple nodes. Configuration changes are made to the master repository and pushed to the local repositories on the nodes by the deployment manager.

- ▶ Administrative agent console:

An administrative agent hosts the administrative console for application server nodes that are registered to it.

When you access the URL for the administrative console, you can select the node type to manage. After your selection is made, you are directed to the appropriate administrative console where you can log into:

- Administrative console for the administrative agent:

This console allows you to manage the administrative agent, including security settings. You can also register nodes that the administrative agent controls with the job manager.

- Administrative console for an application server:

This console is the administrative console for the application server.

- ▶ Job manager administrative console (referred to as the job manager console):

The job manager console provides the interface to manage the job manager itself, including security settings and mail resources. Its primary function is to submit jobs for processing on the nodes that are registered to it.

6.1.1 Starting and accessing the consoles

The way that you access the administrative console is the same whether you have a stand-alone server environment or a distributed server environment. However, the location and how you start the necessary processes varies.

Finding the URL for the console

Each application server process that hosts the administrative console has two admin ports that are used to access the administrative console. These ports are:

- ▶ WC_adminhost
- ▶ WC_adminhost_secure (for SSL communication)

These ports are assigned at profile creation time. If you do not know which is the port number for the administrative console, look in the following location:

- ▶ In case of a Network Deployment environment: *profile_home*/properties/portdef.props
- ▶ In case of a stand-alone environment:
profile_home/config/cells/*cell_name*/nodes/*node_name*/serverindex.xml

Use the following URL to access the administrative console using the non-SSL port:

```
http://<hostname>:<WC_adminhost>/ibm/console
```

Use the following URL to access the administrative console using the SSL port:

```
https://<hostname>:<WC_adminhost_secure>/ibm/console
```

If administrative security is enabled, you are automatically redirected to the secure port.

Administrative console in a stand-alone server environment

In a single application server installation, the administrative console is hosted by the application server. The server must be started to access the administrative console.

To access the administrative console:

1. Make sure that your application server is running by entering the following command:

```
serverStatus.bat (sh) -all
```

The **serverStatus** command is used to obtain the status of one or all of the servers configured on the node. You provide the server name as an argument, or use the **-all** argument. The default server name is *server1*. Example 6-1 shows the output of this command.

Example 6-1 Example output for the serverStatus command in a Windows environment

```
D:\was85\IBM\WebSphere\AppServer\profiles\AppSrv_85_02\bin>serverstatus.bat
-all
ADMU0116I: Tool information is being logged in file

D:\was85\IBM\WebSphere\AppServer\profiles\AppSrv_85_02\logs\serverStatus.log
ADMU0128I: Starting tool with the AppSrv_85_02 profile
ADMU0503I: Retrieving server status for all servers
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server_85_2
ADMU0509I: The Application Server "server_85_2" cannot be reached. It appears
to be stopped.
```

2. If the status of your server is not STARTED, start it with the following command:
`startServer.bat (sh) server_name`
3. Open a web browser to the URL of the administrative console, for example:
`https://<hostname>:9050/ibm/console`
<hostname> is the host name for the machine running the application server. You can always use the IP of the machine instead of the *<hostname>*.
4. The administrative console loads and you are prompted to log in.

Administrative console in a Network Deployment environment

If you are working with a deployment manager and its managed nodes, the administrative console is hosted by the deployment manager. You must start the deployment manager to use the administrative console. The default name for the deployment manager is *dmgr*.

To access the administrative console:

1. Make sure that the deployment manager (dmgr) is running by entering the following command:
`serverStatus.sh -all`
2. If the dmgr status is not STARTED, start it with the following command:
`startManager.sh`
3. Open a web browser to the URL of the administrative console, for example:
`https://<hostname>:9050/admin`
<hostname> is the host name for the machine running the deployment manager. You can always use the IP of the machine instead of the *<hostname>*.
4. The administrative console loads and you are prompted to log in.

Accessing the job manager console

If you are working with a job manager, the administrative console is hosted by the job manager. The default name of the job manager is *jobmgr*. To access the job manager administrative console:

1. Make sure that the job manager process (jobmgr) is running by using the following command:
`serverStatus.sh -all`
2. If the status of jobmgr is not STARTED, start it with the following command:
`startServer.sh jobmgr`
3. Open a web browser to the URL of the administrative console, for example:
`http://<hostname>:9960/ibm/console`
4. The administrative console loads and you are prompted to log in.

Accessing the administrative agent administrative console

If you are working with an administrative agent, the administrative console is hosted by the administrative agent. The default name of the job manager is *adminagent*. To access the administrative agent administrative console:

1. Make sure that the administrative agent process (adminagent) is running by using the following command:
`serverStatus.sh -all`

2. If the status of adminagent process is not STARTED, start it with the following command:
`startServer.sh adminagent`
3. Open a web browser to the URL of the administrative console, for example:
`http://<hostname>:9060/ibm/console`
If you have nodes registered with the administrative agent, you are prompted to select which node to administer (including the administrative agent).
4. The administrative console loads and you are prompted to log in.

6.1.2 Logging into an administrative console

When you access the administrative console, you need to log in by providing a user ID. If WebSphere administrative security is enabled, you also need to provide a password.

The user ID specified during login is used to track configuration changes made by the user. This allows you to recover from unsaved session changes made under the same user ID, for example, when a session times out or the user closes the web browser without saving. The configuration files are copied from the master repository and cached in the temporary workspace because you navigate through different console areas. Configuration changes are stored in the `profile_home/ws temp` temporary workspace directory until the changes are merged with the master repository during a save operation in the administrative console. Workspaces are not removed when you log out, so they can be reused in another login session for the same login user ID.

Note: You cannot log into two instances of administrative consoles that are running on the same machine from a single browser type. For example, if you use Firefox to log into the deployment manager administrative console, you cannot also log into a job manager running on the same machine.

There is a limitation that cookies are unique per domain rather than a combination of domain and port. Therefore, the cookies that control the session and authentication data in the first browser tab or window get overwritten when logging into the other console in a new browser tab or window. However, it is possible to log into two consoles simultaneously from two completely different browsers, for example, Firefox and Internet Explorer.

WebSphere administrative security also affects the log in procedure. The following scenarios relate how to maneuver in either security state:

- If WebSphere administrative security is not enabled

You can enter any user ID, valid or not, to log in to the administrative console. The user ID is used to track changes to the configuration but is not authenticated. You can also simply leave the User ID field blank and click the **Log In** button. The administrative security is not enabled, so you cannot see any password field in the administrative console login window.

Note: Logging in without an ID is not a best practice, especially if you have multiple administrators.

► If WebSphere administrative security is enabled

You must enter a valid user ID and password that was already assigned an administrative security role.

If you enter a user ID that is already in session, a message “Another user is currently logged in with the same User ID” is displayed, and you are prompted to do one of the following actions:

- Log out of the other user with the same user ID. You are allowed to recover changes that were made in the other user’s session.
- Return to the login page and specify a different user ID.

Figure 6-1 illustrates the login window in a user session conflict situation.



Figure 6-1 Administrative console login window - Options when the user is already logged in

Note: You also get the message, in Figure 6-1, if a previous session ended without a logout. For example, if the user closed a web browser during a session and did not log out first or if the session timed out. Changes made in an interrupted session are recoverable.

Recovering from an interrupted session

Until you save the configuration changes you make during a session, the changes do not become effective. During a save operation, the changes propagate and merge with the master configuration repository. If a session is closed without saving the configuration changes made during the session, these changes are remembered, and you have an opportunity to recover the changes. The changes are currently stored in the ws temp temporary workspace directory.

When unsaved changes for the user ID exist during login, you are prompted to complete one of the following actions:

- ▶ Work with the master configuration

Selecting this option specifies that you want to use the last saved administrative configuration. Changes made to the user's session since the last saving of the administrative configuration are lost.

- Recover changes made in a prior session

Selecting this option specifies that you want to use the same administrative configuration last used for the user's session. It recovers all changes made by the user since the last saving of the administrative configuration for the user's session.

As you work with the configuration, the original configuration file and the new configuration file are stored in a folder at:

`<profile_home>/wstemp`

If you log out of the administrative console and the session is correctly ended, the session directories in the `wstemp` folder are automatically removed. If the session is interrupted, for example, by closing the web browser instead of logging out, the directories remain in the file system.

It is safe to delete the contents of the `wstemp` folder to free space on the file system. After deleting the contents, the deployment manager server must be stopped and restarted. Before stopping the server, verify that no one is logged in or their session will be corrupted.

Find more information about this process and how to modify the location of the `wstemp` folder in the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/tcfg_console_workspacefiles.html

Installing and uninstalling the administrative console application

You can install the administrative console during profile creation or after you create a profile. You can uninstall any administrative console that you install.

Unfederated application servers, administrative agents, deployment managers, and job managers can have their own administrative consoles. If you plan to install the administrative console, a profile that does not have an administrative console installed must exist. You cannot have two administrative consoles running in the same profile.

To install an administrative console after profile creation or to uninstall the administrative console, use the `wsadmin` command to run a jython script named `deployConsole.py`. This script is located in the `bin` folder of the IBM WebSphere Application Server installation root and can be run in either connected or disconnected mode. The usual security restrictions for the `wsadmin` command apply to this script. In connected mode, the user must authenticate if security is enabled.

Example 6-2 shows an installation command, and Example 6-3 on page 191 shows an uninstallation command of the administrative console in a deployment manager profile using the jython script `deployConsole.py`. The `wsadmin` command attempts to remotely connect to the deployment manager. So, start your deployment manager process before running either commands. You have to run the commands on the deployment manager node and not on a federated node. In our examples, the user name and its password are `admin85` and they are to be replaced with your values when using the commands.

Example 6-2 Installing the administrative console with the Jython script `deployConsole.py`

```
D:\was85\IBM\WebSphere\AppServer\profiles\Dmgr_85_01\bin>wsadmin -lang jython
-c AdminControl.getNode() -user admin85 -password admin85 -f
"D:\was85\IBM\WebSphere\AppServer\bin\deployConsole.py" install
```

Example 6-3 Uninstalling the administrative console with the Jython script deployConsole.py

```
D:\was85\IBM\WebSphere\AppServer\profiles\Dmgr_85_01\bin>wsadmin -lang jython
-c AdminControl.getNode() -user admin85 -password admin85 -f
"D:\was85\IBM\WebSphere\AppServer\bin\deployConsole.py" remove
```

Administrative console application logs

In case you need the history of the administrative console actions and events for further audits, you have the option to enable the log details for the specific components in the isclite application. Logs are useful especially in sensitive production environments. To enable the log details, select the desired log detail level for the `com.ibm.isclite.*` component. In case of a deployment manager profile, this component is located in the deployment manager server log and trace levels configuration. In case of another type of profile (unfederated application servers, administrative agents, or job manager), this component is located in the server log and trace levels configuration. For a deployment manager profile, the following steps take you through this process:

1. Click **Troubleshooting** → **Logs and trace**.
2. Select the deployment manager server name.
3. Click **Change log detail levels**.
4. In the Configuration tab, expand the **Components and Groups** list.
5. Expand the ***[All Components]** list, and locate the `com.ibm.isclite.*` component.
6. Select the required log detail level for this component or its sub-components.
7. Save the changes, and restart the deployment manager server process.

The log entries for the isclite application are available in the JVM logs of the deployment manager process (the default names are `SystemOut.log` and `SystemErr.log`).

Figure 6-2 on page 192 shows the log details levels for the `com.ibm.isclite.*` component.

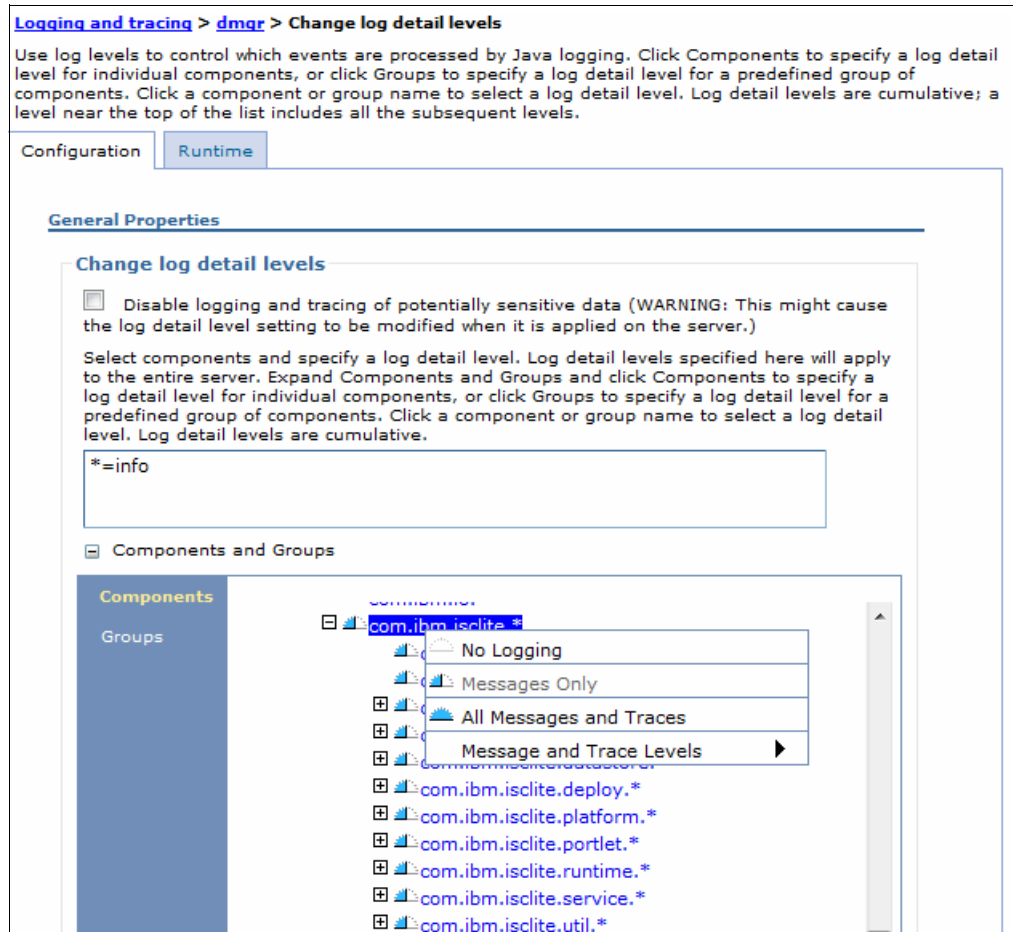


Figure 6-2 Log details for the deployment manager application components

6.1.3 Changing the administrative console session timeout

The idle period, before the administrative console session expires, is referred to as session timeout. The default session timeout value for the administrative console is 15 minutes. The timeout value can be modified by using a JACL script that is available at the information center.

To change the session timeout value, refer to the information center at the following website:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/isc/cons_sessionto.html

6.1.4 The graphical interface

The WebSphere administrative consoles have the same layout pattern. In each administrative console, you can find the following main areas:

- ▶ Banner
- ▶ Navigation tree
- ▶ Work area, including the messages and help display areas

Each area can be resized as desired. The difference in the administrative console types is noted in the Navigation tree. The options that you find there vary depending on the administrative console type.

Figure 6-3 shows the administrative console hosted on a deployment manager to illustrate the console layout.

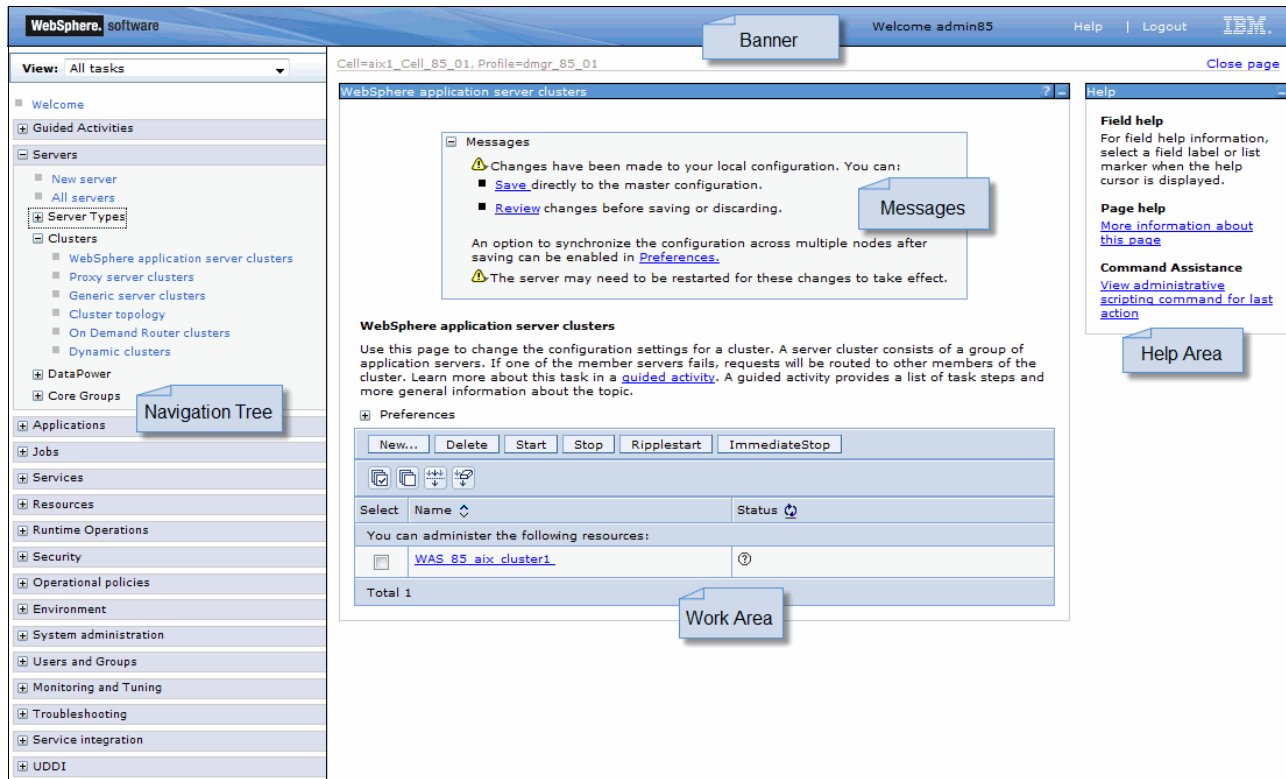


Figure 6-3 The administrative console graphical interface

Banner

The banner is the horizontal bar near the top of the administrative console. It includes a greeting to the user who is logged in. The banner also provides the following actions:

- ▶ Logout logs you out of the administrative console session and displays the login page. If you changed the administrative configuration since last saving the configuration to the master repository, the Save page displays before returning you to the login page. Click **Save** to save the changes. Click **Discard** to return to the administrative console, or **Logout** to exit the session without saving changes.
- ▶ Help opens a new web browser with detailed online help for the administrative console. This is not part of the information center.

Console identity

The banner displays a user ID and it can be customized to show a unique identifier for the administrative console. This can be helpful in cases where administrators log on to multiple administrative consoles. Glancing at the banner lets you know which system you are logged on to. You can add a Console Identity from the administrative console.

To customize the banner:

1. click **System administration** → **Console Identity**. Select **Custom**, and enter the identity string, as shown in Figure 6-4 on page 194.

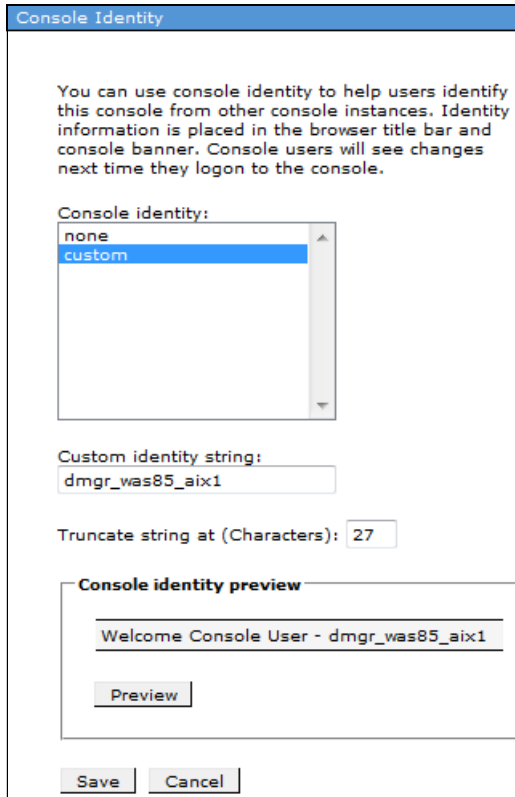


Figure 6-4 Customizing the banner

2. Click **Save**, and log out of the administrative console and then log back in. After you log back in, you will see the new console identity in the banner.

In an administrative agent configuration, the changes are applied to the administrative agent and all of its registered application servers, regardless of where the changes were actually saved.

Navigation tree

The navigation tree on the left side of the administrative console offers links for you to view, select, and manage components. Figure 6-5 shows the navigation tree.



Figure 6-5 Navigation Tree

Clicking a + sign beside a tree folder or item expands the tree for the folder or item. Clicking a - sign collapses the tree for the folder or item. Double-clicking an item toggles its state between expanded and collapsed.

The content displayed on the right side of the administrative console, the *workspace*, depends on the folder or item selected in the tree view.

Guided Activities

The navigation tree includes a category called *Guided Activities*. This section contains step-by-step assistance for performing some common tasks. Each of these activities can be accomplished manually and without guidance, but using the *Guided Activities* option provides additional assistance when desired.

Workspace

The workspace, on the right side of the administrative console, allows you to work with your administrative configuration after selecting an item from the administrative console navigation tree.

When you click a folder in the tree view, the workspace lists information about instances of that folder type in the collection page. For example, clicking **Servers** → **Server Types** → **WebSphere application servers** shows all of the application servers configured in this cell. Selecting an item, an application server in this example, displays the detail page for that item. The detail page can contain multiple tabs. For example, you might have a Runtime tab for displaying the runtime status of the item and a Configuration tab for viewing and changing the configuration of the displayed item.

Messages

When you perform administrative actions, messages are shown at the top of the workspace to display the progress and results. These messages are limited in nature, so if an action fails, review the JVM process logs for more detailed information.

When configuration changes are made, the message area contains links that you can click to review or save the changes.

Breadcrumb trail

As you navigate into multiple levels of a configuration page, a breadcrumb trail is displayed at the top of the workspace. It indicates how you reached the current page and provides links that allow you to go back to previous pages easily without starting the navigation trail over, as shown in Figure 6-6.

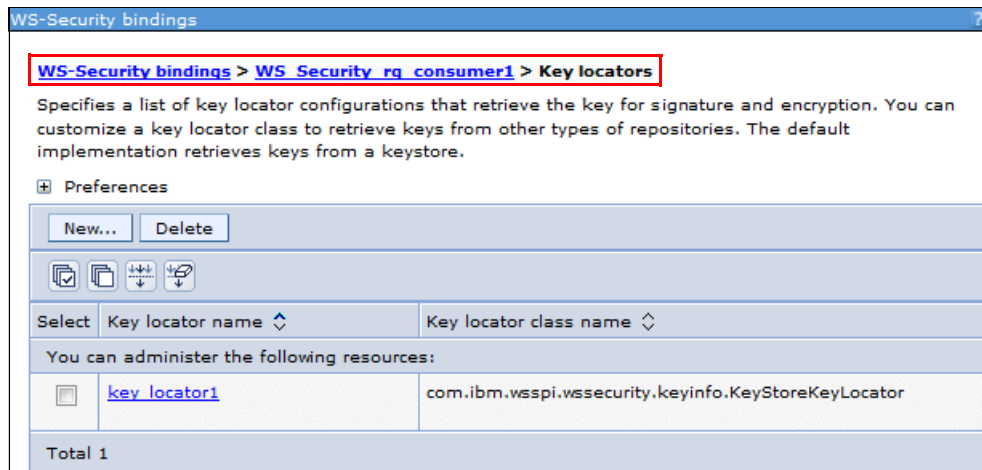


Figure 6-6 Breadcrumb trail

Help area

As you are working in the administrative console, help is available in multiple ways. As you hover the mouse over a field, help text is displayed for that field.

On the right side of the administrative console, the help portlet displays the Help area. Most pages have a **More information about this page** link in the Help area. Clicking the link opens the online help in a separate browser. Many pages have a **View administrative scripting command for last action** link. Clicking this link displays an equivalent scripting command for the action you just performed, as shown in Figure 6-7.

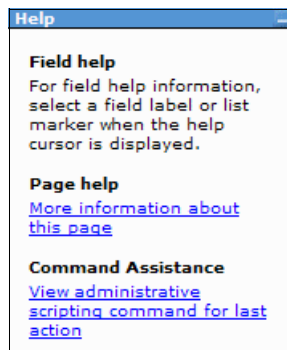


Figure 6-7 Help portlet

The visibility of the help area is controlled by console preferences.

Setting console preferences

The look of the administrative console can be altered by setting console preferences. The preference you see vary slightly depending on the console type. For example, the preference to synchronize changes with nodes is only applicable to an administrative console on a deployment manager. Figure 6-8 displays the console preferences for a deployment manager's administrative console.

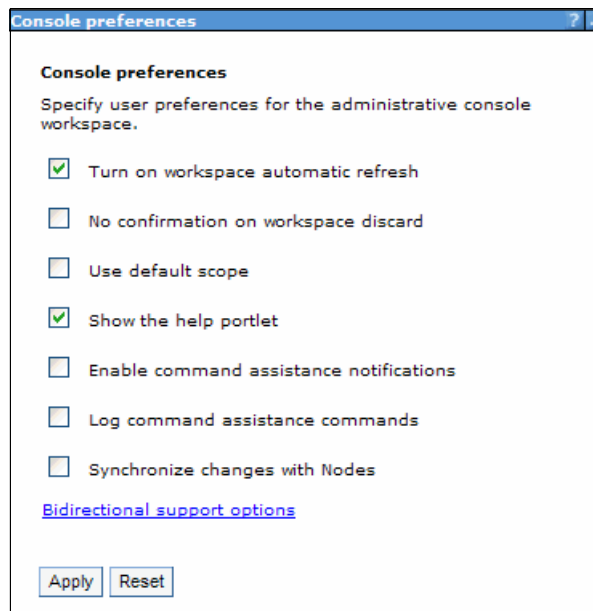


Figure 6-8 Console preferences

To set administrative console preferences, click **System administration** → **Console Preferences** in the navigation tree. You have the following options:

- ▶ Turn on WorkSpace Auto-Refresh specifies that the view automatically refreshes after a configuration change. If it is not selected, you must re-access the page to see the changes.
- ▶ No Confirmation on Workspace Discard specifies that a confirmation window be displayed if you elect to discard the workspace. For example, if you have unsaved changes and log out of the administrative console, you are asked whether you want to save or discard the changes. If this option is not selected, and you elect to discard your changes, you are asked to confirm the discard action.
- ▶ Use default scope (administrative console node) sets the default scope to the node of the administration console. If you do not enable this setting, the default is all scopes.
- ▶ Show the help portlet displays the help portlet to the right of the administrative console.
- ▶ Enable command assistance notifications allows you to send JMX notifications that contain command data. These notifications can be monitored in a Rational Application Developer workspace, providing assistance in creating scripts.
- ▶ Log command assistance commands specifies whether to log all of the command assistance wsadmin data for the current user.

When you select this option, script commands matching actions you take in the administrative console are logged to the following location:

```
profile_root/logs/<server_name>/commandAssistanceJythonCommands_<user_name>.log
```

- ▶ Synchronize changes with Nodes synchronizes changes that are saved to the deployment manager profile with all the nodes that are running.

Select the boxes to choose which preferences you want to enable and then click **Apply**.

Using the bidirectional support options link you can specify bidirectional text preferences for the administrative console. Text is supported in both directions for different types of alphabets. This means that the path hierarchy is displayed left-to-right even if elements of the path have right-to-left text. You can enable Global Preferences, which enables this option for all users and also selects the Current User Preferences option (see Figure 6-9). If you only want to enable this support for the current user logged into the administrative console, enable only the Current User Preferences.

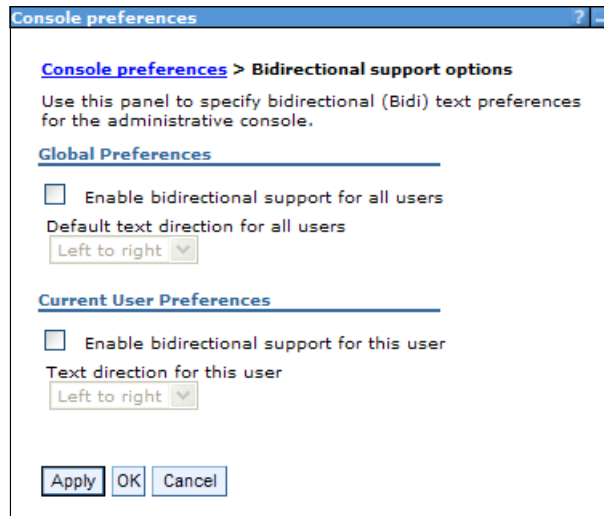


Figure 6-9 Bidirectional support options

6.1.5 Administrative console resources scopes

When working with items in the administrative console, certain resources are defined at a scope level. If applicable, you can select the scope from the drop-down menu, and you can set the preferences to specify how the information is to be displayed on the page.

For example, to display WebSphere Variables that were defined, click **Environment** → **WebSphere Variables**. The window shown in Figure 6-10 on page 199 opens.

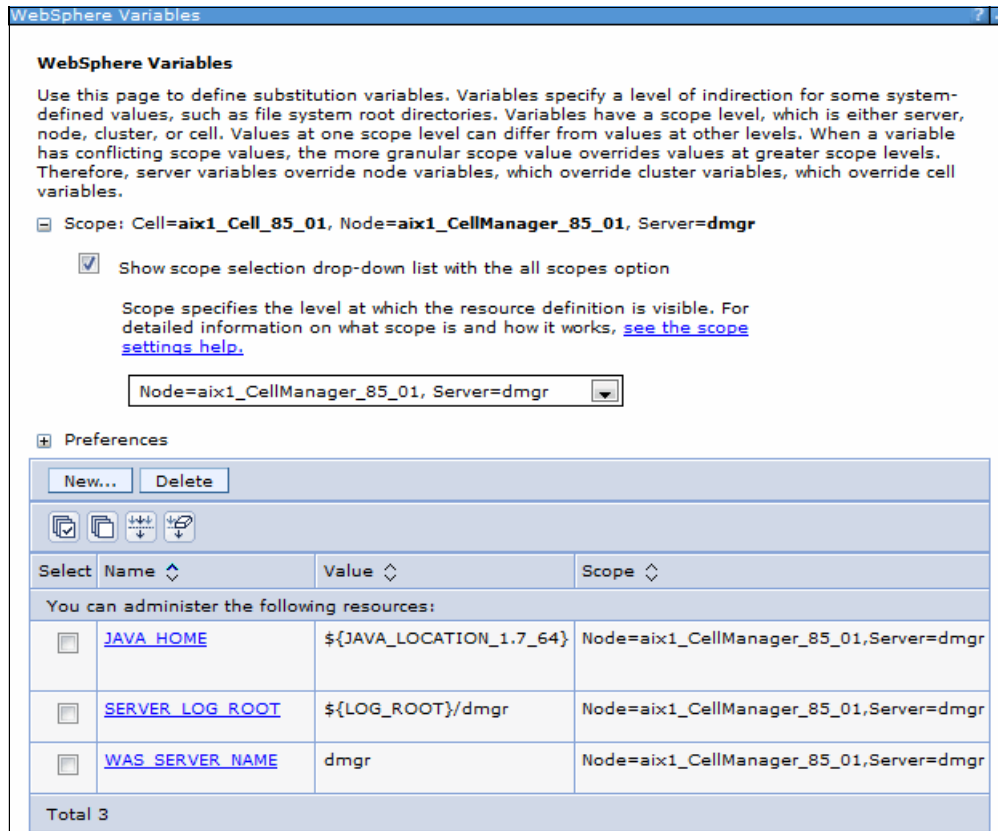


Figure 6-10 Administrative console scope area

Selecting a scope

The scope level determines which applications or application servers see and use that configuration. The scope setting is available for all resource types, WebSphere variables, shared libraries, and name space bindings.

Scope levels

Configuration information is defined at the following levels: cell, cluster, node, server, and application. Here, we list these scopes in overriding sequence. Because you see application scope first, anything defined at this scope overrides any conflicting configuration you might find in the higher-level scopes:

1. Resources and variables scoped at the *application* level apply only to that application. Resources and variables are scoped at the application level by defining them in an enhanced EAR.
2. Resources scoped at the *server* level apply only to that server. If a node and server combination is specified, the scope is set to that server. Shared libraries configured in an enhance EAR are automatically scoped at the server level.
3. Resources scoped at the *node* level apply to all servers on the node.
4. Resources scoped at the *cluster* level apply to all application servers in the cluster. New cluster members automatically have access to resources scoped at this level. If you do not have any clusters defined, you will not see this option.
5. Resources scoped at the *cell* level apply to all nodes and servers in the cell.

Stand-alone application servers: Although the concept of cells and nodes is more relevant in a managed server environment, scope is also set when working with stand-alone application servers. Because there is only one cell, node, and application server, and no clusters, simply let the scope default to the node level.

Configuration information is stored in the repository directory that corresponds to the scope. For example, if you scope a resource at the node level, the configuration information for that resource is in:

```
<profile_home>/config/cells/cell_name/nodes/<node>/resources.xml
```

If you scoped that same resource at the cell level, the configuration information for that resource is in:

```
<profile_home>/config/cells/cell_name/resources.xml
```

Setting scope levels in the administrative console

Collection pages that contain items that require a scope level to be identified provide two different options for defining the scope. Setting the scope level sets the level for any resources that you create and limits what is displayed in the collection page.

Clicking the **Show scope selection drop-down list with the all scopes** option provides a drop-down box with all scopes from which you can select, including the “All scopes” option, as shown in Figure 6-11. Selecting a scope from the drop-down list changes the scope automatically.

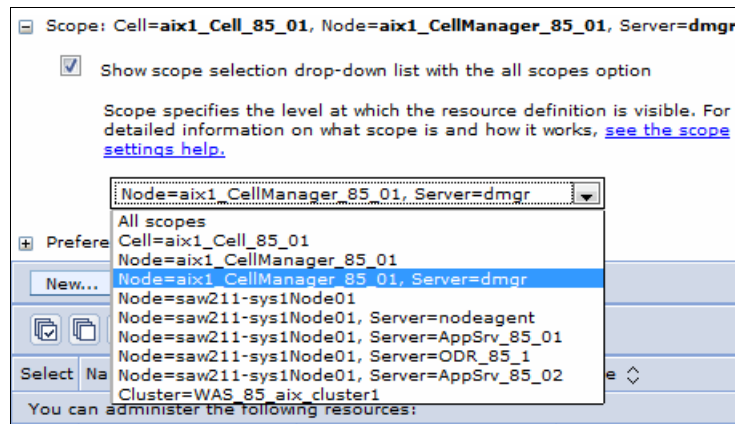


Figure 6-11 Scope level selection

The second option for setting the scope is to clear *Show scope selection drop-down list with the all scopes*. Instead of a drop-down menu, you have fields for each scope level where you can browse a list of applicable entries at that scope level. Click **Apply** to complete the selection, as shown in Figure 6-12 on page 201.

Scope: Cell=aix1_Cell_85_01, Node=aix1_CellManager_85_01, Server=dmgr

Show scope selection drop-down list with the all scopes option

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, [see the scope settings help.](#)

Cell
aix1_Cell_85_01

Node Cluster
aix1_CellManager_8! Browse Nodes Browse Clusters

→ Server
dmgr Browse Servers

Apply

Figure 6-12 Selecting the scope with individual fields

The scope is set to the lowest level entry you select (a red arrow to the left of the field indicates the current scope). To move to a higher scope, simply clear the lower field. For example, if you select a server as the scope level and want to change the scope to the node level, clear the server field, and click **Apply**.

This option is useful in cells that contain a large number of nodes, servers, or clusters. In those situations, the drop-down menu can be difficult to navigate. However, note that the option to view all scopes is not available.

Setting preferences for viewing the administrative console page

After selecting a task and a scope, the administrative console page shows a collection table with all of the objects created at that particular scope. You can change the list of items you see in this table by using the filter and preference settings.

The preference settings that are available vary by the type of item you are displaying. A list of the preference settings and their use is available in the information center, at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/rcon_preferences.html

Figure 6-13 on page 202 illustrates the preference settings that you see when displaying a list of JDBC providers.

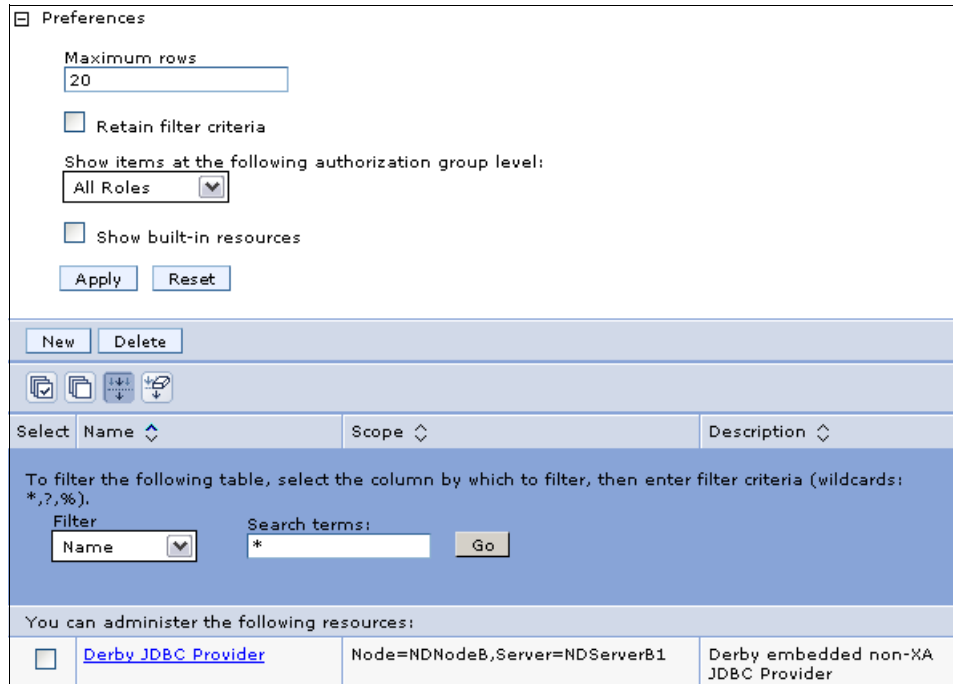



Figure 6-13 Filter and preference settings

The filter options can be displayed or set by clicking the **Show Filter Function** icon () at the top of the table, as shown in Figure 6-14.

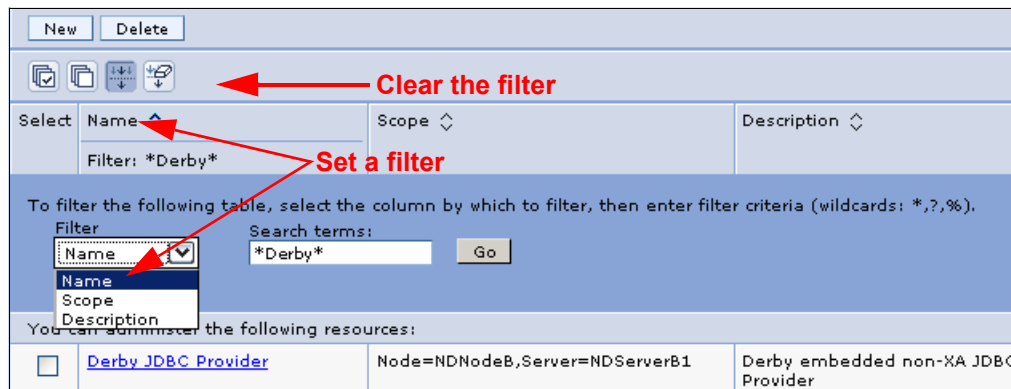



Figure 6-14 Setting filters and preferences

When you click the Show Filter Function icon, a new area appears at the top of the table, allowing you to enter filter criteria. To filter items:

1. Select the column to filter on, for example, in Figure 6-14, the display table has three columns from which to choose. Your options vary depending on the type of item you are filtering.
2. Enter the filter criteria. The filter criteria is case sensitive and wild cards can be used. In our example, to see only providers with names starting with “S”, select the **Name column** to filter on, and enter S* as the filter.
3. Click **Go**.
4. After you set the filter, click the **Show Filter** icon again to remove the filter criteria from the view. You still have a visual indication that the filter is set at the top of the table.

Setting the filter is temporary and only lasts for as long as you are in that collection. To keep the filter active for that collection, select the **Retain filter criteria** box in the Preferences section, and click **Apply**. To clear the filter criteria, click the  icon.

For more help about using the filtering feature, see the Administrative console buttons section at the following information center website:

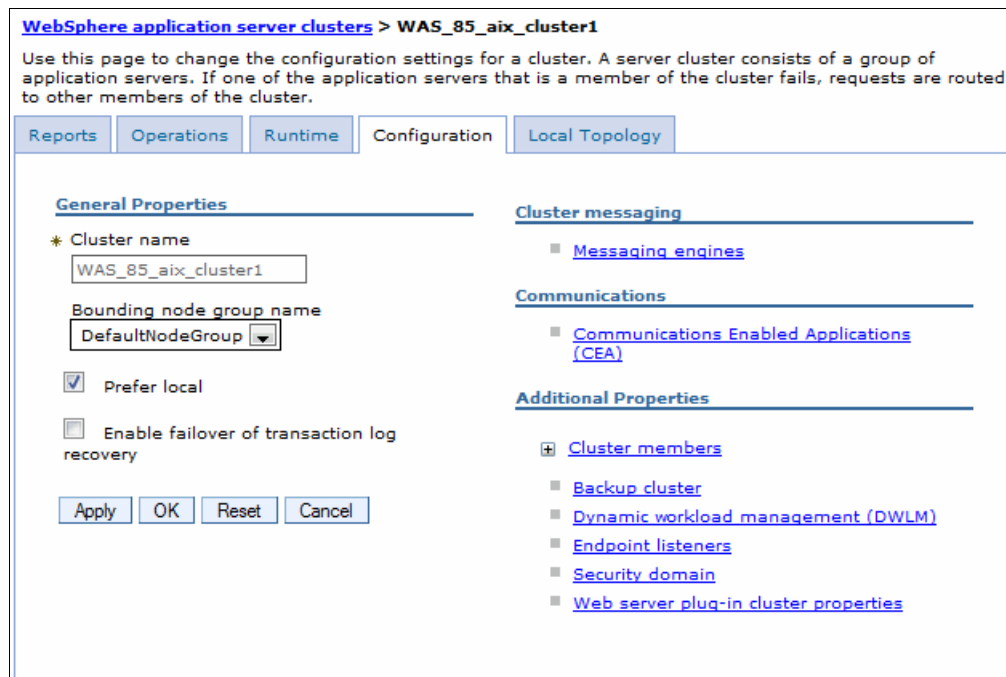
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/rcon_buttons.html

6.1.6 Updating existing items

To edit the properties of an existing item:

1. Select the category and type in the navigation tree, for example, select **Servers** → **Server Types** → **WebSphere application servers**.
2. A list of the items of that type, in the scope specified, are listed in a collection table in the workspace area. Click an item in the table. This action opens a detail page for the item.
3. In some cases, you see a Configuration tab and a Runtime tab on this page. In others, you only see a Configuration tab.

Updates occur under the Configuration tab. Specify new properties or edit the properties already configured for that item. The configurable properties depend on the type of item selected. For example, if you select a WebSphere Application Server cluster, this action opens a detail page resembling Figure 6-15.



The screenshot displays the configuration page for a WebSphere application server cluster. The breadcrumb path is "WebSphere application server clusters > WAS_85_aix_cluster1". A descriptive text states: "Use this page to change the configuration settings for a cluster. A server cluster consists of a group of application servers. If one of the application servers that is a member of the cluster fails, requests are routed to other members of the cluster." The page features five tabs: Reports, Operations, Runtime, Configuration (selected), and Local Topology. The main content area is divided into four sections: "General Properties" with fields for "Cluster name" (WAS_85_aix_cluster1) and "Bounding node group name" (DefaultNodeGroup), and checkboxes for "Prefer local" (checked) and "Enable failover of transaction log recovery" (unchecked); "Cluster messaging" with a link for "Messaging engines"; "Communications" with a link for "Communications Enabled Applications (CEA)"; and "Additional Properties" with a plus sign icon and links for "Cluster members", "Backup cluster", "Dynamic workload management (DWLM)", "Endpoint listeners", "Security domain", and "Web server plug-in cluster properties". At the bottom left, there are buttons for "Apply", "OK", "Reset", and "Cancel".

Figure 6-15 Editing an application server cluster properties

A Local Topology tab is sometimes displayed and shows the topology that is currently in use for this administrative object.

The detail page provides fields for configuring or viewing the more common settings and links to configuration pages for additional settings.

4. Click **OK** to save your changes to the workspace and exit the page. Click **Apply** to save the changes without exiting. The changes are still temporary. They are only saved to the workspace and not to the master configuration. Those changes still need to be done.
5. As soon as you save changes to your workspace, you will see a message in the Messages area reminding you that you have unsaved changes, as shown in Figure 6-16.

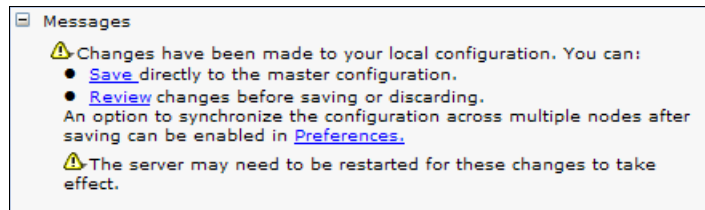


Figure 6-16 Save message

At intervals during your configuration work and at the end, save the changes to the master configuration by clicking **Save** in the Messages area or by clicking **System administration** → **Save Changes to master repository** in the navigation tree.

To discard changes, use the same options. These options simply display the changes you made and give you the opportunity to save or discard.

6.1.7 Adding new items

To create new instances of most item types:

1. Select the category, and type in the navigation tree.
2. Click **Scope**. (When creating a new item, you cannot click the **All** option for scope.)
3. Click the **New** button above the collection table in the workspace.

When you click **New** to add an item, one of two things occur, depending on the type of item you are creating. A wizard guides you through the definitions, or a new details page opens allowing you to fill in the basic details. In the latter case, enter the required information, and click **Apply**. This action usually activates additional links to detail pages that are required to complete the configuration.

Note: In the configuration pages, you can click **Apply** or **OK** to store your changes in the workspace. If you click **OK**, you exit the configuration page. If you click **Apply**, you remain in the configuration page. As you are becoming familiar with the configuration pages, always click **Apply** first. If there are additional properties to configure, you will not see them if you click **OK** and leave the page.

4. Click **Save** in the task bar or in the Messages area when you are finished.

6.1.8 Removing items

To remove an item:

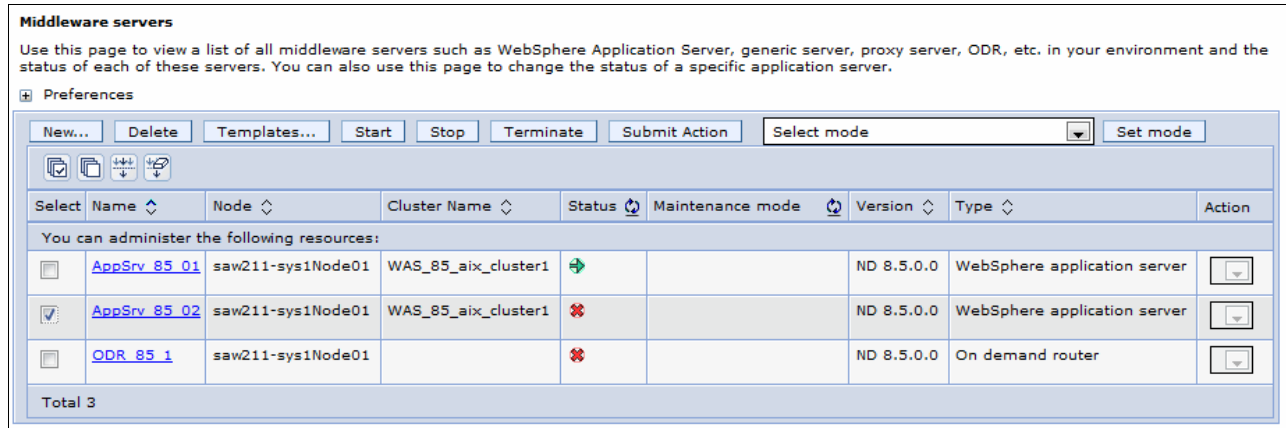
1. Find the item.
2. Select the item in the collection table by selecting the box next to it.
3. Click the **Delete** button above the collection table in the workspace.
4. If asked whether you want to delete it, click **OK**.
5. Click **Save** in the Messages area when you are finished.

6.1.9 Starting and stopping items

To start or stop an item using the administrative console:

1. Select the category and type in the navigation tree.
2. Select the item in the collection table by selecting the **box** next to it.
3. Click **Start** or **Stop**. The collection table shows the status of the item, as shown in Figure 6-17.

For example, to start a specific application server in a distributed server environment, click **Servers** → **All Servers**. Select the **box** beside the resource that you want, and click **Start**.



The screenshot shows the 'Middleware servers' administrative console. At the top, there are buttons for 'New...', 'Delete', 'Templates...', 'Start', 'Stop', 'Terminate', 'Submit Action', and a 'Select mode' dropdown menu. Below these is a toolbar with icons for refresh, print, and other actions. The main area contains a table with the following columns: 'Select', 'Name', 'Node', 'Cluster Name', 'Status', 'Maintenance mode', 'Version', 'Type', and 'Action'. The table lists three resources:

| Select | Name | Node | Cluster Name | Status | Maintenance mode | Version | Type | Action |
|-------------------------------------|------------------------------|-------------------|---------------------|--------|------------------|------------|------------------------------|----------------------------------|
| <input type="checkbox"/> | AppSrv_85_01 | saw211-sys1Node01 | WAS_85_aix_cluster1 | + | | ND 8.5.0.0 | WebSphere application server | <input type="button" value="v"/> |
| <input checked="" type="checkbox"/> | AppSrv_85_02 | saw211-sys1Node01 | WAS_85_aix_cluster1 | ✘ | | ND 8.5.0.0 | WebSphere application server | <input type="button" value="v"/> |
| <input type="checkbox"/> | ODR_85_1 | saw211-sys1Node01 | | ✘ | | ND 8.5.0.0 | On demand router | <input type="button" value="v"/> |

Total 3

Figure 6-17 Starting and stopping a server

Not all items can be started and stopped from the administrative console. For example, the deployment manager must be started independently from the administrative console. Also, there can be multiple options for starting and stopping an item (restart, stop immediate, and so on.) These options are described in Chapter 7, “Administration of WebSphere processes” on page 233.

6.1.10 Using variables

WebSphere variables are name and value pairs used to represent variables in the configuration files, which makes it easier to manage a large configuration. Predefined variables, such as `JAVA_HOME`, `SERVER_LOG_ROOT`, `WAS_SERVER_NAME`, can be found here using specific scope selections. It is important to set a variable to the required scope level to use it properly.

To set a WebSphere variable:

1. Select **Environment** → **WebSphere variables**, as shown in Figure 6-18 on page 206.

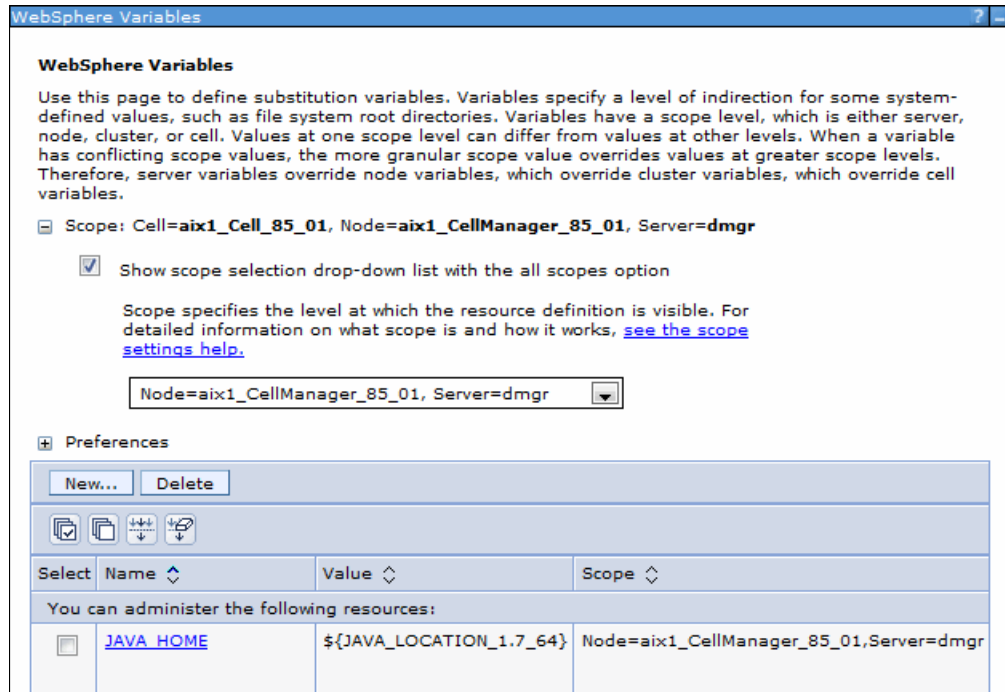


Figure 6-18 WebSphere variables

- To add a new variable, select the desired scope and then click **New**, or click a variable name to update its properties.
- Enter a name and value and then click **Apply**, as shown in Figure 6-19. A detailed description might help you identify the correct variable in a large environment.

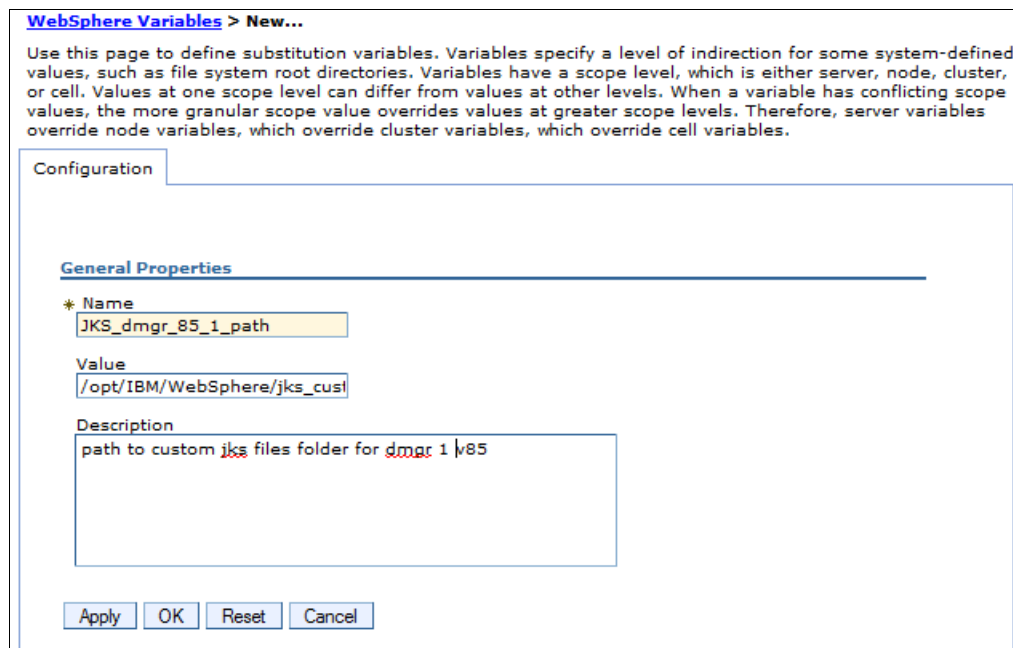


Figure 6-19 New variable

6.1.11 Saving work

As you work with the configuration, your changes are saved to temporary workspace storage. For the configuration changes to take effect, they must be saved to the master configuration. If you have a Network Deployment environment, a second step is required to *synchronize*, or send, the configuration to the nodes. Consider the following possibilities.

If you work on a page, and click **Apply** or **OK**, the changes are saved in the workspace under your user ID. Using this action, you can recover changes under the same user ID if you exit the session without saving.

You need to save changes to the master repository to make them permanent. You have several options:

- ▶ Use the Save window in the Messages area. If it is open, it is the quickest method.
- ▶ Click **System administration** → **Save Changes to master repository**.
- ▶ When you log in, if you logged out without saving the changes, you are given the option to save the changes.

The Save window presents you with the following options:

- ▶ Save.
- ▶ Discard: This option reverses any changes made during the working session and reverts to the master configuration.
- ▶ Cancel: This option does not reverse changes made during the working session. It just cancels the action of saving to the master repository for now.
- ▶ Synchronize changes with nodes: This action distributes the new configuration to the nodes in a distributed server environment.

Before deciding whether you want to save or discard changes, you can see the changed items by expanding **Total changed documents** in the Save window.

Important: All the changes made during a session are cumulative. Therefore, when you decide to save changes to the master repository, all changes are committed. There is no way to be selective about what changes are saved to the master repository.

6.1.12 Getting help

Help is available to you in several ways:

- ▶ Click **Help** on the administrative console banner. This action opens a new web browser or web browser tab with help for the administrative console. It is structured by administrative tasks, as shown in Figure 6-20 on page 208.

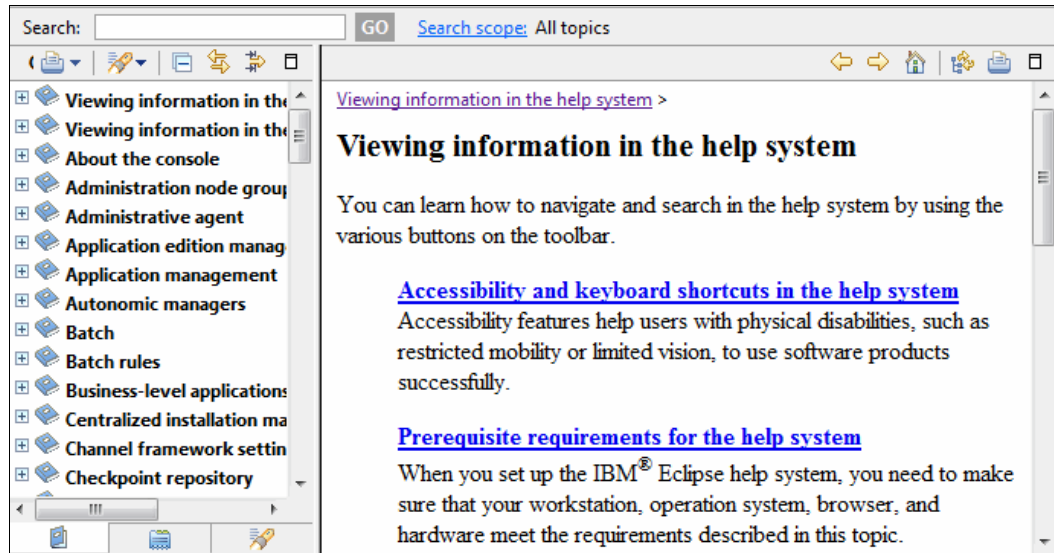


Figure 6-20 Administrative console help

- ▶ With the option **Show the help portlet** enabled, you can see the Help window in the workspace. Click **More information about this page** to open the help system to a topic-specific page.
- ▶ The information center can be viewed online or downloaded from the following website:
<http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp>

The help in the administrative console banner provides a searchable index for administrative tasks. There are a number of icon-based help control functions that allow you to navigate the help area.

6.1.13 New options in version 8.5 deployment manager administrative console

WebSphere Application Server V8.5 introduces new menus and tasks in the administrative console. All of these menus and tasks are discussed in corresponding chapters.

This section contains a list of the new tasks and a brief description about each.

Guided activities task

The deployment manager administrative console for IBM WebSphere Application Server V8.5 contains new guided activities that can help you prepare your environment:

- ▶ Preparing the hosting environment for basic dynamic operations helps you prepare support for WebSphere dynamic operations. This task can help create an On Demand Router, create a dynamic cluster, enable email notification for runtime tasks, and save and synchronize the changes.
- ▶ Deploying an application with defined service levels helps you to deploy an application with defined service levels into the WebSphere Extended Deployment hosting environment. It does so by installing the application, defining service levels with service policies, classifying application requests with service policy work classes, and saving and synchronizing the changes.
- ▶ Defining policies to detect and manage health conditions helps you to plan for detecting and managing health conditions in your environment. It does so by creating policies for

specific health conditions, configuring the health monitoring controller, setting email notifications for runtime tasks, and saving and synchronizing the changes.

Servers task

The deployment manager administrative console for IBM WebSphere Application Server V8.5 contains the following new features in the Servers task:

- ▶ You can view a list of all servers by selecting **Servers** → **All servers**
- ▶ In Server Types, there are the following new links:
 - On Demand Routers helps you manage your on demand routers by viewing, adding, deleting, using templates, and starting and stopping them.
 - PHP servers helps you view, add, delete, use templates, start, stop, terminate, submit action, and set the mode of PHP servers in your environment.
 - WebSphere Application Server Community Edition servers helps you view, add, delete, use templates, start, stop, terminate, submit action, and set the mode of WebSphere Application Server Community Edition servers in your environment.
 - Apache servers helps you view, add, delete, use templates, start, stop, terminate, submit action, and set the mode of Apache servers in your environment.
 - Custom HTTP servers helps you view, add, delete, use templates, start, stop, terminate, submit action, and set the mode of custom HTTP servers in your environment.
- ▶ In Clusters there are the following new features:
 - On demand Router Clusters lets you work with your groups of On Demand Router Clusters.
 - Dynamic clusters lets you view, add, delete, and set the mode of your dynamic clusters.

The Version 5 JMS servers link is no longer available in the **Servers** → **Servers Types** task.

Applications task

In Applications task, there are the following new features:

- ▶ You can view all your applications by selecting the **All applications** link.
- ▶ Using the Install New Middleware Application link, you can add Java 2 Platform enterprise Edition, PHP, Unmanaged Web Application, and WebSphere Application Server Community Edition types of applications to your environment.
- ▶ Edition Control Center enables management and operational control over application editions, including interruption free application upgrade. An application edition is a version of an application composed of distinct versions of modules and bindings. It provides a summary view of each enterprise application, its editions, and their current state. By clicking an enterprise application name, you can manage the individual editions of the selected application. More information about application editions management is available in Chapter 13, “Intelligent management” on page 469.
- ▶ Using the Global deployment settings link you can manage settings that apply to all applications.

Runtime Operations task

Runtime operations are used to configure the dynamic operations environment and use visualization capabilities to understand the operational state of the environment. You can find more information about this task in the 16.5, “Monitoring operations” on page 584.

This task consists of several features:

- ▶ **Dashboard:** Displays a high level of the overall environment and alerts about any possible problems.
- ▶ **Applications:** Displays an operational summary of all the started applications in your environment, including status, stability, and service policy.
- ▶ **Deployment Targets:** Displays an operational summary of the running deployment targets in your environment, such as application servers, middleware servers, clusters, and dynamic clusters.
- ▶ **Service Policies:** Displays performance data and is used to determine the relative performance of your service policies to the defined service policy goals.
- ▶ **Component Stability:** Displays runtime information and is used to review the information for all of the on demand routers in your environment.
- ▶ **Reports:** Displays customized charting to determine if you are meeting your business and performance goals. You can track statistics on various components of your environment.

Operational Policies task

This new task is used to define service policies, visualize service policies topology, define health policies, define custom actions, and manage autonomic managers. You can find more information about this task in the 13.1, “Introduction to Intelligent Management” on page 470.

System administration task

This task contains the following new features:

- ▶ **Extended Repository Service** enables advanced management of the configuration repository. The configuration repository contains the configuration for the cell. This information is essential to the operation of your applications. You can create repository checkpoints to help you save snapshots of your configuration as you make changes, so you can easily undo those changes if necessary. You can configure your repository to create automatic delta checkpoints each time you make a configuration change. A delta checkpoint saves a copy of the configuration documents prior to saving your changes. You can specify the number of automatic checkpoints to save. After this limit is reached, the next checkpoint replaces the oldest.
- ▶ **Middleware nodes** is used to manage nodes in the application server environment. A node corresponds to a physical computer system with a distinct IP host address. You can view a table with the managed and unmanaged nodes in this cell, add new nodes to the cell and to this list by selecting the add node administrative action.
- ▶ **Middleware descriptors** provides information about different middleware server platforms to the Intelligent Management runtime environment. The default middleware platforms are:
 - apacheWebServerRuntime
 - apache_server
 - application_server
 - customhttp_server
 - phpRuntime
 - wasceRuntime
 - wasce_server.
- ▶ **Visualization Data Services** provides performance statistics for the managed cell.
- ▶ **Target Management** → **Notifications** is used when sending email notifications of tasks.
- ▶ **Target Management** → **Runtime Tasks** shows the current tasks that are generated by a runtime component within Intelligent Management. Click the **Task ID** to view the task target objects and corresponding monitors of a specific task. To act on a task, choose the

action from the appropriate list and select the corresponding check **box**. Click **Submit**. You can submit multiple actions concurrently.

6.2 Securing the administrative console

WebSphere Application Server provides the ability to secure the administrative consoles so that only authenticated users can use them by enabling administrative security. Administrative security determines whether security is used at all, provides authentication of users using the WebSphere administrative function, the type of registry against which authentication takes place, and other values. Enabling administrative security activates the settings that protects your server from unauthorized users. Note that enabling administrative security does not enable application security.

Before enabling any type of security for a production system, familiarize yourself with WebSphere security and have a plan for securing your WebSphere environment. Security encompasses many components, including administrative security, application security, infrastructure security, and specialized resource security options. This section only provides an overview of administrative security.

The first decision you have to make is to select the user registry you will use. If you enable security when you create a profile for distributed systems, a file-based registry is automatically created and populated with one administrative user ID. On z/OS platforms, you have the option of using the file-based registry or the z/OS system's SAF-compliant security database.

Though a file-based user registry is not a best practice for securing applications, you can federate additional registries to the existing file-based registry to manage users and groups for application security.

If you are using a registry other than the WebSphere Application Server federated user registry, you must create at least one user ID to be used for the WebSphere administrator.

Although you might have heard about security domains that were introduced in WebSphere Application Server V7, these domains are used for application security (not administrative security).

Before implementing security in a production environment, be sure to consult *WebSphere Application Server V8 Security Guide*, SG24-7971.

6.2.1 Enabling security after profile creation

You can enable administrative security after profile creation through the administrative console by navigating to **Security** → **Global security**. Performing this action allows you more flexibility in specifying security options. You must complete the configuration items for authentication, authorization, and realm (user registry). Populate the chosen user registry with at least one user ID to be used as an administrator ID.

You can use the Security Configuration Wizard in the Security settings page that assists you in securing your environment. To do this, click the **Security Configuration Wizard** button. Click **Next** through the various windows of the wizard. The steps that you need to complete are:

1. In the first step, select whether to enable application security or if you need to use Java2 security to restrict application access to local resources. Be aware that when you select to enable administrative security, the application security check box is enabled automatically.

If you are not prepared to use application security at this time, be sure to clear the box. Java 2 security can be selected at this point or any time after enabling the administrative security.

2. In the second step, select the type of user registry that you need for your environment:
 - Federated repositories: Manage identities that are stored in multiple repositories in a single, virtual realm.
 - Standalone LDAP Server: Uses the Lightweight Directory Access Protocol (LDAP) user registry settings. Select this option in case your users and groups reside in an external LDAP registry
 - Local operating system: Uses the local operating system user registry of the application server.
 - Standalone custom registry: Specifies a custom registry that implements the UserRegistry interface in the `com.ibm.websphere.security` package.
3. In the third step, select the primary administrative user name and other options depending on the previous option selected. In Figure 6-21 on page 213, we use federated repositories, which requires the password for the primary administrative user to be specified and confirmed.
4. The last step summarizes your selected options. Click **Finish** and the **Save** the changes.

Figure 6-21 on page 213 illustrates the security settings page that is displayed after completing the steps.

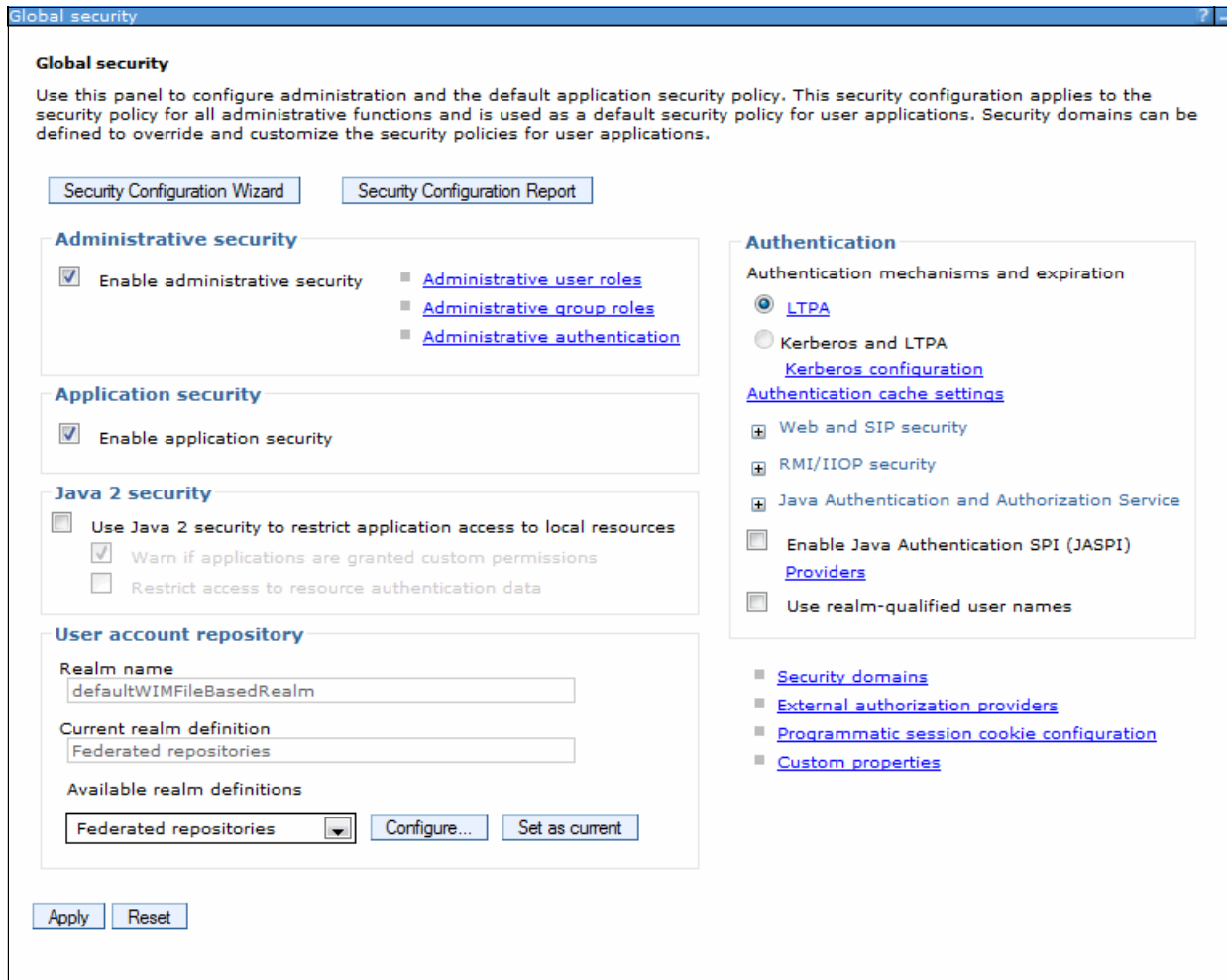


Figure 6-21 Security settings page

Restart your deployment manager server process to be able to login to the administrative console using the user name and password that you selected.

In larger environments, you can use an LDAP server for your user repository. Using WebSphere Application Server V8.5 you can connect to the following LDAP server types:

- ▶ IBM Tivoli Directory Server
- ▶ z/OS Integrated Security Services LDAP Server
- ▶ IBM Lotus® Domino®
- ▶ Novell Directory Services
- ▶ Sun Java System Directory Server
- ▶ Microsoft Windows Active Directory
- ▶ Microsoft Active Directory Application Mode
- ▶ Custom

For detailed LDAP servers support, requirements, configuration steps, and general security information, refer to the information center at the following website:

<http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/welc6topsecuring.html>

To add a connection to an LDAP server, after you enabled the administrative security and set the user repository to federated repositories:

1. In the Java2 security section of the security settings window, click **Configure**.
2. In the Federated repositories window, click **Add repositories (LDAP, custom, etc)**.
3. In the Repository reference window, click **New Repository**.
4. Select **LDAP Repository** from the drop-down list.
5. Add all of the necessary information in the General Properties window according to your LDAP environment.
6. Click **Apply** and then click **Save**.

TIP: If you enable administrative security, and then find that you cannot log in, you can disable security through scripting or manually editing the `security.xml` profile. This action allows you to go back through the security configuration to see what is causing the problem.

Editing an XML configuration file manually is not a best practice. Use scripting to enable or disable administrative or application security and to modify other security settings. To disable administrative security through scripting:

1. Navigate to the `dmgr_profile_home/bin` directory.
2. Start the `wsadmin` scripting client with the `-conntype none` argument.
3. Enter the `securityoff` command in JACL mode or `securityoff ()` command in Jython mode.
4. Exit the `wsadmin` scripting client.
5. Restart your processes.

When starting the `wsadmin` scripting client with the `-conntype none` argument, the `securityoff` command toggles the `enabled="true"` setting in `security.xml` to `enabled="false"`. The `wsadmin` session is in local mode and in this case acts as a text editor to make the needed configuration change.

To manually edit the `security.xml` file:

1. Open the `security.xml` file at `dmgr_profile_home/config/cells/cell_name`.
2. Edit the second line, changing `enabled="true"` to `enabled="false"`. This process is shown in Example 6-4.

Example 6-4 Manually editing the security.xml file

```
<security:Security xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:orb.securityprotocol="http://www.ibm.com/websphere/appserver/schemas/5.0/
orb.securityprotocol.xmi"
xmlns:security="http://www.ibm.com/websphere/appserver/schemas/5.0/security.xmi
" xmi:id="Security_1" useLocalSecurityServer="true"
useDomainQualifiedUserNames="false" enabled="false" cacheTimeout="600"
issuePermissionWarning="true" activeProtocol="BOTH"
enforceJava2Security="false" enforceFineGrainedJCASecurity="false"
appEnabled="true" dynamicallyUpdateSSLConfig="true" allowBasicAuth="true"
activeAuthMechanism="LTPA_1" activeUserRegistry="WIMUserRegistry_1"
defaultSSLSettings="SSLConfig_1">
```

If administrative security is enabled, each time you log in to the administrative console, you must authenticate with the user ID that was identified as having an administrative role. Entering commands from a command window also prompts you for a user ID and password. You can add additional administrative users and assign authorization levels from the administrative console.

6.2.2 Administrative security roles

Administrative security is based on identifying users or groups that are defined in the active user registry and assigning roles to each of those users. When you log into the administrative console or issue administrative commands, you must use a valid administrator user ID and password. The role of the user ID determines the administrative actions that the user can perform.

Fine-grained administrative security

Prior to WebSphere Application Server V6.1, users granted administrative roles and administered all of the resource instances under the cell. Starting with Version 6.1, administrative roles are now per resource instance rather than to the entire cell. Resources that require the same privileges are placed in a group called the *authorization group*. Users can be granted access to the authorization group by assigning to them the required administrative role within the group.

A cell-wide authorization group exists for backward compatibility. Users who are assigned to administrative roles in the cell-wide authorization group can still access all of the resources within the cell.

The following administrative security roles are available:

- ▶ **Administrator:** The administrator role has operator permissions, configurator permissions, and the permission required to access sensitive data, including server password, Lightweight Third Party Authentication (LTPA) password and keys, and so on.
- ▶ **Auditor:** The auditor role has permission to view and change the configuration settings for the security auditing subsystem.
- ▶ **Configurator:** The configurator role has monitor permissions and can change the WebSphere Application Server configuration.
- ▶ **Operator:** The operator role has monitor permissions and can change the runtime state, for example, the operator can start or stop services.
- ▶ **Monitor:** The monitor role has the least permissions. This role primarily confines the user to viewing the WebSphere Application Server configuration and current state.
- ▶ **Deployer:** The deployer role has permission to perform both configuration actions and runtime operations on applications.
- ▶ **Admin Security Manager:** The Admin Security Manager role gives permissions to users to map other users to administrative roles. When fine-grained administrative security is used, users granted this role can manage authorization groups. A user mapped to the administrator role does not have permissions to map users to administrative roles.
- ▶ **ISC Admin:** The ISC Admin role has administrator privileges for managing users and groups from within the administrative console only.

You can find more information about each of these roles at the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/rsec_adminroles.html

Assigning administrative roles to users and groups

If you are using a file-based repository, you can add users and groups through the console by clicking **Users and Groups** → **Manage Users** or **Users and Groups** → **Manage Groups**. Otherwise, the users and groups must be added to the user registry using the tools provided by the registry product.

Role assignments for users and groups are managed through the administrative console. Click **Users and Groups** → **Administrative user roles** or **Users and groups** → **Administrative group roles**. Use these windows to assign an administrative role to a user or group.

Fine-grained security

WebSphere Application Server administrative security is fine-grained, meaning that access can be granted to each user per resource instance. For example, users can be granted configurator access to a specific instance of a resource (an application, an application server, or a node). The administrative roles are assigned per resource instance rather than to the entire cell.

To achieve the instance-based security or fine-grained security, resources that require the same privileges are placed in a group called the administrative authorization group or authorization group. Users can be granted access to the authorization group by assigning to them the required administrative role.

You can define groups of resources that are treated collectively by clicking **Security** → **Administrative Authorization Groups**. The resource instances that are added to an authorization group can be the following types:

- ▶ Clusters
- ▶ Business level applications
- ▶ Assets
- ▶ Nodes including application servers and web servers
- ▶ Applications
- ▶ Node groups

After the authorization group is created, you can assign users or groups an administrative role for the authorization group.

Many administrative console pages have a preference setting that allows you to restrict the items that you can see to those that are valid for your authorization group level. The roles that you can choose from depend on the role of the user ID that logged into the administrative console.

6.3 Job manager console

The job manager console has many of the basic options that you find in the deployment manager's administrative console, including global security settings, the option to add users and groups to the federated user repository, WebSphere variable settings, and others that are common to any administrative environment. What is unique to the job manager administrative console is the ability to submit jobs to nodes registered to it.

Starting with WebSphere Application Server V8, you can complete job manager actions and run jobs from the deployment managers administrative console. The deployment manager administrative console has a Jobs navigation tree option that is similar to that in the job

manager console. The Jobs navigation tree in the job manager console has the following options:

- ▶ Submit a job
- ▶ Review the status of a job
- ▶ Identify job manager target for job
- ▶ Identify target resources used in job
- ▶ Identify target groups for administrative jobs
- ▶ Add or delete Installation Manager installation kits

Figure 6-22 shows a job manager administrative console. The option selected in the Navigation tree is **Jobs** → **Targets**. In Figure 6-22, there is a target already registered.

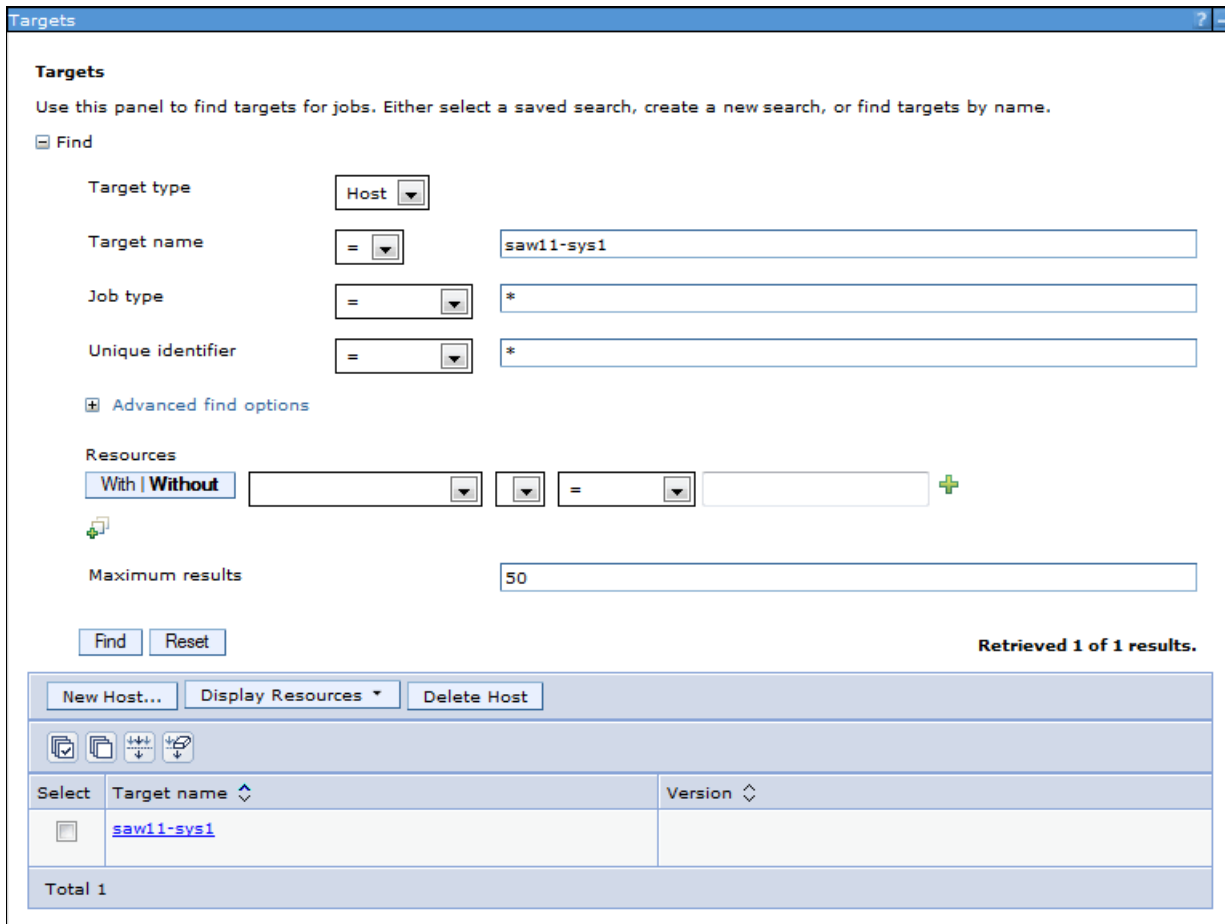


Figure 6-22 Job manager administrative console - List of targets

Groups of nodes: You can create groups of nodes that contain the nodes you will work with from the job manager (click **Jobs** → **Groups of nodes**). A group of nodes can be used as the target of administrative jobs.

When you submit a job, you can select one or more groups from a drop-down menu. The alternative is to type in the name of the node, or use the Find feature to select each node. Using the Find feature takes several steps.

So, even if you do not plan to use multiple nodes as the target of a job, creating a group for each node allows you to easily select a node rather than typing it in or searching for it.

If you include multiple nodes in the group, beware that all of the nodes must have a common user ID and password. When you submit a job, you only have one place where you can enter the user ID and password.

6.3.1 Submitting a job with the job manager

The job manager provides the following job types:

- ▶ Run a wsadmin script
- ▶ Manage applications:
 - distributeFile
 - collectFile
 - removeFile
 - startApplication
 - stopApplication
 - installApplication
 - updateApplication
 - uninstallApplication
- ▶ Manage servers:
 - createApplicationServer
 - deleteApplicationServer
 - createProxyServer
 - deleteProxyServer
 - createCluster
 - deleteCluster
 - createClusterMember
 - deleteClusterMember
 - configureProperties
- ▶ Manage the server run time:
 - startServer
 - stopServer
 - startCluster
 - stopCluster
- ▶ Submit Installation Manager jobs:
 - installIM
 - updateIM
 - manageOfferings
 - findIMDataLocation
- ▶ Submit Liberty profile job:
 - installLibertyProfileResources

- uninstallLibertyProfileResources
- startLibertyProfileServer
- stopLibertyProfileServer
- generateMergedPluginConfigForLibertyProfileServers

Details about each of these job types is at the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/xml_7jobtypes.html

Complete the following steps for an example of submitting a job:

1. Start the job manager, and log into the job manager console:
`http://<job_manager_host>:9960/ibm/console`
2. To submit jobs, nodes must already be registered with the job manager. To verify which nodes are registered, expand **Jobs** in the navigation window, and click **Targets**. If this is the first time you are using the job manager, you might not see all the nodes displayed. To refresh the view, enter * as the value for Node name, and click **Find**, as shown in Figure 6-22 on page 217.
3. Click **Jobs** → **Submit** to select the type of job to submit and then click **Next**, as shown in Figure 6-23.

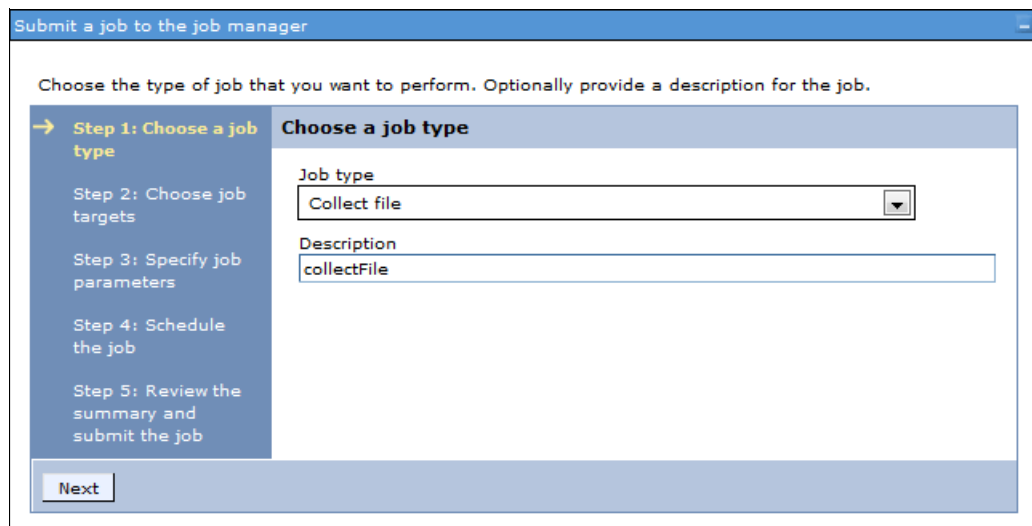


Figure 6-23 Select a job type

4. Select the node on which you want to run the job. You can select the node from a node group using the drop-down menu next to the Groups of nodes option, or you can select specific nodes using the Node names option. Enter the user ID and password for the node that you will run the job against, as shown in Figure 6-24 on page 220.

Choose job targets

Job type: Collect file

Target groups
-- No groups --

Target names

Add Find...

saw11-sys1

Remove

Target authentication

User name
root

Password authentication

* Password
.....

* Confirm password
.....

Figure 6-24 Provide a user name and a password for the target

To use a specific node, select **Target names** and either enter the node name and click **Add**, or click **Find**. Using the Find option opens a new window where you can search and select nodes, as shown in Figure 6-25.

Find targets

Set the find parameters to limit the search for targets. The results of the search are displayed in the chosen targets list that follows. Remove any targets from the chosen list that you do not want as job targets.

Find

Target type: All

Target name: =

Job type: =

Unique identifier: =

Advanced find options

Resources: With | Without

Maximum results: 50

Find Reset

Excluded targets

Chosen targets: saw11-sys1

OK Cancel

Figure 6-25 Search and select a target

The simplest method of searching is to enter an * in the Node name field, and click **Find**. The list of nodes is shown in the Excluded nodes box. Select the nodes you want, and use the arrow button to move them to the Chosen nodes box. Hold the Shift key down to select multiple nodes, or move them one at a time.

Click **OK**. This action returns you to step 2 of the wizard with the node name entered. Click **Next** to continue the job submit process.

5. Specify the job parameters. These parameters vary widely depending on the type of job. The parameters provide the additional information the job needs to perform the task. For example, if you are running a job to start a server, you selected the node in the previous step, but the server name must be entered as a parameter. You can also click **Find** to search for parameter, as shown in Figure 6-26.

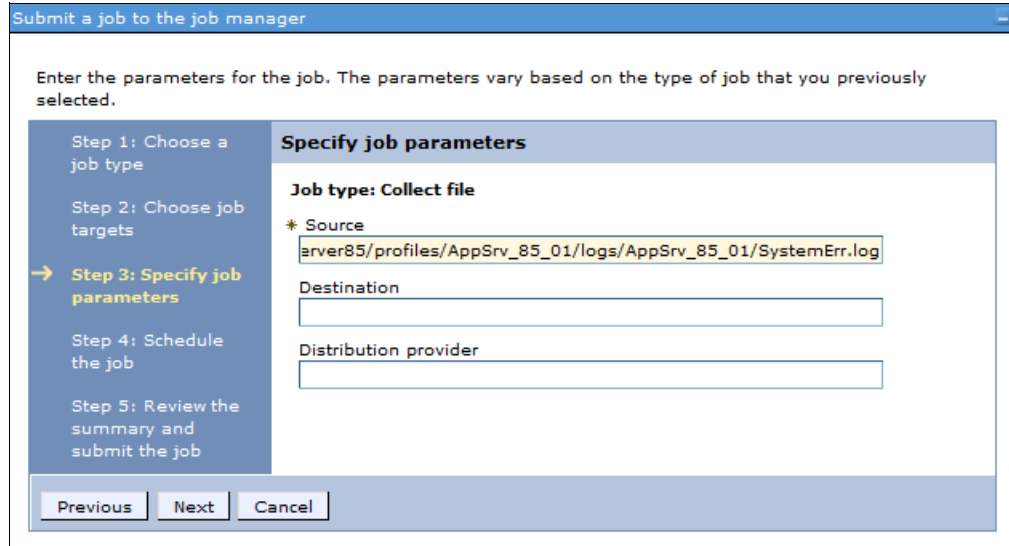


Figure 6-26 Specify job parameters

Click **Next**.

6. The next step contains fields that specify how and when the job runs and if a notification email is to be sent, as shown in Figure 6-27 on page 223.

Schedule the job

Job type: Collect file

Notification

Email addresses

Initial Availability

Specify when this job is first available.

Make the job available now.

Schedule availability

Date (MM/dd/yyyy) Time (HH:mm:ss)

/ / : :

Expiration

Specify when this job is no longer available.

Use default expiration - 1 days.

Expire the job based on a date

Date (MM/dd/yyyy) Time (HH:mm:ss)

/ / : :

Expire the job based on a duration

Expire after

Job Availability Interval

Jobs can run repeatedly based on an interval. Specify the interval that the job is available.

Availability interval

Figure 6-27 Specifying the job scheduling information

The fields, shown in Figure 6-27 on page 223, are:

- Notification: The email address specified receives a notification when the job is finished. To use this field, you must configure a mail provider and mail session.
- Initial availability: You can make the job available now (it will run immediately after you have finished with the job submission process), or you can specify a date and time it will be available.
- Expiration: Specify an expiration date for the job.
- Job availability interval: This field allows you to repeat job submission at intervals. Depending on the selection, you will have an additional field displayed that allows you to choose the days, start and stop time, and so on (see Figure 6-28).

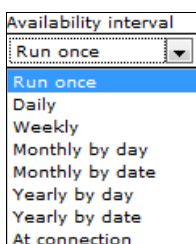


Figure 6-28 Choose the interval for the job submission

If you select **Make this job available now** and **Run once**, the job runs immediately and the Expiration settings have no meaning. The alternative is to set an Initial availability, Expiration date or duration, and select an interval at which the job will run.

7. Review the summary and submit the job. When a job is submitted from the job manager, the job details are saved in a database local to the job manager. The endpoint (deployment manager or administrative agent) pings the job manager at a predefined interval and fetches jobs that are to be executed. If the job submitted is a wsadmin job, the wsadmin script is executed. Otherwise, a corresponding job handler will execute the necessary admin code.
8. You can monitor the results through the Job status window. Click the **Refresh** button, in the Status summary column (🔄), to update the status. The color in the Status summary field indicates the success or failure of the job, as shown in Figure 6-29.

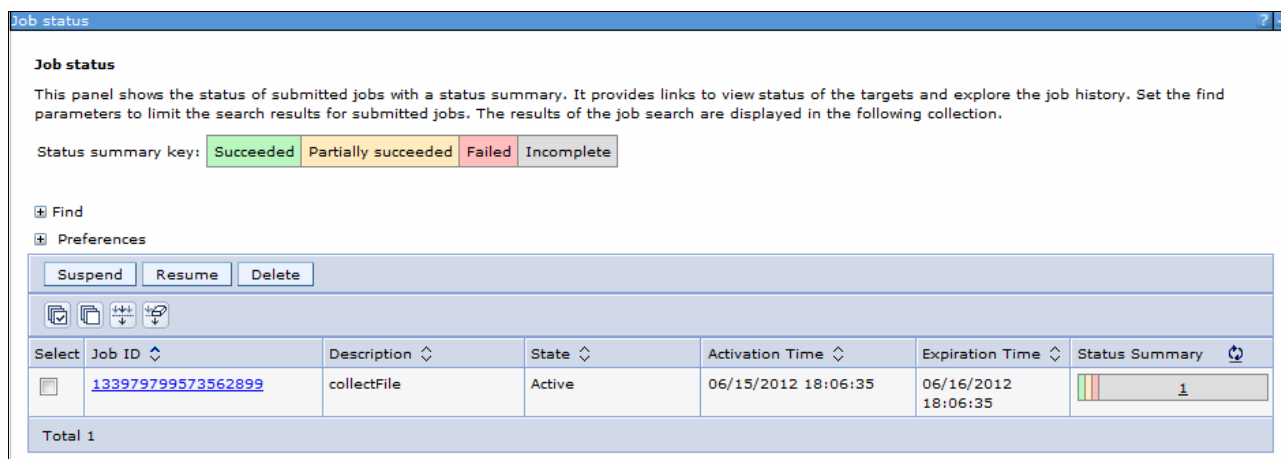


Figure 6-29 Job status

9. Click the **Job ID** to see more information about the job, as shown in Figure 6-30.



Figure 6-30 Job status details

The job status is always sent back to the job manager. Clicking the message in the Status column (Succeeded in this case) shows you additional information.

In the event of an error, you will see any messages produced by the job. Additional messages might be available in the logs for the server where the administrative action was to take place.

When you execute a configuration type job (for example, create server) from the job manager to a deployment manager, the configuration is saved if the job is successful. A job that submits a wsadmin script, however, does not save the configuration (the `wsadmin` script needs to do that).

Executing a job to a deployment manager does not cause node synchronization to occur. Synchronization happens at the next automatic synchronization interval, or a wsadmin script can be submitted to synchronize.

6.3.2 Distributing files using the job manager

Some job types require that files be transferred to the node where the job is to run. The Distribute file job type can be used to transfer these files, which is normally necessary in the following circumstances:

- ▶ When you want to run a wsadmin script on the node. The script must be distributed to the node before you can use the Run wsadmin script job.
- ▶ When you want to install or update an application. The EAR file must be distributed to the node before you can use the Install application or Update application jobs.

The following steps illustrate how to distribute a file to a node for use in later jobs. This example distributes a wsadmin script file to an admin agent:

1. The file to be distributed from the job manager must be in the /config/temp/JobManager directory of the job manager profile.

Create the `jobmgr_profile_root/config/temp/JobManager` directory, and copy the file into it.

If you are developing a script or application in Rational Application Developer, you can export the file directly to the directory.

2. The distribute file job stores the file into the `downloadedContent` directory of the administrative agent or deployment manager profile. The destination parameter is relative to the `downloadedContent` directory. You must create this directory on the admin agent or deployment manager:

- `adminagent_profile_home/downloadedContent`
- `dmgr_profile_root/downloadedContent`

3. In the Job manager administrative console, select the **Job** → **Submit** menu, which launches the Job properties wizard:

- a. Select **Distribute file** as the job type, and click **Next**.
- b. Enter the script file location on the job manager and the location to store the script file on the target node.

In this example, the `appInstall.py` script was stored in the following location:

`jobmgr_profile_root/config/temp/JobManager/appInstall.py`

On the admin agent, it is stored as:

`adminagent_profile_home/downloadedContent/appInstall.py`

The arguments are entered, as shown in Figure 6-31.

The screenshot shows a wizard window titled "Specify job parameters". On the left, a vertical sidebar lists five steps: "Step 1: Choose a job type", "Step 2: Choose job targets", "Step 3: Specify job parameters" (highlighted with a yellow arrow), "Step 4: Schedule the job", and "Step 5: Review the summary and submit the job". The main area of the wizard is titled "Specify job parameters" and contains the following fields:

- Job type:** Distribute file
- * Source:** file:/appInstall.py
- * Destination:** /appInstall.py
- Distribution provider:** (empty text box)

At the bottom of the wizard, there are three buttons: "Previous", "Next", and "Cancel".

Figure 6-31 File distribution parameters

Click **Next**.

- c. Take the defaults for the job schedule. The defaults execute the distribute file job once. Click **Next**.
- d. Click **Finish**. Monitor the status of the job and ensure it completes successfully.

6.4 Using command-line tools

WebSphere Application Server provides various administrative commands that can be run from a command line. These commands can be used for many administrative tasks, for example, to start, view, or stop a WebSphere process. Many commands have an equivalent GUI interface, either created specifically for the command, through an administrative console, or through a First Steps console. However, it is often convenient to simply enter these commands manually from a command line.

Examples of commands are:

- ▶ **startServer** to start a server process
- ▶ **stopServer** to stop a server process
- ▶ **serverStatus** to obtain the status of servers
- ▶ **startNode** to start the node agent process
- ▶ **registerNode** to register a node with the administrative agent
- ▶ **addNode** to add a node to a cell configuration
- ▶ **addNode -asExistingNode** to recover damaged nodes or move nodes to a new machine or a different operating system (new feature from WebSphere Application Server V8)
- ▶ **syncNode** to synchronize a node with the deployment agent
- ▶ **manageprofiles** to manage a profile, for example, to backup or restore a configuration of a profile
- ▶ **managesdk** to manage the software development kits that are available to a WebSphere Application Server installation
- ▶ **versionInfo** to get IBM WebSphere product installation status report

More information about the **addNode -asExistingNode** and **managesdk** commands are in 3.3, “Building systems with profiles” on page 64.

6.4.1 Command location

Command-line tools must be run on the system where the process you are entering the command for resides. They cannot operate on a remote server or node. To administer a remote server, use the administrative console or a wsadmin scripting client script.

For the most part, the commands exist in two places:

- ▶ *install_root/bin*

Commands entered from this location operate against the default profile unless you use the **-profileName** parameter to specify the profile.

- ▶ *profile_root/bin*

Commands entered from this location operate against the profile defined in *profile_root*.

6.4.2 Key usage parameters

The commands are consistent across platforms, though how you enter them, case sensitivity and the extension, varies.

Note: Parameter values that specify a server name, a node name, or a cell name are always case sensitive regardless of the operating system.

There are several commonly used parameters that are valid for every command:

- ▶ **-profileName** specifies the profile against which the command is to run
- ▶ **-username** specifies the user ID with the administrative privileges required to execute the command
- ▶ **-password** specifies the password for the user ID specified in **-username**
- ▶ **-help** displays the usage requirements and a list of parameters for the command

6.4.3 Entering commands

In this section, we show you how to enter commands on the various operating systems.

Windows operating systems

Commands in Windows operating systems have an extension of .bat. It is not necessary to use the extension. Commands are not case sensitive, but parameters and names are case sensitive.

To use a command:

1. Open a command-prompt window.
2. Change to the directory where the command is, for example:
`C:\Program Files\IBM\WebSphere\AppServer\profiles\profile_name`
3. Enter the command, for example:
`serverStatus.bat -all -username <username> -password <password>`

Note: When running command-line tools on Microsoft Windows Vista or later Microsoft operating systems, on Windows Vista, Windows Server 2008, and Windows 7 operating systems, you can install WebSphere Application Server as either Administrator or non-administrator. When it is installed as Administrator, certain operations (such as those involving Windows Services) require Administrator privileges.

To ensure that WebSphere Application Server command-line tools have sufficient privileges, run them as Administrator. When you run these command-line tools from a command prompt, run them from a command prompt window that is launched by performing the following actions:

1. Right-click a command prompt shortcut.
2. Click **Run As Administrator**.
3. When you open the command-prompt window as Administrator, an operating-system dialog appears that asks you if you want to continue. Click **Continue** to proceed.

If you are using a Windows Server Core installation of Windows Server 2008, any WebSphere Application Server commands that require a graphical interface are not supported because a Windows Server Core system does not have a graphical user interface.

Therefore, commands, such as `pmt.bat` or `ifgui.bat`, are not supported on that type of Windows Server 2008 installation.

UNIX operating systems

Commands in UNIX operating systems have an extension of `.sh` and are case sensitive.

To use a command for UNIX operating systems:

1. Open a command prompt or terminal window.
2. Change to the directory where the command is, for example, for root users, the directory is:

- AIX: `/usr/IBM/WebSphere/AppServer/profiles/profile_name/bin`
- HP, Linux, or Solaris: `/opt/IBM/WebSphere/AppServer/profiles/profile_name/bin`

For non-root users, the directory is:

`user_home/IBM/WebSphere/AppServer/profiles/bin`

3. Enter the command, for example:

```
serverStatus.sh -all -username <username> -password <password>
```

IBM i operating systems

For an IBM i operating system:

1. From the IBM i command line, start a Qshell session by issuing the `STRQSH CL` command.
2. Change to the directory where the command is, for example:

```
/QIBM/ProdData/WebSphere/AppServer/V8/ND/profiles/profile_name/bin
```

3. Enter the command, for example:

```
serverStatus.sh -all -username <username> -password <password>
```

z/OS operating systems

You can manage application servers on a z/OS system from a UNIX System Services environment:

1. Enter `uss` (to switch to the UNIX System Services environment).
2. Change to the directory where the command is. On z/OS, this directory is always `app_server_root/profiles/default`, because only the profile name “default” is used in WebSphere Application Server for z/OS.

3. Enter the command, for example:

```
serverStatus.sh -all -username <username> -password <password>
```

Example 6-5 through Example 6-8 on page 230 show examples of using commands for an IBM WebSphere Application Server V8.5 environment on an AIX operation system.

Example 6-5 Backup a application server profile AppSrv_85_01 (works only after you stop the server)

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/manageprofiles.sh  
-backupProfile -profileName AppSrv_85_01 -backupFile  
/opt/IBM/WebSphere/AppSrv_85_01_backup.zip
```

Example 6-6 Enabling SDK V1.7 64 bit to all profiles of an environment

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/managesdk.sh  
-enableProfileAll -sdkname 1.7_64 -enableServers  
CWSDK1017I: Profile dmgr_85_01 now enabled to use SDK 1.7_64.
```

CWSDK1024I: The node default SDK setting for federated profile AppSrv_85_01 has been saved in the master configuration repository.
CWSDK1025I: A synchronization operation is required before configuration changes to federated profile AppSrv_85_01 can be used.
CWSDK1017I: Profile AppSrv_85_01 now enabled to use SDK 1.7_64.
CWSDK1001I: Successfully performed the requested managesdk task

Example 6-7 Synchronizing a node using the deployment manager host and SOAP port parameters

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/syncNode.sh saw211-sys1
8884
ADMU0116I: Tool information is being logged in file
           /opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/logs/syncNode.log
ADMU0128I: Starting tool with the AppSrv_85_01 profile
ADMU0401I: Begin syncNode operation for node saw211-sys1Node01 with Deployment
           Manager saw211-sys1: 8884
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0402I: The configuration for node saw211-sys1Node01 has been synchronized
           with Deployment Manager saw211-sys1: 8884
```

Example 6-8 Backup the entire configuration of a node

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/backupConfig.sh
/opt/IBM/WebSphere/backup_config.zip
ADMU0116I: Tool information is being logged in file

/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/logs/backupConfig.log
ADMU0128I: Starting tool with the AppSrv_85_01 profile
ADMU5001I: Backing up config directory
           /opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/config to file
           /opt/IBM/WebSphere/backup_config.zip
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: AppSrv_85_01
ADMU0506I: Server name: ODR_85_1
ADMU0506I: Server name: AppSrv_85_02
ADMU2010I: Stopping all server processes for node saw211-sys1Node01
ADMU0512I: Server AppSrv_85_01 cannot be reached. It appears to be stopped.
ADMU0512I: Server ODR_85_1 cannot be reached. It appears to be stopped.
ADMU0512I: Server AppSrv_85_02 cannot be reached. It appears to be stopped.
ADMU0512I: Server nodeagent cannot be reached. It appears to be stopped.
.....
ADMU5002I: 2,014 files successfully backed up
```



Part 2

Administration and configuration techniques



Administration of WebSphere processes

In this chapter, we provide information about basic administration tasks. The focus of this chapter is on managing WebSphere processes including the deployment manager, nodes and node agents, application servers, and application server clusters.

We cover the following topics in this chapter:

- ▶ Working with deployment manager
- ▶ Working with the administrative agent
- ▶ Working with the job manager
- ▶ Working with application servers
- ▶ Working with nodes in a Network Deployment environment
- ▶ Working with clusters
- ▶ Working with virtual hosts
- ▶ Managing applications
- ▶ Enabling process restart on failure

7.1 Working with deployment manager

In this section, we provide information about how to manage the deployment manager and introduce you to the configuration settings associated with it.

7.1.1 Deployment manager configuration settings

A deployment manager is built by creating a deployment manager profile. After it is built, there is usually not much that you need to do regarding the configuration of the deployment manager. However, there are settings that you can modify from the administration tools:

- ▶ Configuration
- ▶ Runtime

To view the deployment manager from the administrative console, click **System administration** → **Deployment manager**. You have two pages available, the Runtime tab and the Configuration tab, as shown in Figure 7-1.

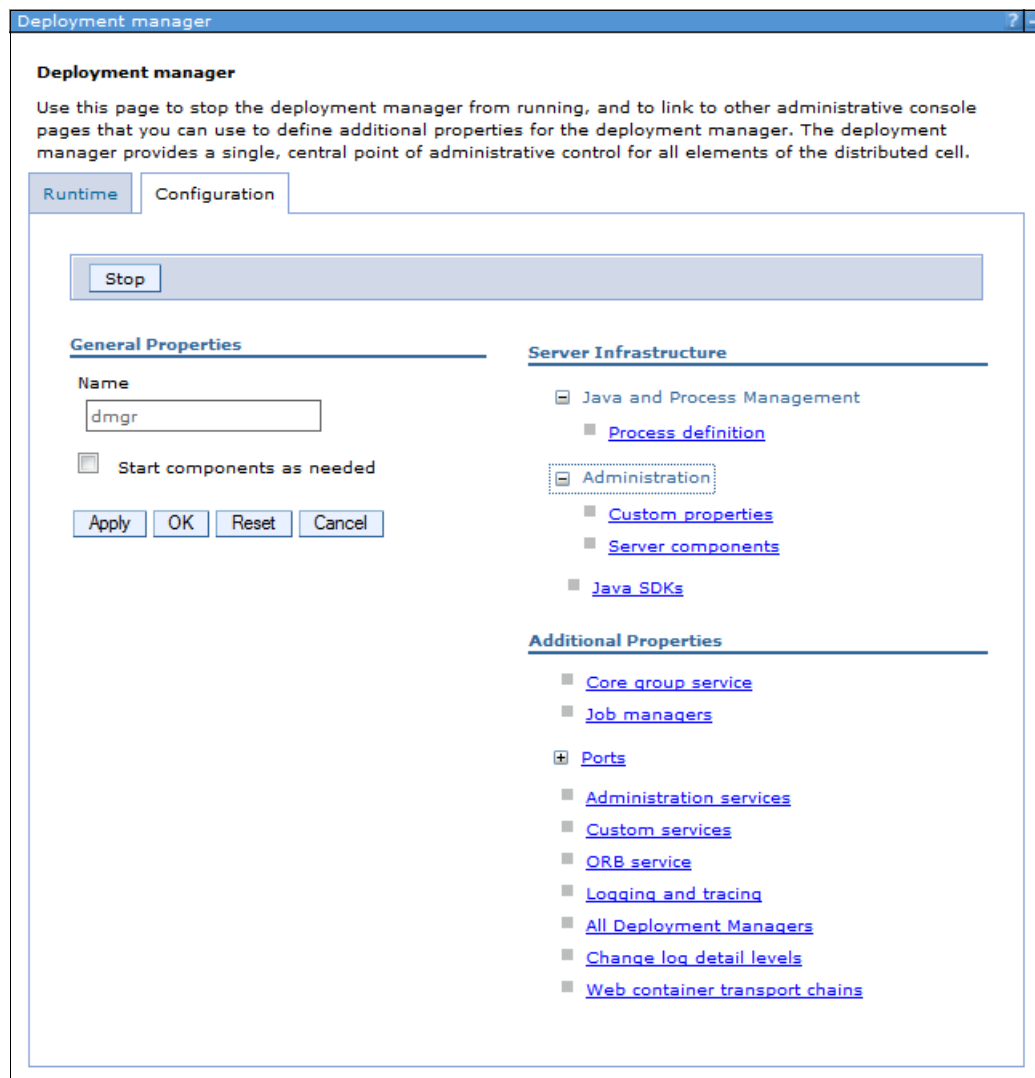


Figure 7-1 Deployment manager configuration tab

Deployment manager Configuration tab

The four options on this window to note are:

- ▶ Java SDKs

This option lets you view and chose the required version of SDKs for your deployment manager process. WebSphere Application Server V8.5 lets you select between Java SDK versions 1.6 and 1.7, both of them available for 32 and 64-bit systems. To be able to select between Java SDK versions 1.6 and 1.7, you must install both versions first. The default version that is selected when installing is Java SDK 1.6. If both versions are installed on your system, you can choose one of them and set it as default by clicking the **Make Default** button. Save your changes and restart the deployment manager process to use the newly selected JAVA SDK version.

- ▶ Job managers

Click the **Job managers** link to work with job managers. You can view the job managers that this deployment manager is registered to, and you can register or unregister the deployment manager with a job manager.

- ▶ Ports

Select the **Ports** link to view and manage the ports used by the deployment manager. This option is useful for finding the SOAP connector port required when federating a custom profile.

- ▶ All Deployment Managers

Select the **All Deployment Mangers** link and, if you have a configured high-availability (HA) deployment manager environment, you can view and manage the configured deployment managers in your environment

Deployment manager Runtime tab

The administrative console contains a Runtime tab for the deployment manager. To view the Runtime tab, click **System administration** → **Deployment manager**, and click the **Runtime** tab at the top of the page. Figure 7-2 on page 236 shows the Runtime tab.

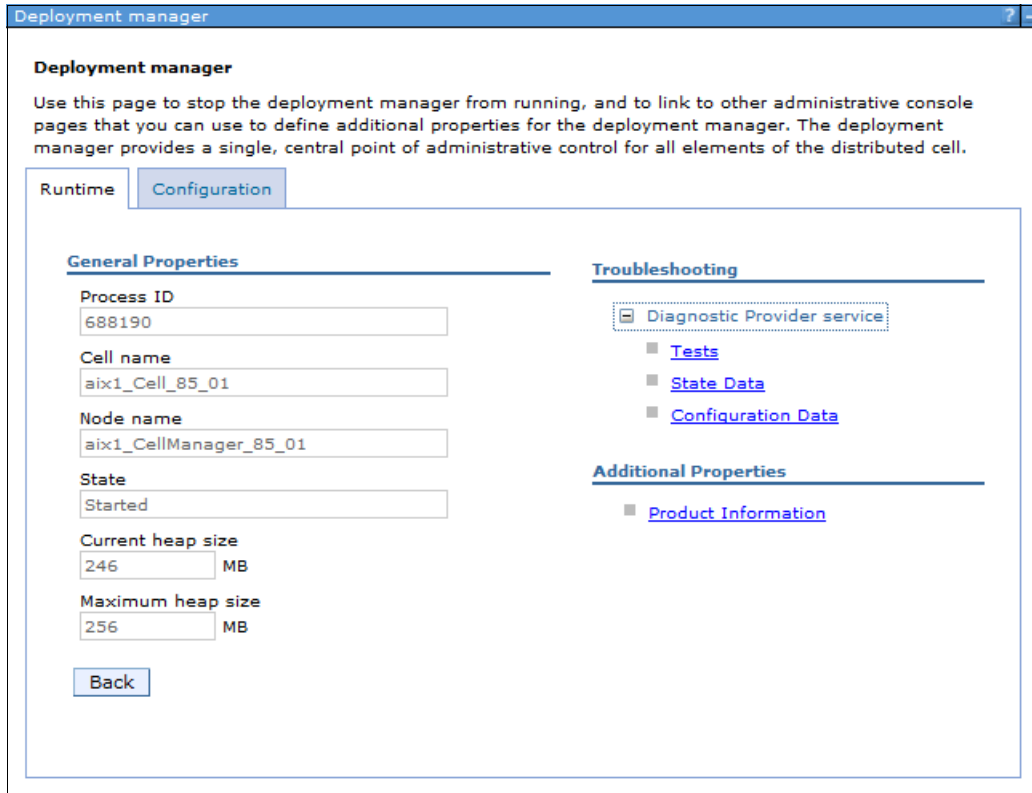


Figure 7-2 Deployment manager Runtime tab

The state is *Started*; otherwise, you cannot access the administrative console. The items on this window panel to note, are:

- ▶ Diagnostic Provider service

This option allows you to query components for current configuration data, state data, and to run a self-diagnostic test routine. Most of the time, you use these options at the request of IBM Support.

- ▶ Product information

Selecting this link opens a new page that provides information about the level of code running on the deployment manager system. It also provides links for more detailed information, including the installation history for the product and maintenance.

The product information is stored as XML files in the *install_root/properties/version/* folder and can be viewed with the administrative console, as shown in Figure 7-3 on page 237.

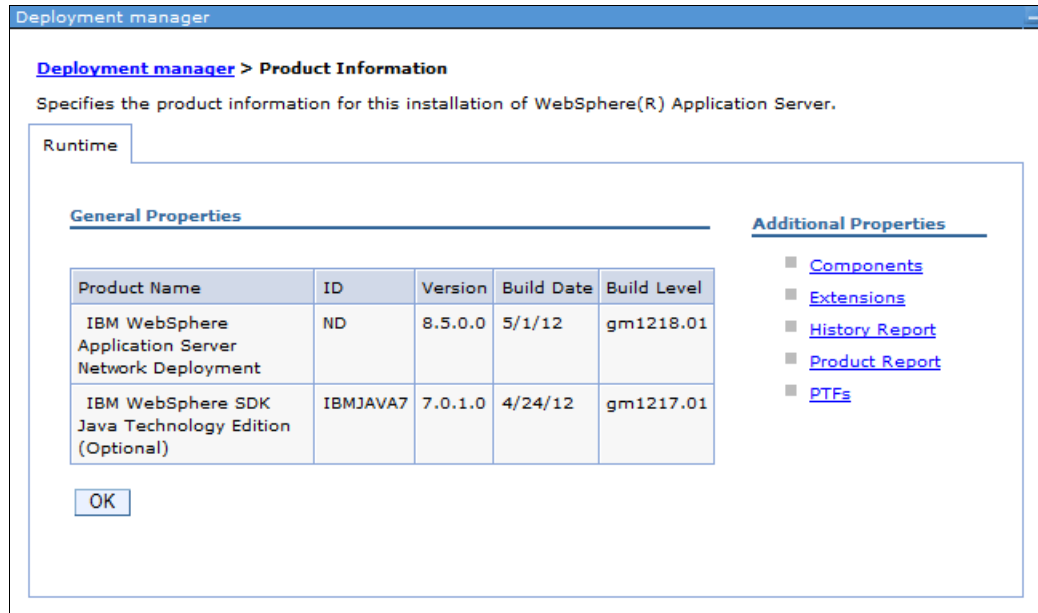


Figure 7-3 Product information

Other common deployment manager configurations

In busy environments, you might consider adjusting the maximum file size and the maximum number of the historical log files for the JVM logs of the deployment manager process. To do this, change the file size values of the System.out and System.err and the maximum number of historical log files to be saved. Consider the requirements of your environment to adjust these values.

Consider having all of your server logs in a file system outside of the default location (which is in the logs folder of the user install root of the profile). To do this you must change:

- ▶ The values of the System.out and System.err file names paths in the **Deployment manager** → **<deployment manager name>** → **JVM Logs (Configuration tab)**
- ▶ The value of the Trace Output file name path in the **Deployment manager** → **<deployment manager name>** → **Diagnostic trace service window (Configuration tab)**
- ▶ The values of the Stdout and Stderr file names paths in the **Deployment manager** → **<deployment manager name>** → **Process Logs window (Configuration tab)**
- ▶ The value of the IBM Service Logs file name path in the **Deployment manager** → **<deployment manager name>** → **IBM Service Logs window (Configuration tab)**
- ▶ The values of the NCSA Access logging and Error logging file names paths in the **Deployment manager** → **<deployment manager name>** → **NCSA Access and HTTP error logging window (Configuration tab)**

Sometimes, for example, when you have a large cell with many managed nodes and applications, you need more memory for the deployment manager process to run in the java run time. In this case, you must adjust the values of the initial heap size and maximum heap size properties in the Configuration tab of the **Deployment manager** → **Process definition** → **Java Virtual Machine** window. The default value for the maximum heap size is 256 MB.

7.1.2 Starting and stopping the deployment manager

The deployment manager can be started and stopped using command line utilities (commands). The administrative console is not available unless the deployment manager is running.

Starting the deployment manager with startManager

The **startManager** command is used to start the deployment manager on distributed systems, as shown in Example 7-1.

Example 7-1 startManager command

```
/opt/IBM/WebSphere/AppServer85/profiles/dmgr_85_01/bin/startManager.sh
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer85/profiles/dmgr_85_01/logs/dmgr/startServer.log
ADMU0128I: Starting tool with the dmgr_85_01 profile
ADMU3100I: Reading configuration for server: dmgr
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server dmgr open for e-business; process id is 704652
```

Run this command from the deployment manager *profile_root/bin* directory. If you run it from the *install_root/bin* directory, use the **-profileName** parameter to ensure that the command is run against the deployment manager profile.

Syntax of startManager

The syntax of the startManager command is:

```
startManager.bat(sh) [options]
```

The options are shown in Example 7-2.

Example 7-2 startManager options

```
Usage: startManager [options]
      options: -nowait
              -quiet
              -logfile <filename>
              -replacelog
              -trace
              -script [<script filename>] [-background]
              -timeout <seconds>
              -statusport <portnumber>
              -profileName <profile>
              -recovery
              -help
```

All arguments are optional. For more information about the **startManager** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_startmanager.html

Starting the deployment manager on z/OS (START command)

On z/OS, the deployment manager can be started using a JCL start procedure. The exact command can be found in the BBOCCINS instruction member of the JCL generated to create the profile. For example:

```
START WPCR, JOBNAME=WPMGR, ENV=WPCELL.WPDMNODE.WPMGR
```

The meaning of the command syntax is:

- ▶ WPCR is the JCL start procedure.
- ▶ WPMGR is the Job name.
- ▶ ENV is the concatenation of the cell short name, node short name, and server short name.

Starting the deployment manager also starts the following components:

- ▶ A daemon. In our example, it is named WPDEM. There is one daemon per cell, per MVS image. One of the functions of the daemon server is to provide the location name service for the cell. All daemons in the cell are fully aware of all of the objects in the cell and use the same port values.
- ▶ A controller region. In our example, it is named WPMGR. The controller region serves many functions, including acting as the endpoint for communications.
- ▶ A servant region. In our example, it is named WPMGRS. The servant region contains the JVM where the applications are run.
- ▶ If you are using messaging, you can also see a control region adjunct the server start.

Stopping the deployment manager

The deployment manager is stopped with the **stopManager** command, as shown in Example 7-3. If administrative security is enabled, you must provide an administrative user name and password for the command. You can provide this information using the **-username** and **-password** options on the command line. If you do not provide the options on the command line, you are prompted to provide credentials.

Example 7-3 stopManager command

```
/opt/IBM/WebSphere/AppServer85/profiles/dmgr_85_01/bin/stopManager.sh
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer85/profiles/dmgr_85_01/logs/dmgr/stopServer.log
ADMU0128I: Starting tool with the dmgr_85_01 profile
ADMU3100I: Reading configuration for server: dmgr
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server dmgr stop completed.
```

Syntax of stopManager

The syntax of the **stopManager** command is:

```
stopManager.bat(sh) [options]
```

The options are shown in Example 7-4.

Example 7-4 startManager options

```
Usage: stopManager [options]
      options: -nowait
              -quiet
              -logfile <filename>
              -replacelog
              -trace
```

```
-timeout <seconds>
-statusport <portnumber>
-conntype <connector type>
-port <portnumber>
-username <username>
-password <password>
-profileName <profile>
-help
```

All arguments are optional. For more information about the **stopManager** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rxml_stopmanager.html

Stopping the deployment manager on z/OS (STOP command)

To stop the deployment manager with a **STOP** command, use the following format:

```
STOP dmgr_job
```

For example:

```
STOP WPDPMGR
```

Stopping the daemon server also stops all servers for that cell, and all the servers on that daemon instance's MVS image are stopped in an orderly fashion, one by one. For example:

```
STOP WPDEMNN
```

Windows start menu and services

On a Windows system, you have the option of starting and stopping the deployment manager using the Start menu, for example, click **Start** → **All Programs** → **IBM WebSphere** → **IBM WebSphere Application Server Network Deployment V8.5** → **Profiles** → *profile_name* → **Start the deployment manager**.

Also, on a Windows system, you have the option of registering the deployment manager as a Windows service. To have it registered, you must select this option when you create the deployment manager profile or register it later using the **WASService** command.

For more information about the **WASService** command, which is also available for Linux environments, see the **WASService** command topic at the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rins_wasservice.html

If the deployment manager is registered as a Windows service, all other options for starting the dmgr process are unchanged from the administrator's point of view. However, the command used starts or stops the process through the Windows service. In addition, you have the option to allow the service to be started automatically when the operating system starts.

7.1.3 The high-availability deployment manager function

The high availability (HA) deployment manager function is configured using a shared-file system. When this configuration option is chosen, multiple deployment managers are configured. The benefit of the HA deployment manager function is that the deployment manager is no longer the single point of failure for cell administration. This is important in

environments relying on automated operations, including application deployment and server monitoring.

The deployment managers exist as peers. One is considered active, also known as primary, and hosts the administrative function of the cell. The others are backups in standby mode. If the active manager fails, a standby takes over and is designated the new active deployment manager. A command line utility is provided to clone the original cell deployment manager into additional deployment managers. Each deployment manager is installed and configured to run on a different physical or logical computer. The deployment managers need not be hosted on homogenous operating platforms, although like platforms are recommended. Each deployment manager shares the same instance of the master configuration repository and workspace area. These must be located on a shared file system.

The file system must support fast lock recovery. The IBM General Parallel File System (GPFS™) is recommended, and the Network File System Version 4 (NFS) is also an option. If you use the high-availability deployment manager on AIX Version 5.3 and are using NFS Version 4, you must have `bos.net.nfs.client` version 5.3.0.60 or later.

Normal operation includes starting at least two deployment managers. A new highly-available deployment manager component runs in each deployment manager to control which deployment manager is elected as the active one. Any other deployment manager in the configuration is in standby mode. The on demand router (ODR) is configured with the communication endpoints for the administrative console, the `wsadmin` tool, and scripting. The ODR recognizes which deployment manager instance is active and routes all administrative communication to that instance. The HA deployment manager function supports only use of the JMX SOAP connector. The JMX RMI connector is not supported in this configuration.

More information about configuring a high-availability deployment manager is at the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.wve.doc/ae/twve_xdsconfig.html

7.2 Working with the administrative agent

An administrative agent process is managed in the same manner as an application server process. The process name is `adminagent`.

7.2.1 Starting and stopping the administrative agent

To view the status of the administrative agent process:

```
<profile_root_path>/bin/serverStatus.sh(bat) -all
```

To start an administrative agent, run the following command:

```
<profile_root_path>/bin/startServer.sh(bat) adminagent
```

To stop an administrative agent, run the following command:

```
<profile_root_path>/bin/stopServer.sh(bat) adminagent
```

7.2.2 Administrative agent configuration settings

To view the administrative manager from the administrative console, click **System administration** → **Administrative agent**. You have two pages available, the Runtime tab and the Configuration tab. Figure 7-4 shows the Configuration tab.

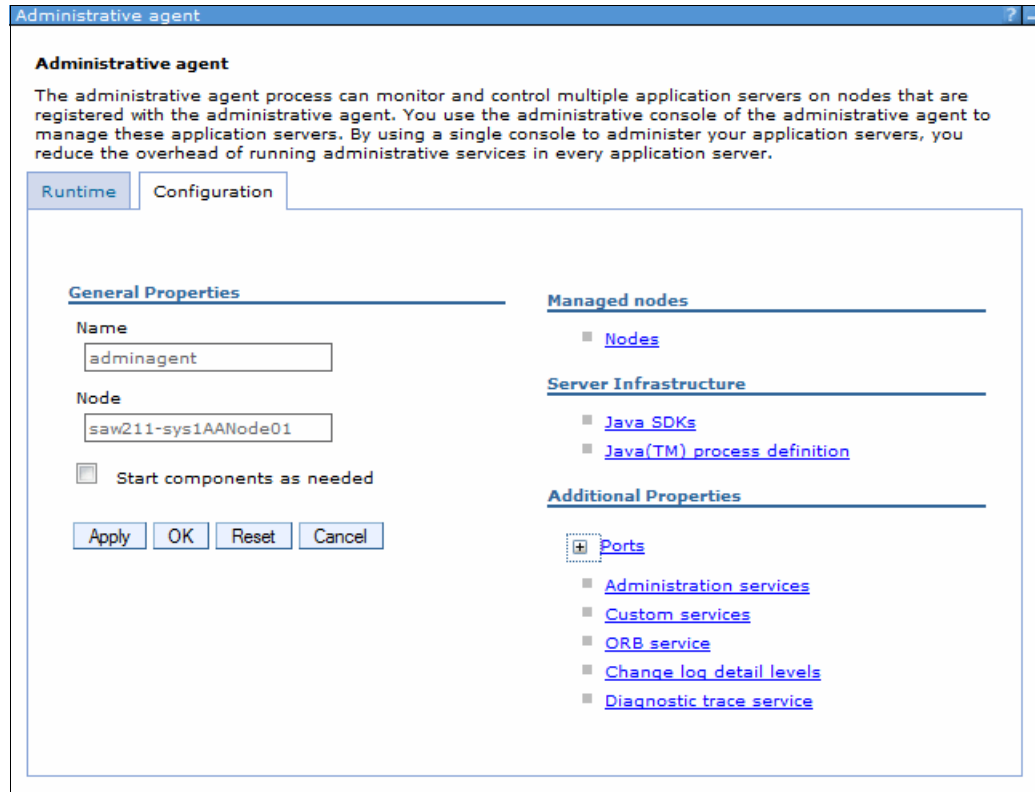


Figure 7-4 Administrative agent configuration tab

Administrative agent Configuration tab

The following list describes the two notable options in this window:

- ▶ Java SDKs

This option lets you view and choose the required version of SDKs for your deployment manager process. WebSphere Application Server V8.5 lets you select between Java SDK versions 1.6 and 1.7, both of them available for 32 and 64-bits systems. To be able to select between Java SDK versions 1.6 and 1.7, you must install both versions first. The default version that is selected when installing is Java SDK 1.6. If both versions are installed on your system, you can choose one of them and set it as default by clicking the **Make Default** button. Save your changes and restart the deployment manager process to use the newly selected JAVA SDK version.

- ▶ Ports

Select the **Ports** link to view and manage the ports that the administrative agent uses.

Administrative agent Runtime tab

In addition to the Configuration page, the administrative console contains a Runtime tab for the administrative agent. To view the Runtime tab, click **System administration** → **Administrative agent** and then click the **Runtime** tab at the top of the page. Figure 7-5 on page 243 shows the Runtime tab.

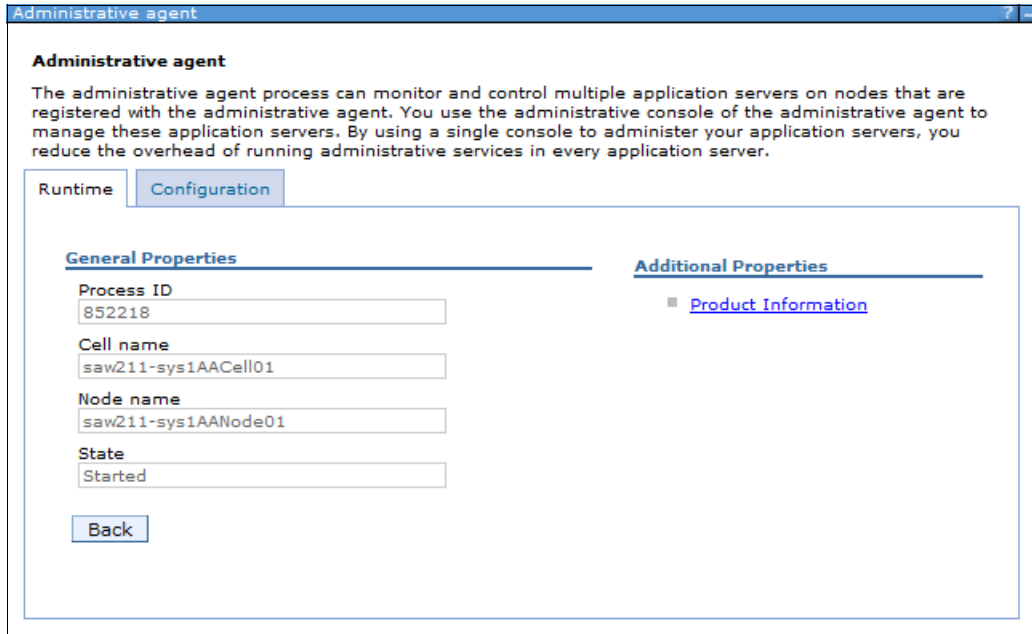


Figure 7-5 Administrative agent Runtime tab

The state is *Started* because otherwise you cannot access the administrative console.

The only link in the window panel is for Product Information. Select **Product Information** to open a new page that provides information about the level of code running on the administrative agent system. This page also provides links for more detailed information, including the installation history for the product and maintenance.

Product information is stored as XML files in the *install_root/properties/version/* folder and can be viewed with the administrative console, as shown in Figure 7-6.

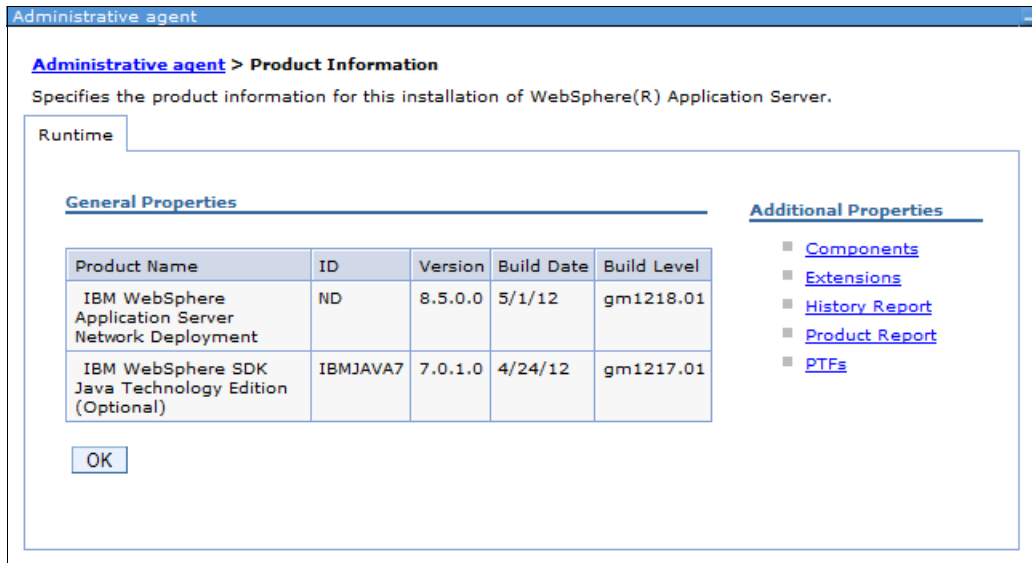


Figure 7-6 Product information

7.3 Working with the job manager

A job manager process is managed in the same manner as an application server process. The process name is **jobmgr**.

7.3.1 Starting and stopping the job manager

To view the status of the **jobmgr**, run the following command:

```
<profile_root_path>/bin/serverStatus.sh(bat) -all
```

To start a job manager, run the following command:

```
<profile_root_path>/bin/startServer.sh(bat) jobmgr
```

To stop a job manager, run the following command:

```
<profile_root_path>/bin/stopServer.sh(bat) jobmgr
```

7.3.2 Job manager configuration settings

To view the job manager from the administrative console, click **System administration** → **Job manager**. You have two pages available, the Runtime tab and the Configuration tab. Figure 7-7 shows the Configuration tab.

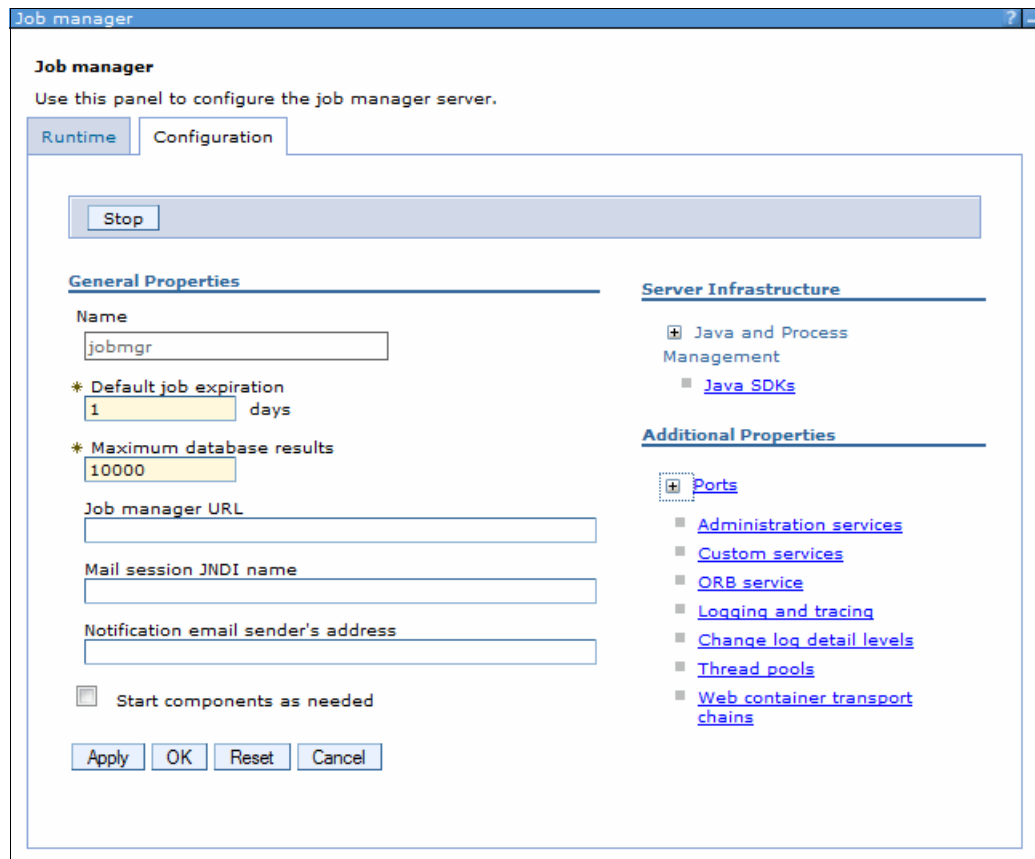


Figure 7-7 Job manager configuration tab

Job manager Configuration tab

You can stop your job manager process from this window by clicking the **Stop** button.

The two options in this window to note are:

► **Java SDKs**

This option lets you view and chose the required version of SDKs for your deployment manager process. WebSphere Application Server V8.5 lets you select between Java SDK versions 1.6 and 1.7, both of them available for 32 and 64-bits systems. To be able to select between Java SDK versions 1.6 and 1.7, you must install both versions first. The default version that is selected when installing is Java SDK 1.6. If both versions are installed on your system, you can chose one of them and set it as default by clicking the **Make Default** button. Save your changes and restart the deployment manager process to use the newly selected JAVA SDK version.

► **Ports**

Select the **Ports** link to view and manage the ports that the job manager uses.

Job manager Runtime tab

The administrative console contains a Runtime tab for the job manager. To view the Runtime tab, click **System administration** → **Job manager** and then click the **Runtime** tab at the top of the page. Figure 7-8 shows the Runtime tab.

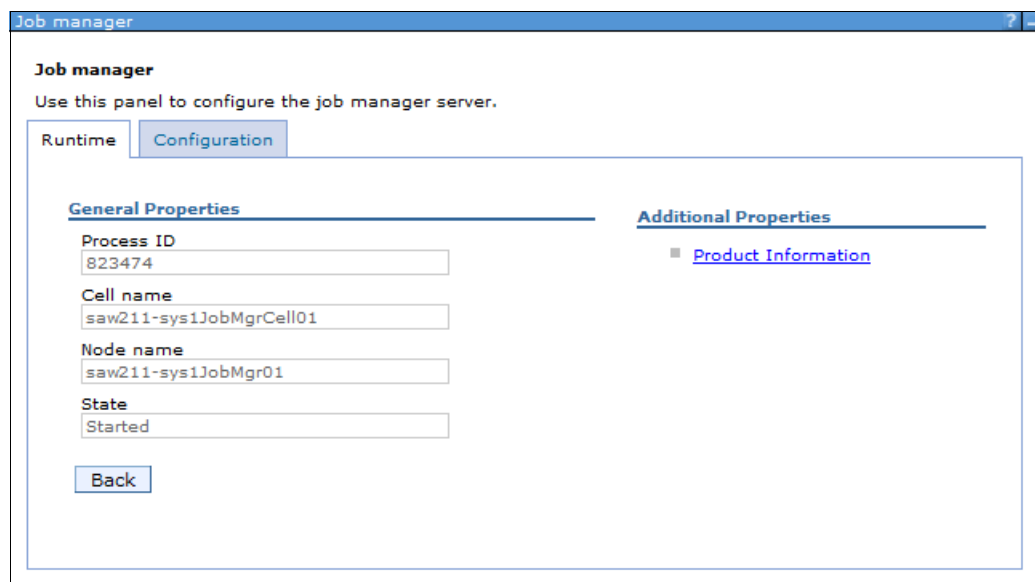


Figure 7-8 Job manager Runtime tab

The state is *Started* because otherwise you cannot access the administrative console. You can use the Process ID information to monitor the job manager process from your operating system.

The Product Information link is the only link in the window panel. Selecting **Product Information** opens a new page that provides information about the level of code running on the job manager system. This page provides links for more detailed information, including the installation history for the product and maintenance.

Product information is stored as XML files in the *install_root/properties/version/* folder and can be viewed with the administrative console, as shown in Figure 7-9 on page 246.

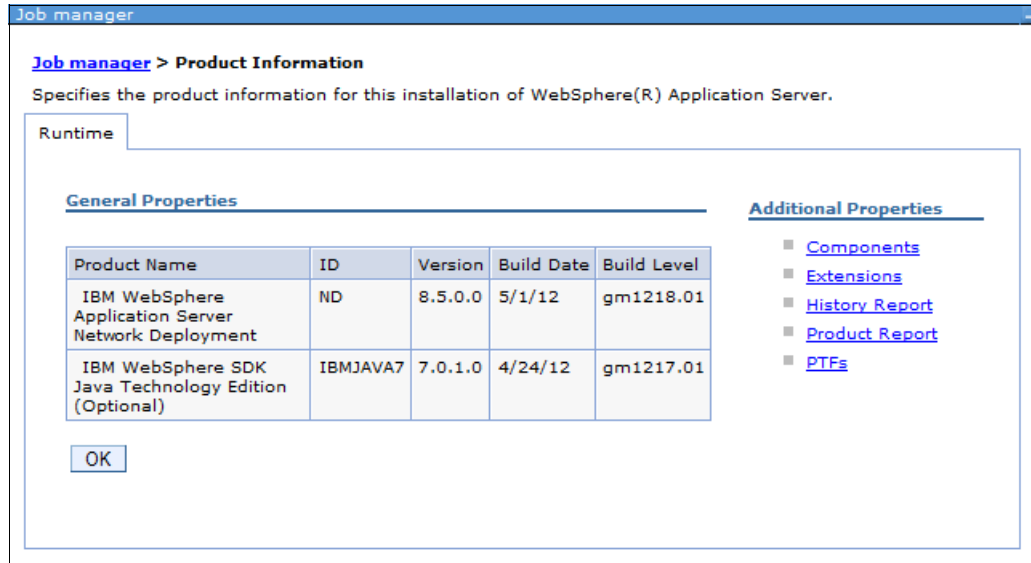


Figure 7-9 Product information

Monitoring your job manager environment

To monitor the state of your job manager environment, use the information provided by logs and traces. You can use either the basic mode for logging and tracing or the High Performance Extensible Logging (HPEL) mode. To find more information about the differences between these two modes refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/ctrb_HPELCompat.html?resultof=%2248%50%45%4c%22%20%2268%70%65%6c%22%20

Basic mode for logging and tracing

This logging and tracing mode lets you monitor your environment using the following logs:

- ▶ JVM logs (SystemOut.log and Systemerr.log): To access these logs, use the administrative console, and click **Troubleshooting** → **Logs and trace** → **jobmgr** → **JVM Logs**. Click the **Runtime** tab, and click the **View** button beside the System.out or System.err file paths. Use the Configuration tab to configure the file paths, file names, file formatting, log rotation, and size.
- ▶ Diagnostic trace service log (trace.log): To access this log, use the administrative console, and click **Troubleshooting** → **Logs and trace** → **jobmgr** → **Diagnostic trace service**. Click the **Runtime** tab, and click the **View** button beside the trace.log file path. Use the Configuration tab to configure the file paths, file names, file formatting, and size.
- ▶ Process logs (native_stdout.log and native_stderr.log): To access this log, use the administrative console, and click **Troubleshooting** → **Logs and trace** → **jobmgr** → **Process logs**. Click the **Runtime** tab, and click the **View** button beside the Stdout or Stderr file paths.
- ▶ IBM service Logs (activity.log), which has to be enabled first: To access this log, use the administrative console, and click **Troubleshooting** → **Logs and trace** → **jobmgr** → **IBM Service Logs** and then click the **Configuration** tab. You have the option to enable the correlation ID that can be used to correlate activity to a particular client request or to correlate activities on multiple application servers, if applicable.

- ▶ NCSA access and HTTP error logs (`http_access.log` and `http_error.log`): Both have to be enabled first. To enable these logs, use the administrative console, and click **Troubleshooting** → **Logs and trace** → **jobmgr** → **NCSA access and HTTP error logging** and then click the **Configuration** tab.

High Performance Extensible Logging mode for logging and tracing

It is recommended that you switch to High Performance Extensible Logging (HPEL) if you have no existing procedures that prevent you from taking advantage of it.

To enable HPEL mode:

1. Click **Troubleshooting** → **Logs and trace** → **jobmgr**.
2. Click the **Switch to HPEL Mode** button.
3. Save the changes and then restart the job manager server process.
4. Log back into the administrative console.
5. Click **Troubleshooting** → **Logs and trace** → **jobmgr**.

The window for logging and tracing the jobmgr process in HPEL mode is different from the basic mode, as shown in Figure 7-10. The HPEL mode includes two sections:

- ▶ *General Properties* that allow you to configure the HPEL logging, trace, and text log.
- ▶ *Related Items* that allow you to work with the logs.

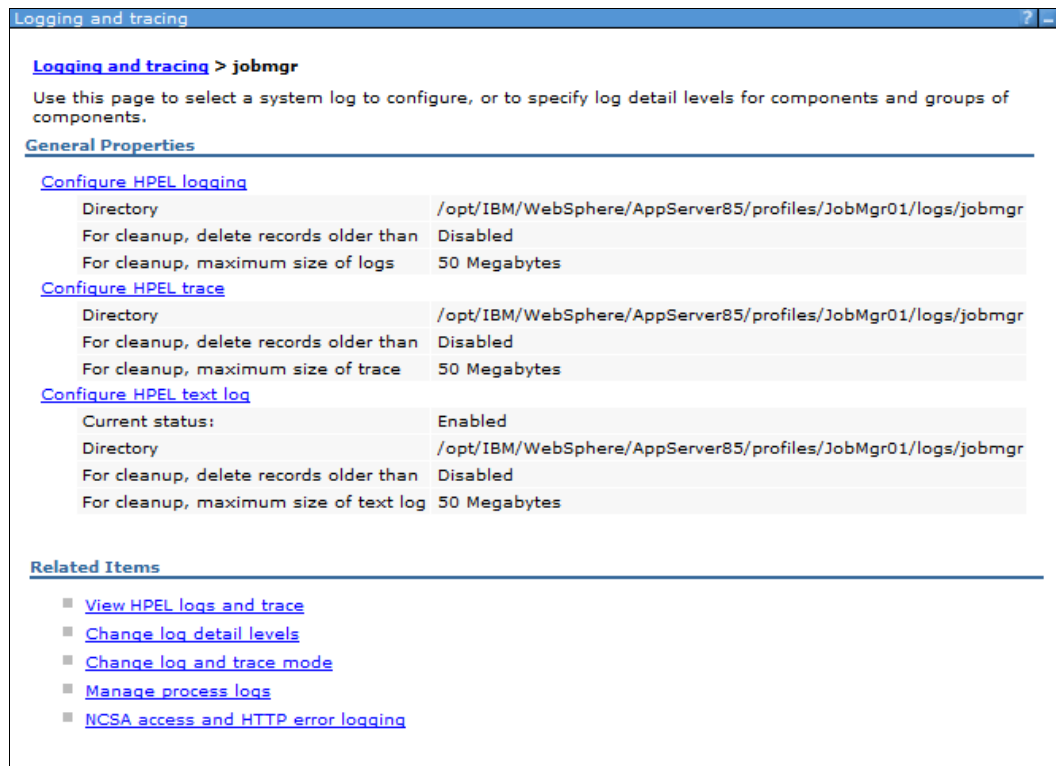


Figure 7-10 HPEL mode

To find further information about HPEL mode, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/ctrb_HPELOverview.html

7.4 Working with application servers

This section covers the following topics:

- ▶ Creating an application server
- ▶ Viewing the status of an application server
- ▶ Starting an application server
- ▶ Stopping an application server
- ▶ Viewing runtime attributes of an application server
- ▶ Customizing application servers

Terminology for application server types:

- ▶ A *stand-alone application server* is created with an application server profile and is not federated to a cell or registered with an administrative agent. A stand-alone application server hosts its own administrative services and operates independently from other WebSphere processes. It cannot participate in application server clusters. Each stand-alone application server has its own node.

This option is available on all WebSphere Application Server packages, but is the only option in the Base and Express environments.

- ▶ An *unfederated application server* is an application server that resides on a node managed from an administrative agent. Unfederated application servers have the characteristics of a stand-alone application server in that they cannot be used in a cluster. However, multiple application servers can exist on one node.

This option is available on all WebSphere Application Server packages.

- ▶ A *managed application server* is one that resides on a node that is managed from a deployment manager. A managed server can either be an application server that was created using an application server profile and subsequently federated to the cell, or it can be created directly from the deployment manager's administrative console.

Managed application servers can be clustered for high-availability and workload balancing. This action is only possible with the Network Deployment package.

7.4.1 Creating an application server

The process to create an application server depends on your WebSphere Application Server package.

Stand-alone application servers

Stand-alone application servers are created by creating an application server profile. This action results in a profile that defines one stand-alone application server. This application server hosts the sample applications and the administrative console application.

In previous versions, a stand-alone server was always named `server1`. Starting with WebSphere Application Server V7, you have the opportunity to give the server a different name during profile creation.

For information about creating an application server profile, see 3.3.3, “*Creating an application server profile*” on page 74.

Unfederated application server

An administrative agent can monitor and control multiple unfederated application servers on one or more nodes. Unfederated application servers can be created in multiple ways:

- ▶ The first server on a node to be managed by an administrative agent must be created with a stand-alone profile and then registered with the administrative agent.

The registration process disables the administrative console on the server and makes a console for the application server node available on the administrative agent process. See 3.3.10, “*Registering nodes to an administrative agent*” on page 90 for more information.

- ▶ Additional unfederated application servers on that node are created from the administrative agent. See “*Creating an application server from the administrative console*” on page 251 for more information.
- ▶ When you use the administrative agent console to register the application server node to a job manager, additional application servers can be created on the node by submitting a job from the job manager console.

What about wsadmin?

The administrative service remains active in unfederated application servers that are registered to an administrative agent. You can connect to either the application server or the administrative agent to run `wsadmin` commands, but all admin operations performed by connecting to the application server are forwarded to the agent. Connecting to the agent avoids that extra step.

Managed application servers

In a distributed server environment, you create an application server from the deployment manager administrative console. See “*Creating an application server from the administrative console*” on page 251 for more details.

If you are creating an application server with the intention of adding it to a cluster, click **Servers** → **Clusters** → **WebSphere application server cluster**. See 7.6, “*Working with clusters*” on page 292 for more details.

Application server options

You need to consider certain options as you create an application server. The method by which you select these options varies depending on how you are creating the server, but the values are the same.

Templates

An application server is created based on a template that defines the configuration settings. Four template options are provided:

- ▶ default Standard production server: You get this option if you do not specify a template for a server on a distributed system.
- ▶ default z/OS: This option is available only on z/OS platforms and is the only option until you create new templates.
- ▶ DeveloperServer: The DeveloperServer template is used when setting up a server for development use. This template configures a JVM for a quick start by disabling bytecode verification and performing JIT compilations with a lower optimization level. Do not use this

option on a production server where long run throughput is more important than early server startup.

- ▶ Custom template: You can create templates based on existing application servers (see “*Creating an application server template (optional)*” on page 258 for more details).

Ports

Each server process uses a set of ports that must be unique on the system. When you create an application server, you have the following options:

- ▶ Use the default ports: Use this selection if you will only have one application server on the system or if this is the first application server created, and port selection is not an issue.
- ▶ Have a set of ports selected that are unique to the WebSphere system installation: This selection ensures that no two WebSphere processes in the installation have the same port assigned. It does not guarantee that ports are selected that are not in use by non-WebSphere processes or by WebSphere processes installed as a separate installation.
- ▶ Specify the ports: This option is best if you have a convention for port assignment on your system that ensures unique ports are used by all processes, both WebSphere and non-WebSphere.

This option is only available when you create a new profile using the Advanced option or the **manageprofiles** command.

If you have a port conflict, you can change the ports after the application server is created.

z/OS settings

Here, we describe the various z/OS settings:

- ▶ Long name:
The long name of an application server is the name used in scripting and the administrative console. Long names can be up to 50 characters long and include mixed-case alphabetic characters, numeric characters, and the following special characters: ! ^ () _ - . { } [, and].
- ▶ Short name:
Short names are specific to the z/OS implementation of WebSphere Application Server and are the principal names by which cells, nodes, servers, and clusters are known to z/OS.
- ▶ Specific short name (z/OS):
The short name is also used as the JOBNAME for the server. If you do not specify a value for the short name field, the short name defaults to BBOSnnn, where nnn is the first free number in the cell that can be used to create a unique short name. Make sure that you set up a RACF SERVER class profile that includes this short name.
- ▶ Generic short name (z/OS):
Nodes that have not been clustered have a server generic short name, also called a *cluster transition name*. When a cluster is created from an existing application server, the server's generic short name becomes the cluster name.

No two servers on the same z/OS system can have the same server generic short name unless they are in the same cluster.

If you do not specify a value for the generic short name field, the generic short name defaults to BB0Cnnn, where nnn is the first free number in the cell that can be used to create a unique generic short name.

► Bit mode (z/OS):

The default setting is that the application server runs in 64-bit mode, but you can elect to run in 31-bit mode. Note that 31-bit mode is deprecated.

Creating an application server from the administrative console

To create an application server from the administrative console:

1. Open the deployment manager administrative console.
2. Click **Servers** → **Server Types** → **WebSphere application server**.
3. Click **New**.
4. Select the node for the new server, and enter a name for the new server (Figure 7-11).

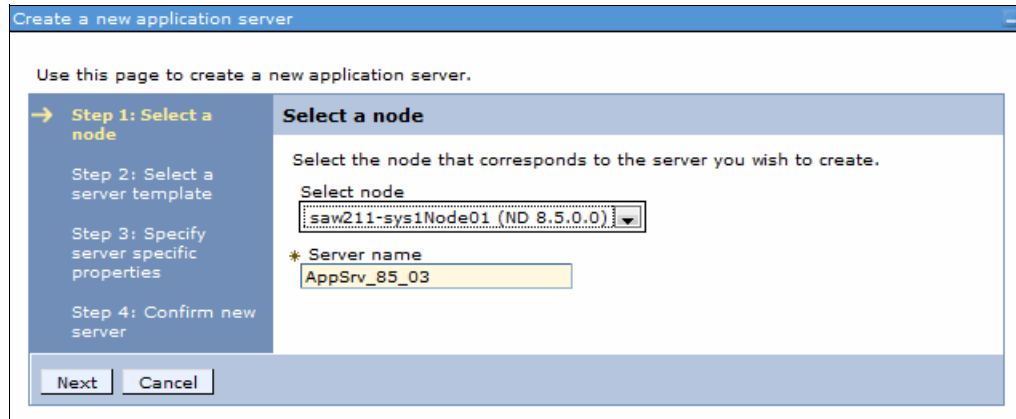


Figure 7-11 Create an application server - Select a node

Click **Next**.

5. Select a template to use by clicking the appropriate radio button (Figure 7-12).

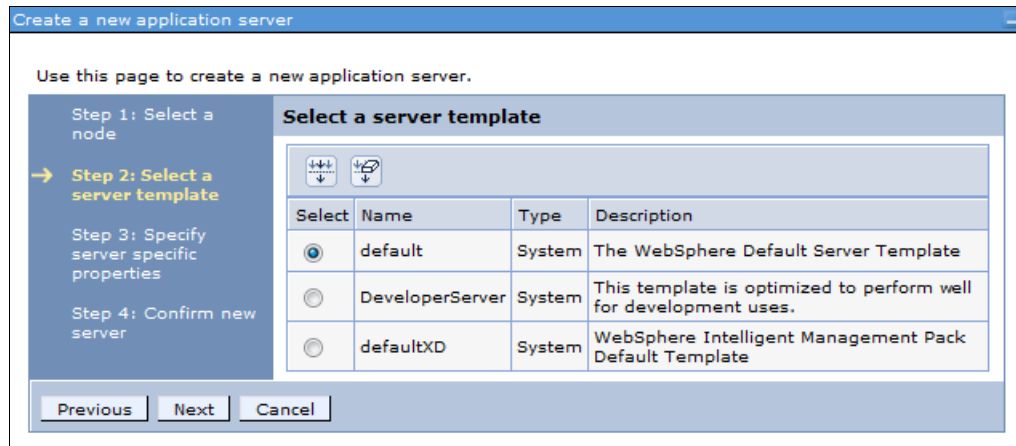


Figure 7-12 Create an application server: Select a template

On z/OS systems, there is one system-defined template called default ZOS.

Click **Next**.

- The options you see on the next window vary depending on the platform. For distributed platforms, you see the window shown in Figure 7-13.

Select the **Generate Unique Ports** box to have unique ports generated for this server. Clearing this option generates the default set of ports.

If you have multiple core groups defined, you have the option to select the core group.

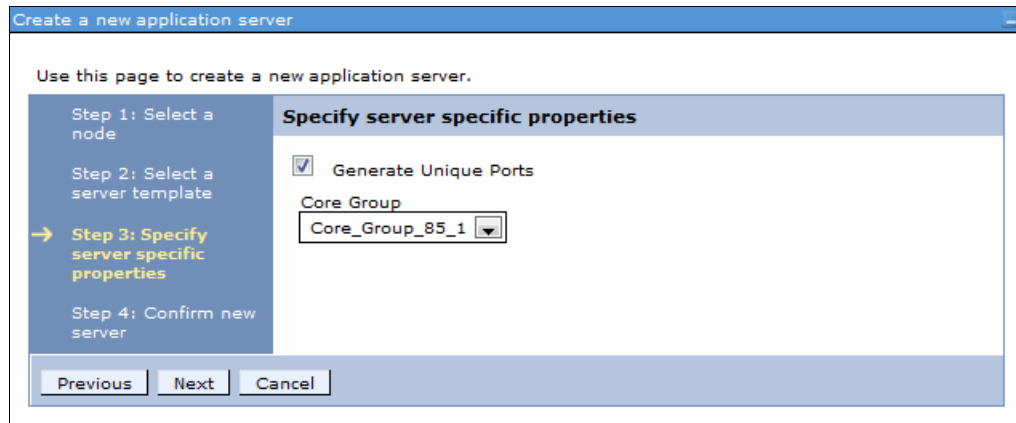


Figure 7-13 Create an application server - Generate unique ports

For z/OS systems, you see the window shown in Figure 7-14.

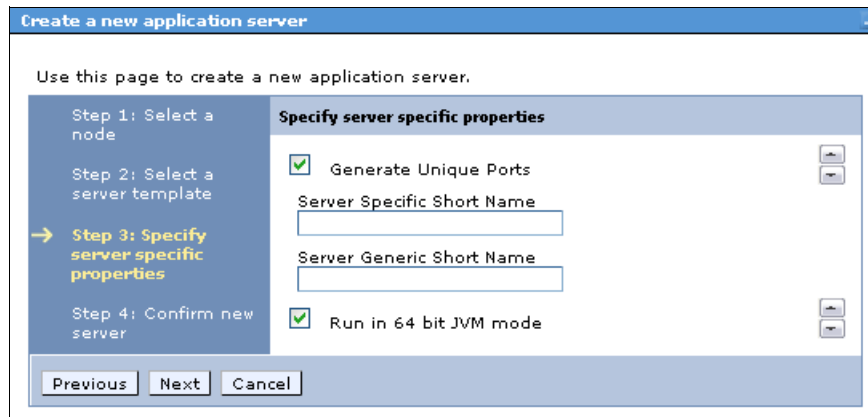


Figure 7-14 Create an application server - Generate unique ports z/OS

The server-specific short name specifies the short name for the server. This item is also used as the job name (for example, BB0S002). The generic short name is the short name that is converted to a cluster short name if the server is later used in a cluster.

Click **Next**.

- A summary window is presented with the options you chose. Click **Finish** to create the server.
- In the messages box, click **Save** to save the changes to the master repository.
- Review and update the virtual host settings (see *“Updating the virtual host settings”* on page 257 for more details).
- Regenerate the web server plug-in and propagate it to the web server (see 12.5, *“Working with the plug-in configuration file”* on page 450 for more details).

Information about managing web server plug-ins is provided in the information center. See the topic *Communicating with web server* at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/twsv_plugin.html

Note: If you are creating an application server on a Windows operating system, this process does not give you the option of registering the new server as a Windows service. You can do this task later using the **WASService** command (see 7.10, “Enabling process restart on failure” on page 313 for more details).

Creating an application server from the job manager

To create an application server from the job manager, make the following selections as you step through the process to submit the job:

1. Start the job manager and targets. Access the job manager console.
2. Click **Jobs** → **Submit**.
3. Click the **Create application server** job type. Click **Next**, as shown in Figure 7-15.

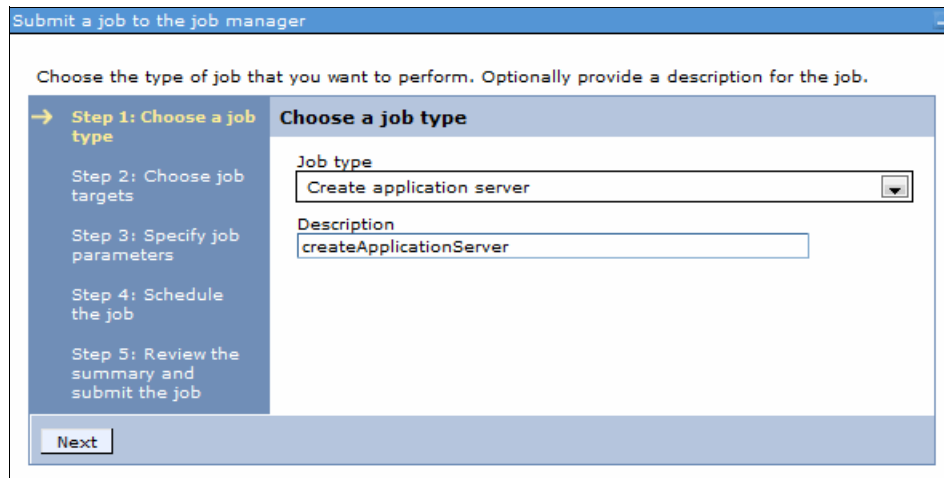


Figure 7-15 Create an application server

4. Select the job target:
 - If you are creating an application server on an unfederated node, select the **application server node**.
 - If you are creating a new managed server, select the **deployment manager node**.Add a target name by entering the name and then clicking **Add**. You can also search for a target name by clicking **Find**.

Enter the user name and password with administrative authority on the target node, as shown in Figure 7-16.

Figure 7-16 Choose a job target

Click **Next**.

5. Specify the job parameters, as shown in Figure 7-17 on page 255. At minimum:
 - Specify a unique name of the new application server. Use the **Find** button to get a list of existing server names on the target.
 - If the target is a deployment manager, enter the name of the node on which the server will be created.

Submit a job to the job manager

Enter the parameters for the job. The parameters vary based on the type of job that you previously selected.

Step 1: Choose a job type
 Step 2: Choose job targets
 → Step 3: Specify job parameters
 Step 4: Schedule the job
 Step 5: Review the summary and submit the job

Specify job parameters

Job type: Create application server

* Server name: appsrv2 Find...

Node name

Additional job parameters.

Server template

Template name

Template location

Port control

Generate unique ports

Platform specific

Specific short name

Generic short name

Bit mode: 64

Previous Next Cancel

Figure 7-17 Specify the options for the new server

You can expand the Additional job parameters section where the optional settings allow you to add platform specific settings, specify a different template, and specify the setting that determines if ports unique to the installation are generated.

- The template name field defaults to the default server template for the operating system on which the application server will run. You only need to specify this setting if you want to use a custom template or the DeveloperServer template.
- The Generate unique ports option is selected by default to generate unique ports for the installation.
- Platform-specific information is where you can provide a short name, generic name, or bit mode for creating a server on the target. If you do not provide this information, the product generates unique names and uses the default bit mode.

Click **Next**.

- Schedule the job. Take the defaults for the job schedule, as shown in Figure 7-18. The defaults execute the job once. Click **Next**.

Figure 7-18 Schedule the job

- Review the settings, and click **Finish**, as shown in Figure 7-19.

| Options | Values |
|-----------------------|-----------------------------|
| Job type | Create application server |
| Description | createApplicationServer |
| Target names | appsrv1 |
| Initial availability | Make the job available now. |
| Expiration | Use the default expiration. |
| User name | admin85 |
| Server name | appsrv2 |
| Generate unique ports | true |
| Bit mode | 64 |

Figure 7-19 Summary review

- Wait until the job status is Succeeded, as shown in Figure 7-20 on page 257.



Figure 7-20 Application server create job status

- Verify that the server was created. Click **Jobs** → **Target resources** to see the new server in the list of resources, as shown in Figure 7-10 on page 247.

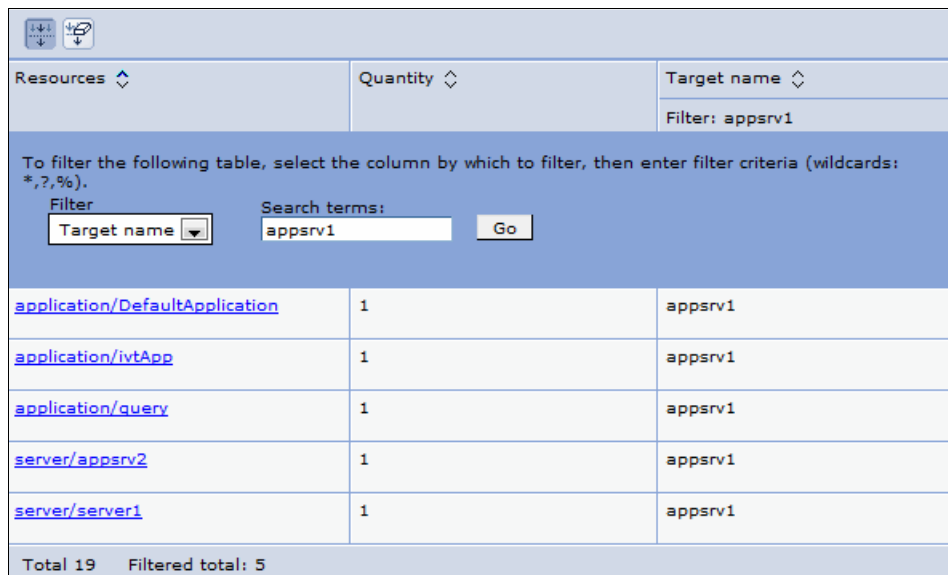


Figure 7-21 The new application server in target resources

Updating the virtual host settings

When you install applications, you associate a virtual host with each web module.

When you create a new application server, the default `t_host` virtual host is set as the default virtual host for web modules installed on the server. You can change this default in the web container settings for the application server or simply select a new virtual host when you install the applications.

If the application will only be accessed through a web server, and the virtual host that you will use is set up with the web server port in the list of host aliases, no action is necessary; however, if application clients will access the web container directly, or if you will be installing SIP applications on this server, ensure that the relevant ports generated for this application

server are added to the host alias list. See 7.7, “Working with virtual hosts” on page 303 for more information.

Creating an application server template (optional)

WebSphere Application Server provides the ability to create a customized server template that is based on an existing server configuration. Next, you can use that server template to create new servers. If you need more than one application server, for example, for a cluster, and if the characteristics of the server are different from the default server template, it is more efficient to create a custom template and use that template to create your server.

To create an application server template based on an existing server:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Click **Templates** at the top of the server list.
3. Click **New**.
4. Select a server from the list to build the template from, and click **OK**.
5. Enter a name and description for the template, and click **OK**.
6. Save your configuration.

The new template will be in the list of templates and is available to select the next time you create an application server, as shown in Figure 7-22.

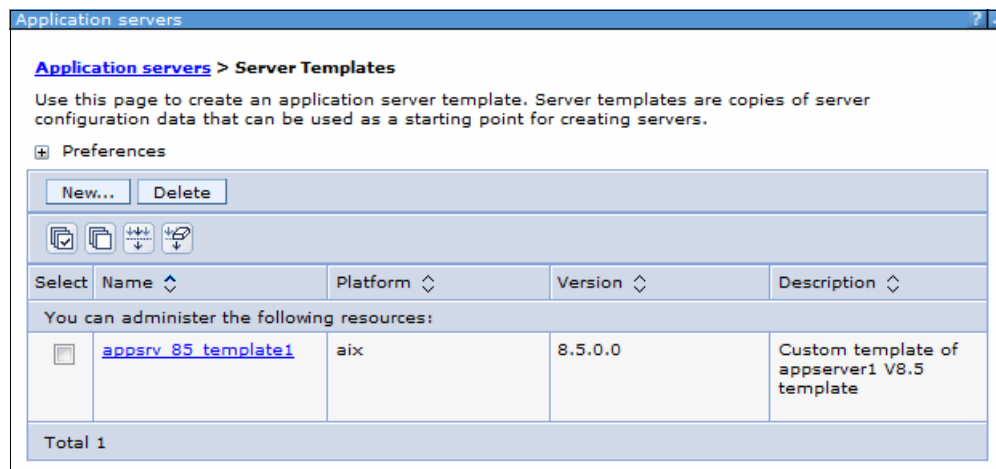


Figure 7-22 Server template listing

7.4.2 Viewing the status of an application server

There are multiple ways to check the status of an application server:

- ▶ Use the **serverStatus** command on the system where the application server is running.
- ▶ In a distributed environment, you can view the status from the administrative console. The node for the application server must be active for the deployment manager to know the status of a server on that node.
- ▶ From the job manager console.
- ▶ If the server is registered as a Windows service, you can check the status of the service.

Using the administrative console

To check the status of a managed server using the deployment manager's administrative console, the node agent must be started. To use the administrative console:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.

The servers are listed. The last column on the right side contains an icon to indicate the status of each server, as shown in Figure 7-23.

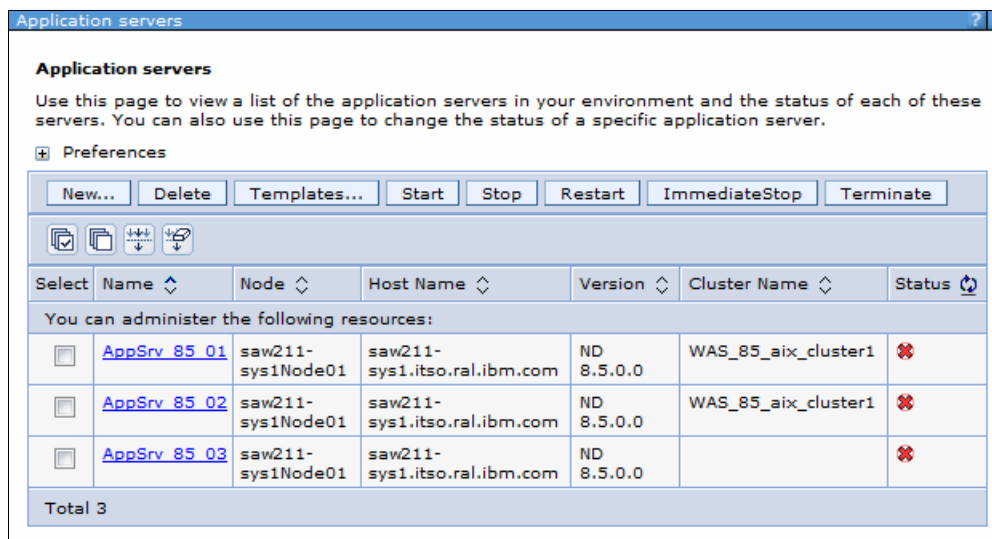


Figure 7-23 Administrative console server status

Figure 7-24 shows the icons and their corresponding status.

| Icon | Status | Description |
|------|-------------------|---|
| | Started | The server is running. |
| | Partially stopped | The server is in the process of changing from a started state to a stopped state. |
| | Stopped | The server is not running. |
| | Unavailable | The server status cannot be determined. |

Figure 7-24 Status icons

Note: If the server status is Unknown, the node agent on the node in which the application server is installed is not active. The server cannot be managed from the administrative console unless its node agent is active.

Using the serverStatus command

The syntax of the `serverStatus` command is:

```
serverStatus.bat(sh) [options]
```

The options are shown Example 7-5.

Example 7-5 serverStatus options

```
Usage: serverStatus <server name | -all>
[-logfile <filename>]
[-replaceLog]
[-trace]
[-username <username>]
```

```
[-password <password>]
[-profileName <profile>]
[-help]
```

The first argument is mandatory. The argument is either the name of the server for which status is desired or the **-all** keyword, which requests status for all servers defined on the node.

If you have administrative security enabled, you must enter the user ID and password of an administrator ID. If you do not include it in the command, you are prompted for it for example, to view the status of a server, run the following command:

```
cd profile_home/bin
serverStatus.sh server_name -username adminID -password adminpw
```

To check the status of all servers on the node, run the following command:

```
cd profile_home/bin
serverStatus.sh -all -username adminID -password adminpw
```

For more information about the **serverStatus** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_serverstatus.html

Example 7-6 shows an example of using the **serverStatus** command.

Example 7-6 serverStatus example - AIX operating system

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/serverStatus.sh -all
-username admin85 -password admin85
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/logs/serverStatus.log
ADMU0128I: Starting tool with the AppSrv_85_01 profile
ADMU0503I: Retrieving server status for all servers
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: AppSrv_85_01
ADMU0506I: Server name: ODR_85_1
ADMU0506I: Server name: AppSrv_85_02
ADMU0506I: Server name: k
ADMU0506I: Server name: AppSrv_85_03
ADMU0508I: The Node Agent "nodeagent" is STARTED
ADMU0509I: The Application Server "AppSrv_85_01" cannot be reached. It appears
to be stopped.
ADMU0509I: The Application Server "ODR_85_1" cannot be reached. It appears to
be stopped.
ADMU0509I: The Application Server "AppSrv_85_02" cannot be reached. It appears
to be stopped.
ADMU0509I: The Server "k" cannot be reached. It appears to be stopped.
ADMU0509I: The Application Server "AppSrv_85_03" cannot be reached. It appears
to be stopped.
```

From the job manager console

To display the servers and their status from the job manager console, complete the following steps:

1. Click **Jobs** → **Targets**.
2. Select the **box** to the left of the node name. In the **Display resources** drop-down menu, click **Server**, as shown in Figure 7-25.

| Select | Target name | Version |
|-------------------------------------|-----------------------------|---|
| <input checked="" type="checkbox"/> | appsrv1 | XD 8.5.0.0 WXDOP 8.5.0.0 ND 8.5.0.0 |
| <input type="checkbox"/> | saw211-sys1 | |
| Total 2 | | |

Figure 7-25 Display the servers on a node

3. This action displays the list of servers in the node. Click the name of the server, as shown in Figure 7-26.

| Resources | Quantity | Target name |
|--------------------------------|----------|-------------|
| server/appsrv2 | 1 | appsrv1 |
| server/server1 | 1 | appsrv1 |
| Total 2 | | |

Figure 7-26 Display the servers on a node

The status of the server is displayed, as shown in Figure 7-27.

| Resource ID | Target name | Status |
|--|-------------|---------|
| appsrv1/server/appsrv2 | appsrv1 | Stopped |
| Total 1 | | |

Figure 7-27 Display the servers on a node

7.4.3 Starting an application server

How you start an application server depends largely on personal preference and on whether the application server is stand-alone or managed. This section provides information about how to start individual application servers. You can also start all application servers in a cluster by starting the cluster (see 7.6.3, “Managing clusters” on page 302 for more details).

Using the administrative console to start a managed server

Tip: Before managing a server in a distributed server environment using the administrative console, the node agent for the server's node must be running. To check the status of the node, click **System administration** → **Node Agents**. The status of the node agent is in the far right column. If it is not started, you must start it manually (see 7.5.1, "Starting and stopping nodes" on page 282 for more details).

From the administrative console:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select the box to the left of each server that you want to start.
3. Click **Start**.
4. Verify the results in the Server status feedback window.

If there are any errors, check the log files for the application server process. The default location for the logs is:

- ▶ `profile_home/logs/server_name/SystemOut.log`
- ▶ `profile_home/logs/server_name/startServer.log`

On z/OS, check the output in the application server job log.

Tip: By default, all the applications on a server start when the application server starts. To prevent an application from starting, see 7.9.2, "Preventing an enterprise application from starting on a server" on page 307 for more details.

Using the startServer command

The syntax of the **startServer** command is shown in Example 7-7.

Example 7-7 startServer options

```
Usage: startServer <server> [options]
      options: -nowait
               -quiet
               -logfile <filename>
               -replacelog
               -trace
               -script [<script filename>] [-background]
               -timeout <seconds>
               -statusport <portnumber>
               -profileName <profile>
               -recovery
               -help
```

For reference, <server> is the name of the server to be started. The first argument is mandatory and case sensitive.

For more information about the **startServer** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_startserver.html

startServer example

Example 7-8 shows an example of using the **startServer** command. Note that the user ID and password are not required to start the server.

Example 7-8 startServer example

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/startServer.sh
AppSrv_85_03
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/logs/AppSrv_85_03/startServer
.log
ADMU0128I: Starting tool with the AppSrv_85_01 profile
ADMU3100I: Reading configuration for server: AppSrv_85_03
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server AppSrv_85_03 open for e-business; process id is 749660
```

Starting a server from the job manager

To start an application server from the job manager:

1. Access the job manager console.
2. Click **Jobs** → **Submit**.
3. Select the **Start server** job type. Click **Next**, as shown in Figure 7-28.

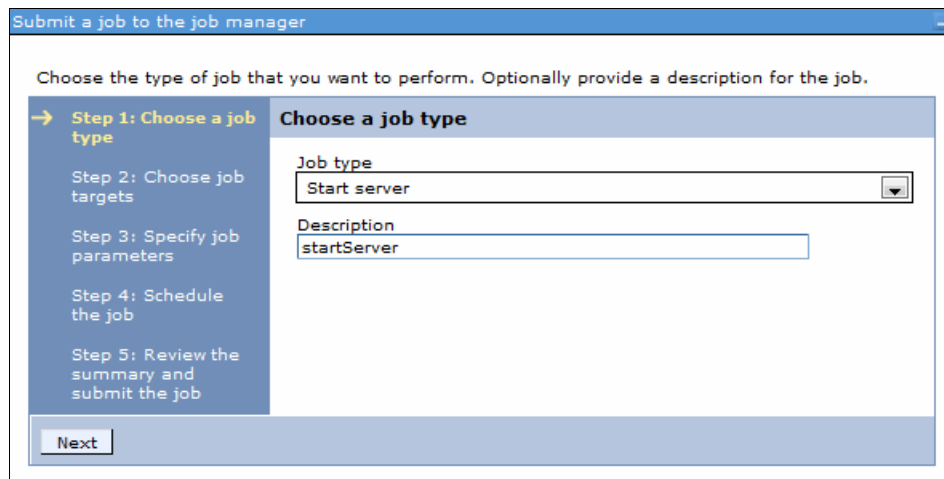


Figure 7-28 Select a job type

4. Select the job target:
 - If you are starting an application server on an unfederated node, select the **application server node**.
 - If you are starting a new managed server, select the **deployment manager node**.Add a target name by entering the name and clicking **Add**. You can also search for a target name by clicking **Find**.

Enter the user ID and password with administrative authority on the target node, as shown in Figure 7-29.

The screenshot shows a dialog box titled "Choose job targets". On the left, a vertical sidebar lists five steps: "Step 1: Choose a job type", "Step 2: Choose job targets" (highlighted with a yellow arrow), "Step 3: Specify job parameters", "Step 4: Schedule the job", and "Step 5: Review the summary and submit the job". The main area is titled "Choose job targets" and has a "Job type: Start server". There are two radio buttons: "Target groups" (unselected) and "Target names" (selected). Below "Target groups" is a dropdown menu showing "-- No groups --". Below "Target names" is a text input field, an "Add" button, and a "Find..." button. A list box below contains "appsrv1" which is highlighted in blue. A "Remove" button is at the bottom right of the list box. Below this is a section titled "Target authentication" with a horizontal line. It contains a "User name" field with "admin85". There are two radio buttons: "Password authentication" (selected) and "Target authentication" (unselected). Below are two password fields: "* Password" and "* Confirm password", both masked with dots.

Figure 7-29 Select a target

Click **Next**.

5. Specify the job parameters:

- Specify the name of the application server. Use the **Find** button to get a list of existing server names on the target.
- If the target is a deployment manager, enter the name of the node on which the server will be created.

Click **Next**, as shown in Figure 7-30.

The screenshot shows a dialog box titled "Specify job parameters". On the left, a vertical sidebar lists five steps: "Step 1: Choose a job type", "Step 2: Choose job targets", "Step 3: Specify job parameters" (highlighted with a yellow arrow), "Step 4: Schedule the job", and "Step 5: Review the summary and submit the job". The main area is titled "Specify job parameters" and has a "Job type: Start server". There are two required fields: "* Server name" and "* Node name". The "Server name" field contains "appsrv2" and has a "Find..." button to its right. The "Node name" field is empty. At the bottom of the dialog are three buttons: "Previous", "Next", and "Cancel".

Figure 7-30 Select the job parameters

6. Schedule the job. Take the defaults for the job schedule. The defaults execute the job once. Click **Next**, as shown in Figure 7-31 on page 265.

Step 1: Choose a job type

Step 2: Choose job targets

Step 3: Specify job parameters

→ Step 4: Schedule the job

Step 5: Review the summary and submit the job

Schedule the job

Job type: Start server

Notification

Email addresses

Initial Availability

Specify when this job is first available.

Make the job available now.

Schedule availability

Date (MM/dd/yyyy) / / Time (HH:mm:ss) : :

Expiration

Specify when this job is no longer available.

Use default expiration - 1 days.

Expire the job based on a date

Date (MM/dd/yyyy) / / Time (HH:mm:ss) : :

Expire the job based on a duration

Expire after

Job Availability Interval

Jobs can run repeatedly based on an interval. Specify the interval that the job is available.

Availability interval

Figure 7-31 Schedule the job

7. Review the summary, and click **Finish**, as shown in Figure 7-32.

Step 1: Choose a job type

Step 2: Choose job targets

Step 3: Specify job parameters

Step 4: Schedule the job

→ Step 5: Review the summary and submit the job

Review the summary and submit the job

Summary of actions:

| Options | Values |
|----------------------|-----------------------------|
| Job type | Start server |
| Description | startServer |
| Target names | appsrv1 |
| Initial availability | Make the job available now. |
| Expiration | Use the default expiration. |
| User name | admin85 |
| Server name | appsrv2 |

Figure 7-32 Summary review

8. Monitor the status of the job and ensure it completes successfully, as shown in Figure 7-33 on page 266.

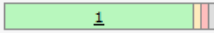
| Select | Job ID | Description | State | Activation Time | Expiration Time | Status Summary |
|--------------------------|------------------------------------|-------------|--------|------------------------|------------------------|---|
| <input type="checkbox"/> | 134015553016994487 | startServer | Active | 06/19/2012 19:25:30 | 06/20/2012 19:25:30 |  |
| Total 1 | | | | | | |

Figure 7-33 Start server job completed

7.4.4 Stopping an application server

This section shows multiple methods for stopping a server.

Using the administrative console to stop a managed server

Note: These directions assume that the node agent for the application server is running.

From the administrative console, you have the following options to stop an application server (Figure 7-34):

- ▶ The Stop button quiesces the application server and stops it. In-flight requests are allowed to complete.
- ▶ The Restart button stops and then starts the server.
- ▶ The Immediate Stop button stops the server, but bypasses the normal server quiesce process, which enables in-flight requests to complete before shutting down the entire server process. This shutdown mode is faster than the normal server stop processing, but some application clients can receive exceptions.
- ▶ The Terminate button deletes the application server process. Use this if immediate stop fails to stop the server.




| Select | Name | Node | Host Name | Version | Cluster Name | Status |
|---|------------------------------|-------------------|------------------------------|---------------|---------------------|---|
| You can administer the following resources: | | | | | | |
| <input type="checkbox"/> | AppSrv 85 01 | saw211-sys1Node01 | saw211-sys1.itso.ral.ibm.com | ND 8.5.0.0 | WAS_85_aix_cluster1 |  |
| <input type="checkbox"/> | AppSrv 85 02 | saw211-sys1Node01 | saw211-sys1.itso.ral.ibm.com | ND 8.5.0.0 | WAS_85_aix_cluster1 |  |
| <input type="checkbox"/> | AppSrv 85 03 | saw211-sys1Node01 | saw211-sys1.itso.ral.ibm.com | ND 8.5.0.0 | |  |
| Total 3 | | | | | | |

Figure 7-34 Administrative console buttons

From the administrative console, complete the following steps to stop an application server:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select the **box** to the left of each server you want to stop.
3. Click the appropriate stop option.

If there are any errors, check the log files for the application server process:

- ▶ `profile_home/logs/server_name/SystemOut.log`
- ▶ `profile_home/logs/server_name/stopServer.log`

On z/OS, check the output in the application server job log.

Restarting all servers on a node

If you want to stop and then restart all the application servers on a node, complete the following steps from the administrative console:

1. Click **System administration** → **Node agents**.
2. Select the **box** to the left of the node agent.
3. Click **Restart all Servers on Node**.

Stopping all servers in a cluster

If you want to stop all the servers in a cluster, complete the following steps from the administrative console:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Select the **box** to the left of the cluster.
3. Click **Stop** or **Immediate Stop**.

Restarting all servers in a cluster

If you want to stop and then restart all the servers in a cluster, complete the following steps from the administrative console:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Select the **box** to the left of the cluster.
3. Click **Ripplestart**.

Using the stopServer command

The syntax of the `stopServer` command is:

```
stopServer.bat(sh) [options]
```

The options are shown in Example 7-9.

Example 7-9 stopServer command

```
Usage: stopServer <server> [options]
      options: -nowait
               -quiet
               -logfile <filename>
               -replacelog
               -trace
               -timeout <seconds>
               -statusport <portnumber>
               -conntype <connector type>
               -port <portnumber>
               -username <username>
               -password <password>
               -profileName <profile>
               -help
```

For reference, `<server>` is the name of the server to be started. The first argument is mandatory and is case sensitive. If administrative security is enabled, a user name and password is also required.

For more information about the **stopServer** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rxml_stopserver.html

Example 7-10 shows an example of the **stopServer** command.

Example 7-10 stopServer command example

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/stopServer.sh
AppSrv_85_03 -username admin85 -password admin85
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/logs/AppSrv_85_03/stopServer.
log
ADMU0128I: Starting tool with the AppSrv_85_01 profile
ADMU3100I: Reading configuration for server: AppSrv_85_03
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server AppSrv_85_03 stop completed.
```

Note: If you attempt to stop a server and for some reason it does not stop, you can use the operating system commands to stop the Java process for the server.

7.4.5 Viewing runtime attributes of an application server

To view runtime attributes using the administrative console:

1. Click **Servers** → **Server Types** → **WebSphere application servers** to display the list of servers.
2. Click the **server name** to access the detail page.
3. If the server is running, four tabs are displayed: Configuration, Reports, Operation and Runtime. If the server is not running, the Runtime tab is not displayed. Click the **Runtime** tab. Figure 7-35 on page 269 shows the Runtime tab and the information that it provides.

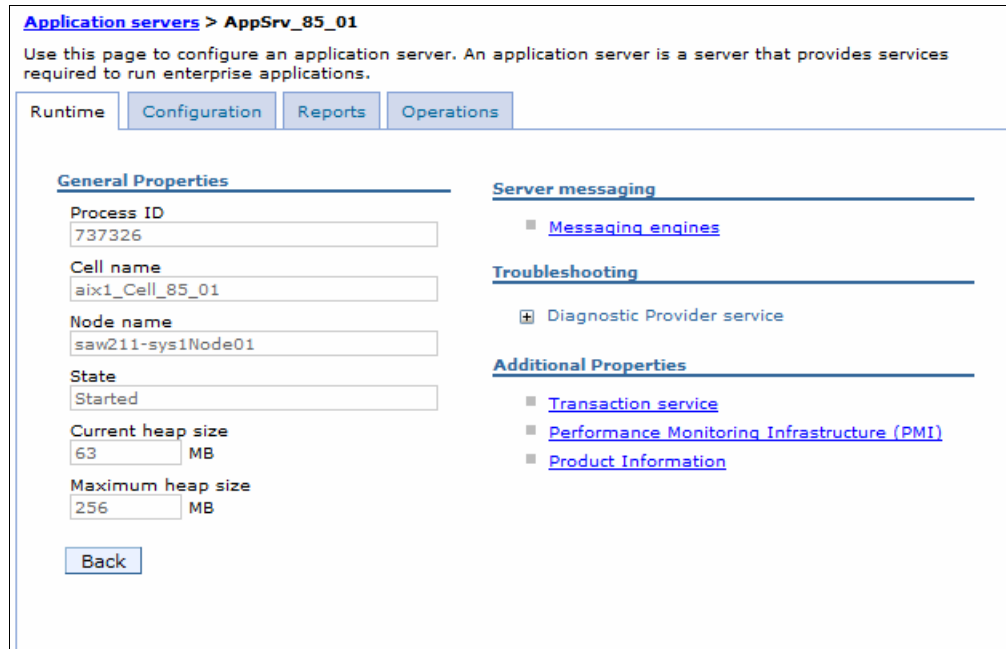


Figure 7-35 Application server Runtime tab

The Runtime tab is composed of the following items:

- General information about the server process, including the process ID, cell name, node name, state, and the current and maximum heap size.
- Access to a list of messaging engines that run on this application server. There will be one messaging engine for each bus of which the server is a member. You can start and stop the messaging engine from this window.
- Access to the Diagnostic Provider service. This allows you to query current configuration data, state, and initiate diagnostic tests.
- Access to the Transaction Service properties settings. You can change the timeout settings while the server is running but not the transaction log directory setting. Changes made in this window are active until the server is stopped.

To make these settings permanent and to configure additional Transaction Service settings, click the **Configuration** tab while you have this page open. This action takes you directly to the Transaction Service settings.

You can also view or act on transactions in the following states by clicking **Review**, located to the right of the state. This action is not normally necessary, but in an exceptional situation, it might be useful.

- Manual transactions:

These transactions await administrative completion. For each transaction, the local or global ID is displayed. You can display each transaction resource and its associated resource manager. You can choose also to commit or rollback transactions in this state.

- Retry transactions:

These are transactions with some resources being retried. For each transaction, the local or global ID is displayed, and whether the transaction is committing or rolling back. You can display each transaction resource and its associated resource

manager. You can choose also to finish, or abandon retrying, transactions in this state.

- Heuristic transactions:

These are transactions that completed heuristically. For each transaction, the local or global ID and the heuristic outcome is displayed. You can display each transaction resource and its associated resource manager. You can also choose to clear the transaction from the list.

- Imported prepared transactions:

Transactions that were imported and prepared but not yet committed. For each transaction, the local or global ID is displayed. You can display each transaction resource and its associated resource manager. You can also choose to commit or roll back transactions in this state.

- Performance Monitoring Service settings allow you to change the instrumentation levels while the server is running or make permanent changes to the configuration for that server.
- Product Information gives you access to extensive information about the product installation and Fix Pack information.

7.4.6 Customizing application servers

When you create a new application server, it inherits most of its configuration settings from the specified template server. To view or modify these settings, click **Servers** → **Server Types** → **WebSphere application servers**. A list of application servers defined in the cell appears in the workspace. Click the name of the application server to make a modification.

This section gives you a quick overview of the types of settings that you can customize. See Figure 7-36 on page 271 for a list of most of the settings (not all settings are shown due to the size of the configuration window).

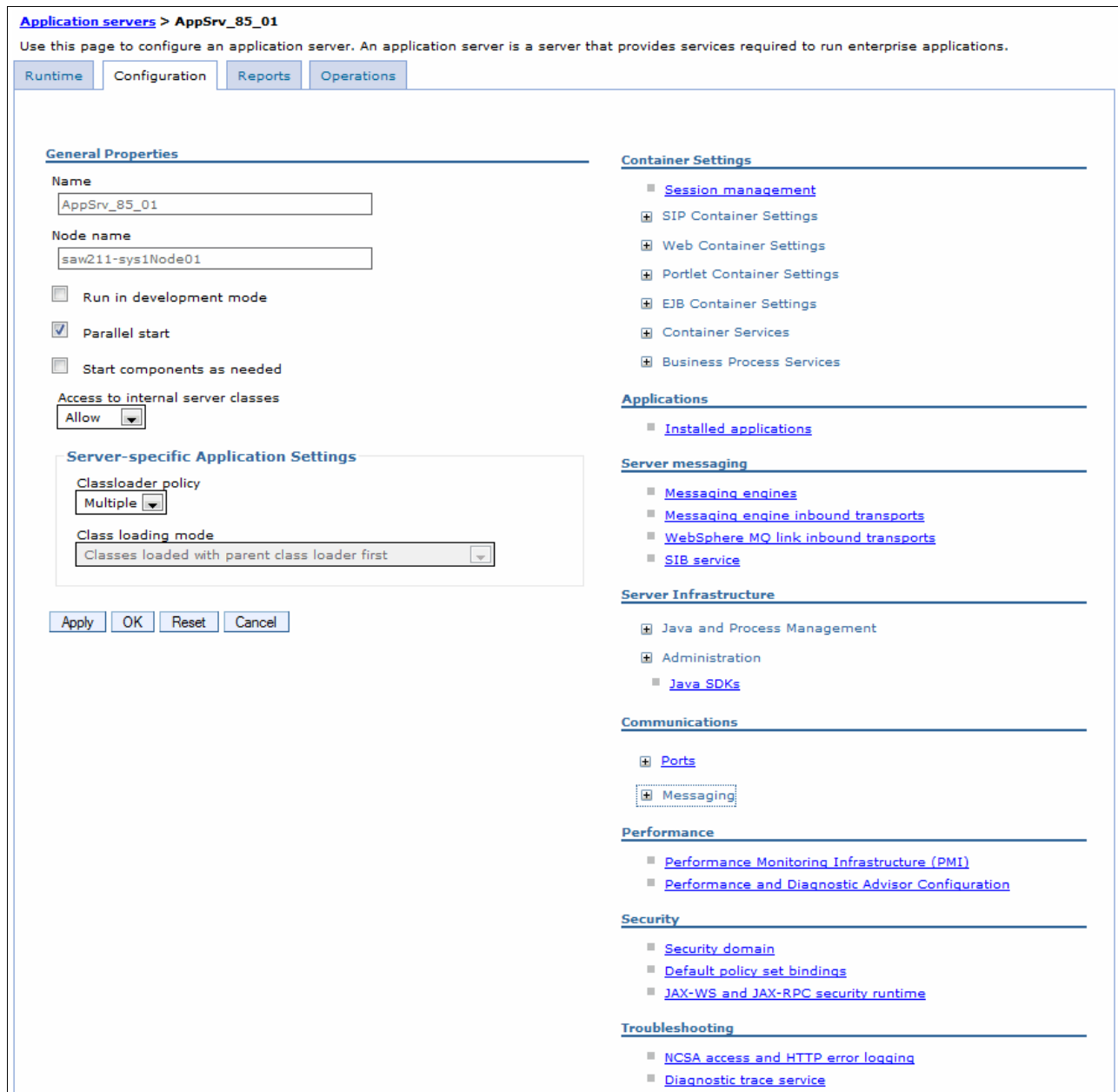


Figure 7-36 Application server configuration

General properties

The general properties consist of a few items that you can see immediately:

- ▶ Server name and node name is defined. You also have short name and unique ID for servers on z/OS.
- ▶ Run in development mode: Enable this option to streamline the startup time of an application server. Do *not* enable this setting on production servers.
- ▶ Parallel start: Select this field to start the server components, services, and applications on multiple threads. This action might shorten the startup time.

The order in which the applications start depends on the weights you assigned to each of them. Applications that have the same weight are started in parallel.

To set the weight of an application, in the administrative console, click **Applications** → **Application Types** → **WebSphere enterprise applications** → *application_name* → **Startup behavior** and then specify an appropriate value in the Startup order field.

- ▶ **Start components as needed:** Select this field to start the application server components as they are needed. This action might shorten the start time.
- ▶ **Access to internal server classes:** Specifies whether the applications can access many of the server implementation classes.
- ▶ **Application classloader policy and class loading mode:** These settings allow you to define an application server-specific classloader policy and class loading mode. Information about class loaders is provided in *Chapter 22, “Understanding class loaders” on page 789*.

Note: You also have the run in 64-bit JVM mode setting on z/OS, which provides additional virtual storage for user applications in 64-bit mode. Removing the check from this selection enables your server to start in 31-bit mode.

There is no interdependence between the modes in which you are running different servers. You can run some of your servers in 64-bit mode and some of your servers in 31-bit mode. However, eventually convert all of your servers to run in 64-bit mode because support for running servers in 31-bit mode is deprecated.

Container settings

Each application server has containers that run specific application components. This section in the configuration page for the server provides links to pages where you can modify the settings for the containers.

Tip: Modifying container settings is not something you normally do on a daily basis. Information about the most commonly used settings in these sections is provided throughout this book with the appropriate topics. Each container has settings that allow you to modify its configuration.

Session management

The session management settings allow you to manage HTTP session support, which includes specifying a session tracking mechanism, setting maximum in-memory session count, controlling overview, and configuring session timeout. Information about session management is provided in Chapter 28, “Session management” on page 961.

SIP container settings

Session Initiation Protocol (SIP) support extends the application server to allow it to run SIP applications written to the JSR 116 specification. SIP is used to establish, modify, and terminate multimedia IP sessions, including IP telephony, presence, and instant messaging. If you have SIP applications, review these settings.

Web container settings

The web container serves application requests for servlets and JSPs. The web container settings allow you to specify the default virtual host, enable servlet caching, disable servlet request and response caching, set the number of timeout thread and the default timeout, configure settings for using thread pools or a work manager to start runnable objects, specify session manager settings (such as persistence and tuning parameters), and HTTP transport properties.

The web container settings are shown in Figure 7-37.

[Application servers](#) > [AppSrv_85_01](#) > [Web container](#)

Use this page to configure the web container.

Configuration

General Properties

Default virtual host:
default_host

Enable servlet caching

Disable servlet request and response pooling

Asynchronous Servlet Properties

* Number of timeout threads
2

Set timeout
30000 milliseconds

Use thread pool to start Runnable objects

Use a work manager to start Runnable objects

* Work manager JNDI name:
wm/default

Apply OK Reset Cancel

Additional Properties

- [Asynchronous Request Dispatching](#)
- [Custom properties](#)
- [Web container transport chains](#)
- [Session management](#)

Figure 7-37 Web container settings

The Asynchronous Request Dispatcher (ARD) enables servlets and JSPs to make standard include calls concurrently on separate threads. Selecting this link allows you to enable ARD and configure related settings.

Portlet container services

The portlet container is the runtime environment for portlets using the JSR 168 Portlet Specification. Portlets based on this JSR 168 Portlet Specification are referred to as standard portlets. You can use these settings to enable portlet fragment caching to save the output of portlets to the dynamic cache.

EJB container properties

These properties allow you to configure the services provided by the EJB container, which include setting the passivation directory path, pool cleanup interval, a default data source JNDI name, and to enable stateful session bean failover using memory-to-memory replication. You can also configure EJB cache, EJB timer service settings and EJB asynchronous method invocation settings. Figure 7-38 on page 274 shows the EJB container properties window.

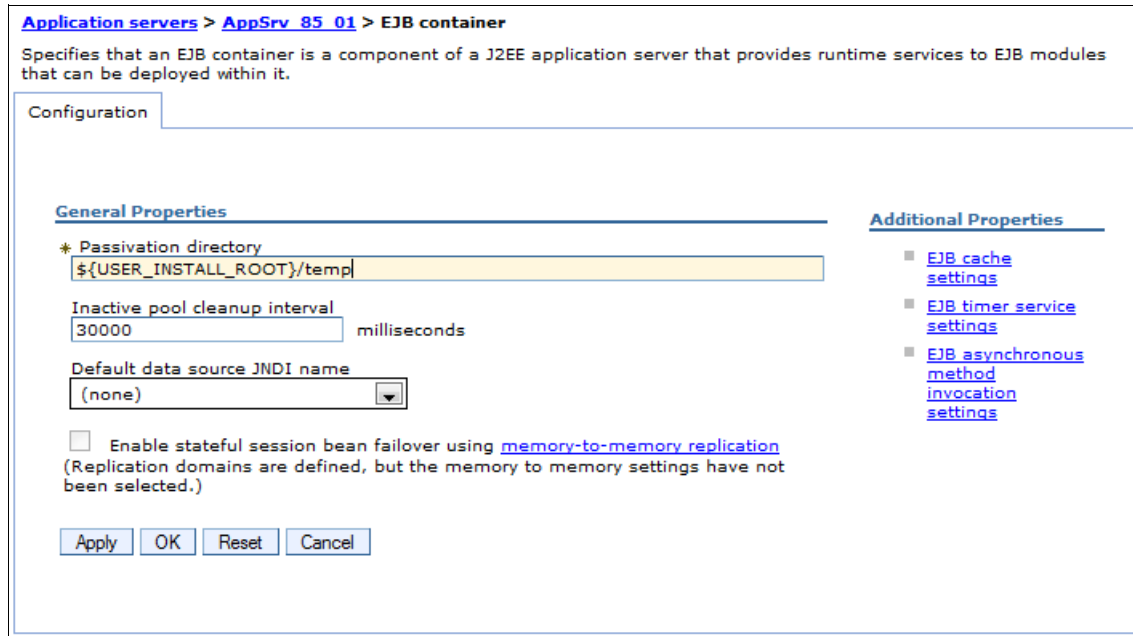


Figure 7-38 EJB container settings

Container services

The following settings are available under the container services section:

- ▶ Application profiling service: Application profiling is a WebSphere extension that, when used along with access intents, allows you to define strategies to dynamically control concurrency, pre-fetch, and read-ahead. The container services settings allows you to enable this service and to set the compatibility mode for J2EE 1.3 applications.
- ▶ Transaction service: The transaction service properties allow you to specify settings for the transaction service and manage active transaction locks. The settings include the directory location for the transaction service on the application server to store log files for recovery, the total transaction lifetime timeout, and client inactivity timeout.

When the application server is running, a Runtime tab is available in the Transaction Service properties workspace. From here, you can manage running transactions and modify timeout settings at run time.

- ▶ Dynamic cache service: This page allows you to specify settings for the dynamic cache service of this server.
- ▶ Compensation service: The compensation service supports server-level configuration for compensation enablement and logging. This service is not started automatically. If you plan to run applications that require this service, you must enable it here.
- ▶ Internationalization service: This service enables you to configure and manage an internationalization context for an application for which components are distributed across the enterprise. This section of the configuration window allows you to enable this service. It is not enabled by default.
- ▶ Default Java Persistence API settings: JPA provides a mechanism for managing persistence and object-relational mapping and functions for the EJB 3.0 specifications. This page allows you to configure default settings for JPA. JPA settings in an application override these settings.
- ▶ Object pool service: The object pool service manages object pool resources that are used by the application server. This section of the configuration window allows you to disable this service (it is enabled by default).

- ▶ ORB service: These settings allow you to specify settings for the Object Request Broker service. These include request timeout, thread settings, and connection cache minimum and maximum.
- ▶ Startup beans service: Startup beans are session beans that run business logic through the invocation of start and stop methods when applications start and stop. This section of the configuration window allows you to enable this service (it is disabled by default).

Business process services

The business process settings allow you to manage the following features:

- ▶ Activity session service
- ▶ Work area partition service
- ▶ Work area service

Applications

Use the **Installed Applications** link to view the applications installed on this server. This link will display the collection of applications as links to the configuration page for each application.

Server messaging

The server messaging settings provide configuration settings and information for the messaging services.

Server infrastructure

The server infrastructure settings include settings for Java and process management and administration services:

- ▶ Java and Process Management:
 - Class loader: Creates and configures class loader instances. Information about class loaders is provided in *Chapter 22, “Understanding class loaders” on page 789*.
 - Process definition: Defines runtime properties, such as the program to run, arguments to run the program, and the working directory. Within the process definitions, are the JVM definitions, such as the initial and maximum heap sizes, debug options, the process class path, or different runtime options, such as profiler support and heap size.
 - Process execution: Includes settings, such as the process priority or the user and group to be used to run the process. These settings are not applicable on the Windows platform.
 - Monitoring policy: Determines how the node agent monitors the application server. It includes ping intervals, timeouts, and an initial state setting. These settings can be used to ensure that the server is started when the node starts and is restarted in the event of a failure.
- ▶ Administration:
 - Custom properties: Specifies additional custom properties for this component.
 - Administration services: This group of settings allows you to specify various settings for administration facility for this server, such as administrative communication protocol settings and timeouts. (These settings are not something with which you are normally concerned.)
 - Server components: Creates additional runtime components that are configurable.
 - Custom services: Creates custom service classes that run within this server and their configuration properties.

If you plan to extend the administration services by adding custom MBeans, refer to the topic *Extending WebSphere Application Server Administrative System with custom MBeans* in the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.mutiplatform.doc/ae/tjmx_extend.html

- ▶ **Java SDKs:** This option lets you view and chose the required version of SDKs for your application server process. WebSphere Application Server V8.5 lets you select between Java SDK versions 1.6 and 1.7, both of them available for 32 and 64-bits systems. The default version that is selected when installing is Java SDK 1.6. If both versions are installed on your system(Figure 7-39), you can chose one of them and set it as default by clicking the **Make Default** button. Save your changes and restart the application server process to use the newly selected JAVA SDK version.

Application servers > AppSrv_85_01 > Java SDKs

This page lists the software development kits (SDKs) that are installed on the server. These SDKs are available to the servers.

⊕ Preferences

Make Default

⊞ ⊞ ⊞ ⊞

| Select | Name | Version | Location | Bits | Default |
|---|--------|---------|----------------------------------|------|---------|
| You can administer the following resources: | | | | | |
| <input type="checkbox"/> | 1.6_64 | 1.6 | \${WAS_INSTALL_ROOT}/java | 64 | false |
| <input type="checkbox"/> | 1.7_64 | 1.7 | \${WAS_INSTALL_ROOT}/java_1.7_64 | 64 | true |
| Total 2 | | | | | |

Figure 7-39 Application server JAVA SDKs

Communications

The following communications settings are available:

- ▶ **Ports:** These settings contain the basic port definitions for the server, which are shown in Figure 7-40 on page 277.

| Port Name | Port | Details |
|---------------------------------------|-------|---------|
| BOOTSTRAP_ADDRESS | 9813 | |
| SOAP_CONNECTOR_ADDRESS | 8886 | |
| ORB_LISTENER_ADDRESS | 9106 | |
| SAS_SSL_SERVERAUTH_LISTENER_ADDRESS | 9423 | |
| CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS | 9424 | |
| CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS | 9425 | |
| WC_adminhost | 9066 | |
| WC_defaulthost | 9084 | |
| DCS_UNICAST_ADDRESS | 9360 | |
| WC_adminhost_secure | 9049 | |
| WC_defaulthost_secure | 9447 | |
| SIP_DEFAULTHOST | 5068 | |
| SIP_DEFAULTHOST_SECURE | 5069 | |
| OVERLAY_UDP_LISTENER_ADDRESS | 11007 | |
| OVERLAY_TCP_LISTENER_ADDRESS | 11008 | |
| IPC_CONNECTOR_ADDRESS | 9638 | |
| SIB_ENDPOINT_ADDRESS | 7282 | |
| SIB_ENDPOINT_SECURE_ADDRESS | 7290 | |
| SIB_MQ_ENDPOINT_ADDRESS | 5562 | |
| SIB_MQ_ENDPOINT_SECURE_ADDRESS | 5582 | |

Figure 7-40 Viewing application server ports

The WebSphere Application Server V8.5 provides two new ports for the application server:

- **OVERLAY_UDP_LISTENER_ADDRESS**: Used for peer-to-peer (P2P) communication. The On Demand Configuration and asynchronous PMI components use P2P as their transport. This port is required by every WebSphere Extended Deployment process. The Default Value (incremented for multiple processes) is 11001.
- **OVERLAY_TCP_LISTENER_ADDRESS**: Used for P2P communication. The On Demand Configuration and asynchronous PMI components use P2P as their transport. This port is required by every WebSphere Extended Deployment process. The Default Value (incremented for multiple processes) is 11002.

Tip: You might not ever need to manually change these ports. It is likely, however, that you will want to view them. For example, if you use the `dumpNameSpace` command, you can specify the bootstrap port of the process from which to dump the name space. When you federate a node, you must know the SOAP connector port of the node or deployment manager. The inbound communications ports are essential for accessing applications and the administrative console.

Click **Details** to go to a page that has links for the configurable port settings.

Some port settings are defined to use the channel framework. These settings have an associated transport chain that represents the network protocol stack. A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP or HTTP. Network ports can be shared among all of the channels within a chain. The Channel Framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

- ▶ **Message listener service**

The message listener service provides the message-driven bean (MDB) listening process, in which message-driven beans are deployed against listener ports that define the JMS destination to listen upon. These listener ports are defined within this service along with settings for its thread pool.

- ▶ **Communications Enabled Applications (CEA)**

The Communication Enabled Applications setting provides the ability to add dynamic web communications to any application or business process. You can enable this setting and then configure the Representational State Transfer (REST) interface and the computer-telephony interaction (CTI) gateway.

Performance

These settings allow you to specify settings for the Performance Monitoring Infrastructure (PMI) and the Performance and Diagnostic Advisor Configuration framework. These performance monitoring settings are covered in *Chapter 16, "Monitoring distributed systems" on page 553*.

Security

Security settings for the application server allow you to set specific settings at the server level. Security settings are covered in *WebSphere Application Server V8 Security Guide, SG24-7971*.

Troubleshooting

These settings include the ones for logging and tracing. For information about troubleshooting and using these settings, see *WebSphere Application Server V6: Diagnostic Data, REDP-4085*.

Additional properties

The following settings are defined under the additional properties section:

- ▶ **Class loader viewer service:** This service is used to enable or disable the service that keeps track of classes that are loaded.
- ▶ **Core group service:** These settings are related to high availability.
- ▶ **Endpoint listeners:** An endpoint listener receives requests from service requester applications within a specific application server or cluster.
- ▶ **Debugging service:** On this page, you can specify settings for the debugging service to be used in conjunction with a workspace debugging client application, for example, the Application Server Toolkit.
- ▶ **Thread pools:** The thread pool specifies the possible maximum number of concurrently running threads in the web container. Because one thread is needed for every client request, this setting directly relates to the number of active clients that can possibly access the web container on this application server at any given time. A timeout value can be specified for the application server to remove threads from the pool based on a timed period of inactivity.

Finally, an option for creating threads beyond the maximum pool size is available. Be careful when using this option. It can have the unexpected effect of allowing the web container to create more threads than the JVM might be able to process, creating a resource shortage and bringing the application server to a halt. Figure 7-41 on page 279 shows the default thread pools.

Application servers > AppSrv_85_01 > Thread pools

Use this page to specify a thread pool for the server to use. A thread pool enables server components to reuse threads instead of creating new threads at run time. Creating new threads is typically a time and resource intensive operation.

⊕ Preferences

New... Delete

⊞ ⊞ ⊞ ⊞ ⊞ ⊞

| Select | Name | Description | Minimum Size | Maximum Size |
|---|---|--|--------------|--------------|
| You can administer the following resources: | | | | |
| <input type="checkbox"/> | Default | | 20 | 20 |
| <input type="checkbox"/> | ORB.thread.pool | | 10 | 50 |
| <input type="checkbox"/> | SIBFAPInboundThreadPool | Service integration bus FAP inbound channel thread pool | 4 | 50 |
| <input type="checkbox"/> | SIBFAPThreadPool | Service integration bus FAP outbound channel thread pool | 4 | 50 |
| <input type="checkbox"/> | SIBJMSRAThreadPool | Service Integration Bus JMS Resource Adapter thread pool | 35 | 41 |
| <input type="checkbox"/> | TCPChannel.DCS | | 20 | 20 |
| <input type="checkbox"/> | WMQJCAResourceAdapter | WebSphere MQ Resource Adapter thread pool | 10 | 50 |
| <input type="checkbox"/> | WebContainer | | 50 | 50 |
| <input type="checkbox"/> | server.startup | This pool is used by WebSphere during server startup. | 1 | 3 |
| Total 9 | | | | |

Figure 7-41 Default thread pools in an application server

- ▶ Reliable messaging state: This link allows you to view and manage the WS-ReliableMessaging runtime state.
- ▶ Web server plug-in properties: This setting is used to change the HTTP plug-in configuration without having to stop the server and start it again.

7.4.7 Repository checkpoints service

The extended repository service enables advanced management of the configuration repository. The configuration repository contains the configuration for the cell. This information is essential to the operation of your applications. You can create repository checkpoints to help you save snapshots of your configuration as you make changes so that you can easily undo those changes if necessary. You can configure your repository to create automatic delta checkpoints each time you make a configuration change. A delta checkpoint saves a copy of the configuration documents prior to saving your changes. You can specify the number of automatic checkpoints to save. After this limit is reached, the next checkpoint replaces the oldest.

To configure the repository checkpoints service:

1. Log into the administrative console, and click **System administration** → **Extended Repository Service** (Figure 7-42 on page 280).

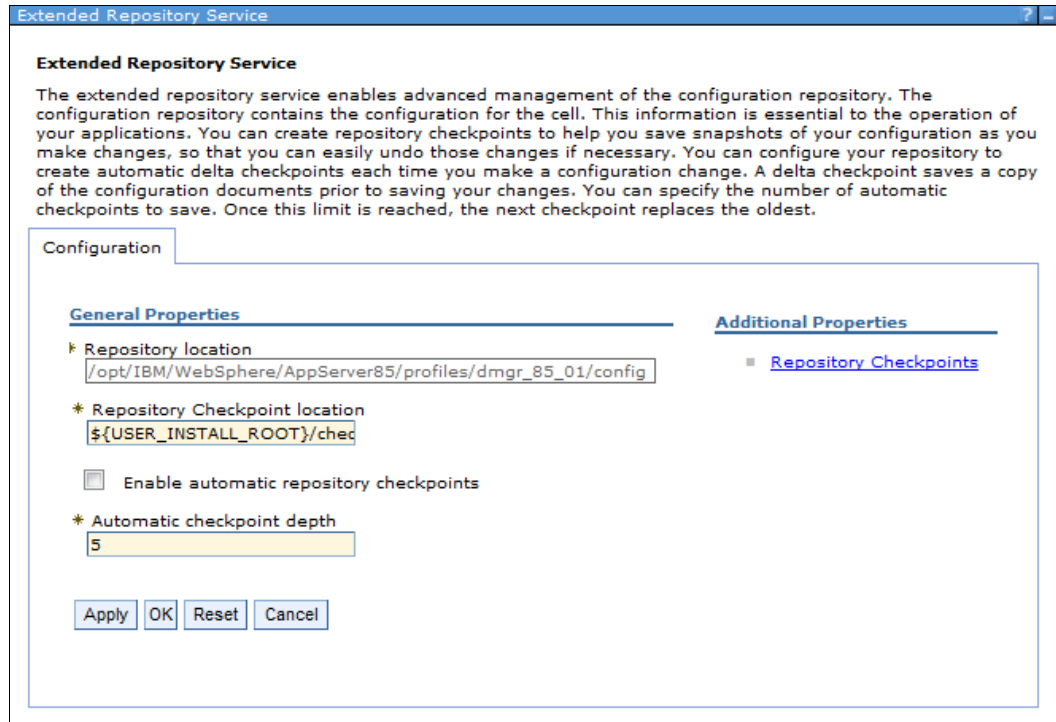


Figure 7-42 Extended Repository Service

2. If you want to enable automatic checkpoints, select the **Enable automatic repository checkpoints** option and then click **Apply**. To create a single checkpoint, select **Additional Properties** → **Repository checkpoints** → **New**. Next, enter a name for the checkpoint, and click **OK**. A full repository checkpoint is created.
3. Click **System administration** → **Extended Repository Service** → **Repository checkpoints** and a list of available checkpoints are displayed (Figure 7-43 on page 281).

| Select | Name | Documents | Type | Sequence | Timestamp | Description |
|---|--|-----------|-------|---------------|-------------------------|------------------------------|
| You can administer the following resources: | | | | | | |
| <input type="checkbox"/> | Delta-1340326633507 | 1 | DELTA | 1340326633507 | Jun 21, 2012 6:57:13 PM | Autosave delta image |
| <input type="checkbox"/> | Delta-1340326643478 | 1 | DELTA | 1340326643478 | Jun 21, 2012 6:57:23 PM | Autosave delta image |
| <input type="checkbox"/> | Delta-1340326697287 | 1 | DELTA | 1340326697287 | Jun 21, 2012 6:58:17 PM | Autosave delta image |
| <input type="checkbox"/> | Delta-1340327023566 | 36 | DELTA | 1340327023566 | Jun 21, 2012 7:03:43 PM | Autosave delta image |
| <input type="checkbox"/> | Delta-1340327371909 | 1 | DELTA | 1340327371909 | Jun 21, 2012 7:09:31 PM | Autosave delta image |
| <input type="checkbox"/> | Delta-1340327395212 | 1 | DELTA | 1340327395212 | Jun 21, 2012 7:09:55 PM | Autosave delta image |
| <input type="checkbox"/> | major update checkpoint1 | 594 | FULL | 1340333165628 | Jun 21, 2012 8:46:05 PM | checkpoint before app update |
| Total 7 | | | | | | |

Figure 7-43 Repository checkpoints

Other checkpoint options that are available are adding, deleting, exporting as compressed files (only for delta checkpoints), and viewing the contents by clicking the name of the checkpoint (Figure 7-44).

Extended Repository Service > Repository Checkpoints > major_update_checkpoint1

A repository checkpoint comprises a set of configuration documents saved before a configuration change was made. The set of documents saved in this checkpoint are available for inspection below.

Attributes

| | | | |
|------|---------------|-------------------------|------------------------------|
| Type | Sequence | Timestamp | Description |
| FULL | 1340333165628 | Jun 21, 2012 8:46:05 PM | checkpoint before app update |

Preferences

| Document | URI |
|--|--|
| You can administer the following resources: | |
| AppServiceGroup.wsdl | cells/aix1_Cell_85_01/applications/WebSphereWSDM.ear/deployments/WebSphereWSDM/WebSphereWSDM.war/WEB-INF/classes/wsdl/AppServiceGroup.wsdl |
| Application.wsdl | cells/aix1_Cell_85_01/applications/WebSphereWSDM.ear/deployments/WebSphereWSDM/WebSphereWSDM.war/WEB-INF/classes/wsdl/Application.wsdl |
| ApplicationServer.wsdl | cells/aix1_Cell_85_01/applications/WebSphereWSDM.ear/deployments/WebSphereWSDM/WebSphereWSDM.war/WEB-INF/classes/wsdl/ApplicationServer.wsdl |
| CeaNotificationConsumer.wsdl | cells/aix1_Cell_85_01/applications/commsvc.ear/deployments/commsvc/commsvc.rest.war/WEB-INF/wsdl/CeaNotificationConsumer.wsdl |
| ControllerService.wsdl | cells/aix1_Cell_85_01/applications/commsvc.ear/deployments/commsvc/commsvc.rest.war/WEB-INF/wsdl/ControllerService.wsdl |

Page: 1 of 119 Total 594

Figure 7-44 Repository checkpoint contents

For more information about Repository checkpoints service, refer to the *IBM WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022-00.

7.5 Working with nodes in a Network Deployment environment

Managing nodes is a concept specific to a Network Deployment environment. Nodes are managed by the deployment manager through a process known as a *node agent* that resides on each node. To manage a node in a Network Deployment environment, the node must be defined, and the node agent on each WebSphere Application Server node must be started.

Nodes are created when you create a profile. Nodes are added to a cell through federation (See 3.3.7, “*Federating nodes to a cell*” on page 83 for more details).

7.5.1 Starting and stopping nodes

A node consists of the node agent and the servers. There are several ways to start and stop a node and node agent or to stop them individually. Before using any of these methods, be sure to note whether it affects the entire node, including servers, or just the node agent.

In this section, we cover the following topics:

- ▶ Starting a node agent
- ▶ Starting a node on z/OS using the START command
- ▶ Stopping a node agent
- ▶ Stopping a node on z/OS using the STOP command
- ▶ Stopping a node (the node agent and servers)
- ▶ Restarting a node agent

Starting a node agent

When a node agent is stopped, the deployment manager has no way to communicate with it. Therefore, the node agent cannot be started using the administrative console and has to be started with the **startNode** command run from the profile node system.

startNode command

The syntax of the **startNode** command is:

```
startNode.bat(sh) [options]
```

The options are shown in Example 7-11.

Example 7-11 startNode command

```
Usage: startNode [options]
      options: -nowait
              -quiet
              -logfile <filename>
              -replacelog
              -trace
              -script [<script filename>] [-background]
              -timeout <seconds>
              -statusport <portnumber>
              -profileName <profile>
              -recovery
              -help
```

For information about the use of the **startNode** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rxml_startnode.html

See Example 7-12 for use of the **startNode** command. Note that a user ID and password are not required.

Example 7-12 startNode command example for a node agent: AIX operating system

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv85_01/bin/startNode.sh
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv85_01/logs/nodeagent/startServer.log
ADMU0128I: Starting tool with the AppSrv85_01 profile
ADMU3100I: Reading configuration for server: nodeagent
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server nodeagent open for e-business; process id is 712746
```

Starting a node on z/OS using the START command

To start a node agent on z/OS using the **START** command, use the following format:

```
START nodeagent_procname, JOBNAME=server_shortcode,
ENV=cell_shortcode.node_shortcode.server_shortcode
```

For example:

```
START WPACRA, JOBNAME=WPAGNTA, ENV=WPCELL.WPNODEA.WPAGNTA
```

Stopping a node agent

To stop the node agent and leave the servers running, complete the following actions, depending on your preferred method. You can use the administrative console or a command prompt.

To stop a node from the administrative console:

1. In the administrative console, click **System administration** → **Node agents**.
2. Select the box beside the node agent for the server, and click **Stop**.

To stop a node using a command prompt:

1. Open a command window.
2. Enter the **stopNode** command.

Note: After you stop the node agent, the deployment manager has no way to communicate with the servers on that node. The servers might be up and running, but the administrative console cannot determine their status.

stopNode command

The syntax of the **stopNode** command is:

```
stopNode.bat(sh) [options]
```

The options are shown in Example 7-13.

Example 7-13 The stopNode command

```
Usage: stopNode [options]
```

```
options: -nowait
         -stopservers [-saveNodeState]
         -quiet
         -logfile <filename>
         -replacelog
         -trace
         -timeout <seconds>
         -statusport <portnumber>
         -conntype <connector type>
         -port <portnumber>
         -username <username>
         -password <password>
         -profileName <profile>
         -help
```

For more information about the **stopNode** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rxml_stopnode.html

See Example 7-14 for a sample output of the **stopNode** command.

Example 7-14 stopNode command

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv85_01/bin/stopNode.sh -username
admin85 -password admin85
ADMU0116I: Tool information is being logged in file
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/logs/nodeagent/stopServ
ADMU0128I: Starting tool with the AppSrv_85_01 profile
ADMU3100I: Reading configuration for server: nodeagent
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server nodeagent stop completed.
```

Stopping a node on z/OS using the STOP command

To stop a node agent on z/OS, use the following command:

```
STOP nodeagent_JOBNAME
```

For example:

```
STOP WPAGNTA
```

Stopping a node (the node agent and servers)

You can use the administrative console to stop a node and its servers with one action. Complete the following steps:

1. From the administrative console, click **System administration** → **Nodes**.
2. Select the box beside the node, and click **Stop**.

Restarting a node agent

You can restart a running node agent from the administrative console by completing the following steps from the administrative console:

1. Click **System administration** → **Node agents**.
2. Select the box beside the node agent for the server, and click **Restart**.

7.5.2 Node agent synchronization

During a synchronization operation, a node agent checks with the deployment manager to see if any configuration documents that apply to the node were updated. New or updated documents are copied to the node repository, and deleted documents are removed from the node repository.

Automatic synchronization

Automatic configuration synchronization between the node and the deployment manager is enabled by default. You can configure the interval between synchronizations in the administrative console by completing the following steps:

1. Expand **System administration** → **Node agents** in the administrative console.
2. Select the node agent process on the appropriate server to open the Properties page.
3. In the Additional Properties section, click **File synchronization service**.
4. Configure the synchronization interval. By default, the synchronization interval is set to one minute.

The default synchronization interval on z/OS is five minutes.

Tip: Increase the synchronization interval in a production environment to reduce the impact of system usage.

Note that a separate setting about synchronization exists as an administrative console preference. The *Synchronize changes with Nodes* option, when selected, indicates that any time a change is saved to the console, it is automatically synchronized out to the running nodes. To set this preference, click **System administration** → **Console Preferences**, and select the **Synchronize changes with Nodes** option, as shown in Figure 7-45.

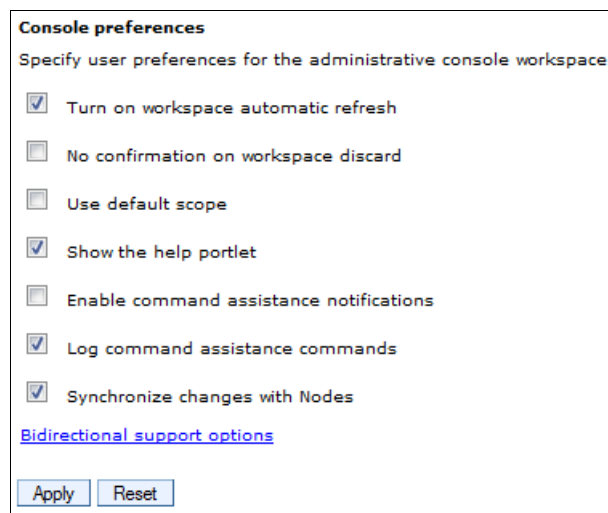


Figure 7-45 Administrative console preferences (including synchronization)

Forced synchronization

Synchronization can be forced by clicking **System administration** → **Nodes**. Select the box beside a node, and click **Synchronize** or **Full Synchronization**:

- ▶ Synchronize performs an immediate synchronization on the selected node. This type of synchronization is optimized for performance and only synchronizes changed files. If there

are issues with manually edited files, this action might not result in a complete synchronization.

- ▶ The Full Synchronization option disregards optimization and ensures that the node and cell configuration are identical.

Using the syncNode command

The **syncNode** command can be used from the node to force the synchronization of a node's local configuration repository with the master repository on the deployment manager node.

Tip: The **syncNode** command is normally only used in exception situations. To use the **syncNode** command, the node agent must be stopped. You can use the **-stopservers** and **-restart** options on the **syncNode** command to stop the node agent and application servers, and then restart the node agent.

The syntax of the **syncNode** command is:

```
syncNode.bat(sh) [options]
```

The options are shown in Example 7-15.

Example 7-15 syncNode command

```
Usage: syncNode dmgr_host [dmgr_port] [-conntype <type>] [-stopservers]
      [-restart] [-quiet] [-nowait] [-logfile <filename>] [-replacelog]
      [-trace] [-username <username>] [-password <password>]
      [-localusername <localusername>] [-localpassword <localpassword>]
      [-profileName <profile>] [-help]
```

For more information about the **syncNode** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_syncnode.html

Example 7-16 shows the **syncNode** command and options. The command is executed from the node. The **-stopservers** and **-restart** options are used to stop all the servers on the node, including the node agent, and then restart the node agent after the synchronization.

Example 7-16 syncNode usage examples

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/syncNode.sh saw211-sys1
8884 -stopservers -restart -username admin85 -password admin85
ADMU0116I: Tool information is being logged in file
          /opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/logs/syncNode.log
ADMU0128I: Starting tool with the AppSrv_85_01 profile
ADMU0401I: Begin syncNode operation for node saw211-sys1Node01 with Deployment
          Manager saw211-sys1: 8884
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: AppSrv_85_01
ADMU0506I: Server name: ODR_85_1
ADMU0506I: Server name: AppSrv_85_02
ADMU0506I: Server name: AppSrv_85_03
ADMU2010I: Stopping all server processes for node saw211-sys1Node01
ADMU0510I: Server AppSrv_85_01 is now STOPPED
ADMU0512I: Server ODR_85_1 cannot be reached. It appears to be stopped.
```

```
ADMU0512I: Server AppSrv_85_02 cannot be reached. It appears to be stopped.
ADMU0512I: Server AppSrv_85_03 cannot be reached. It appears to be stopped.
ADMU0510I: Server nodeagent is now STOPPED
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: saw211-sys1Node01
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is:
          737330
ADMU0402I: The configuration for node saw211-sys1Node01 has been synchronized
          with Deployment Manager saw211-sys1: 8884
```

7.5.3 Removing a node from a cell

There are two ways to remove a node from a network distributed administration cell.

Note: When a node is removed, it is restored to its original configuration.

Using the administrative console

From the administrative console, complete the following steps:

1. Click **System administration** → **Nodes**.
2. Select the check box beside the node you want to remove, and click **Remove Node**.

This method runs the **removeNode** command in the background.

Using the removeNode command

The **removeNode** command detaches a node from a cell and returns it to a stand-alone configuration.

The syntax of the **removeNode** command is:

```
removeNode.bat(sh) [options]
```

The options are shown in Example 7-17.

Example 7-17 removeNode command

```
Usage: removeNode [-force] [-quiet] [-nowait] [-statusport <port>] [-logfile
<filename>]
          [-replacelog] [-trace] [-username <username>] [-password <password>]
          [-profileName <profile>] [-help]
```

For more information about the **removeNode** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_removeNode.html

To use the command:

1. Change the directory to the *profile_root/bin* directory.
2. Run **removeNode**. All parameters are optional for this command.

On z/OS, the **removeNode.sh** command is in the *install_root/bin* directory. You need to specify the **-profileName** parameter to specify the profile for the node you want to remove.

The command performs the following operations:

1. Connects to the deployment manager process to read the configuration data.
2. Stops all of the running server processes of the node, including the node agent process.
3. Removes servers in the node from clusters.
4. Restores the original stand-alone node configuration. This original configuration was backed up when the node was originally added to the cell.
5. Removes the node's configuration from the master repository of the cell. The local copy of the repository held on each node is updated at the next synchronization point for each node agent. Although the complete set of configuration files are not pushed out to other nodes, some directories and files are pushed out to all nodes.
6. Removes installed applications from application servers in the cell that are part of the node being removed.
7. Copies the original application server cell configuration into the active configuration.

The command provides the **-force** option to force the local node's configuration to be decoupled from the cell even if the deployment manager cannot be contacted. However, if this situation occurs, the cell's master repository then has to be separately updated to reflect the node's removal, for example, through manual editing of the master repository configuration files.

Example 7-18 shows an example of using the **removeNode** command.

Example 7-18 removeNode example

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/bin/removeNode.sh
-profileName AppSrv_85_01 -username admin85 -password admin85
ADMU0116I: Tool information is being logged in file

/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_85_01/logs/removeNode.log
ADMU0128I: Starting tool with the AppSrv_85_01 profile
ADMU2001I: Begin removal of node: saw211-sys1Node01
ADMU0009I: Successfully connected to Deployment Manager Server:
          saw211-sys1:8884
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: AppSrv_85_01
ADMU0506I: Server name: ODR_85_1
ADMU0506I: Server name: AppSrv_85_02
ADMU0506I: Server name: AppSrv_85_03
ADMU2010I: Stopping all server processes for node saw211-sys1Node01
ADMU0512I: Server AppSrv_85_01 cannot be reached. It appears to be stopped.
ADMU0512I: Server ODR_85_1 cannot be reached. It appears to be stopped.
ADMU0512I: Server AppSrv_85_02 cannot be reached. It appears to be stopped.
ADMU0512I: Server AppSrv_85_03 cannot be reached. It appears to be stopped.
ADMU0510I: Server nodeagent is now STOPPED
ADMU2021I: Removing all servers on this node from all clusters in the cell.
ADMU2014I: Restoring original configuration.
ADMU2017I: The local original configuration has been restored.
ADMU0306I: Note:
ADMU2031I: Any applications that were uploaded to the aix1_Cell_85_01 cell
          configuration during addNode using the -includeapps option are not
          uninstalled by removeNode.
ADMU0307I: You might want to:
```


ADMU2032I: Use wsadmin or the Administrative Console to uninstall any such applications from the Deployment Manager.

ADMU0306I: Note:

ADMU2033I: Any buses that were uploaded to the aix1_Cell_85_01 cell configuration during addNode using the -includebuses option are not uninstalled by removeNode.

ADMU0307I: You might want to:

ADMU2034I: Use wsadmin or the Administrative Console to uninstall any such buses from the Deployment Manager.

ADMU2024I: Removal of node saw211-sys1Node01 is complete.

7.5.4 Renaming a node

The **renameNode** command allows you to modify the node name of a federated server.

renameNode command

The syntax of the **renameNode** command is:

```
renameNode.bat(sh) [options]
```

The options are shown in Example 7-19.

Example 7-19 renameNode command syntax

```
Usage: renameNode dmgr_host dmgr_port node_name [-nodeshortname <name>]
        [-conntype <type>] [-logfile <filename>] [-trace]
        [-username <username>] [-password <password>] [-help]
```

For more information about the **renameNode** command, go to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_renamenode.html

To run the command, complete the following steps:

1. Change to the *profile_root/bin* directory of the deployment manager.
2. Run the **renameNode** command.

The command:

- ▶ Connects to the deployment manager.
- ▶ Stops all servers.
- ▶ Changes the node configuration on the deployment manager.
- ▶ Synchronizes the node.

7.5.5 Recovering an existing node

You can use the **-asExistingNode** option of the **addNode** command to recover and move nodes of a deployment manager. Using the **-asExistingNode** option, federates a new custom node to a deployment manager as an existing node. During federation, the product uses information in the deployment manager master configuration to transform the custom node into the existing node.

Other **addNode** options for node configuration are incompatible with the **-asExistingNode** option. Do not use **-asExistingNode** with the following incompatible options:

- ▶ **-includeapps**
- ▶ **-includebuses**
- ▶ **-startingport**
- ▶ **-portprops**
- ▶ **-nodeagentshortname**
- ▶ **-nodegroupname**
- ▶ **-registerservice**
- ▶ **-serviceusername**
- ▶ **-servicepassword**
- ▶ **-coregroupname**
- ▶ **-excludesecuritydomains**

As an example, let us assume that there is a node profile named `appsrv85` on a host named `aix1`, and the profile is federated to a deployment manager. The `appsrv85` profile was damaged and you have to recover it. To do this, the following actions must be followed:

1. Make sure that the node agent of `appsrv85` profile is stopped.
2. Remove the damaged node with **manageprofiles -delete** command from the WebSphere `install root` of `aix1`. If you need to recover the damaged node on another machine, the host name for that machine has to be `aix1`.
3. On host `aix1`, create a profile with the same profile path, profile name, and node name as the unavailable one.
4. Run the **addNode** command with the **-asExistingNode** option from a command line at the `bin` directory of the new profile. Here is the syntax for this command:

```
addNode dmgr_host dmgr_port -asExistingNode -username user_name -password password
```

For more information about the **-asExistingNode** option, refer to the IBM WebSphere Application Server V8.5 Concepts, Planning, and Design Guide, SG24-8022-00.

7.5.6 Node groups

In a Network Deployment environment, you can have nodes in a cell with different capabilities. However, there are restrictions on how the nodes can coexist.

Node groups are created to group nodes of similar capability together to allow validation during system administration processes. Effectively, this situation means that a node group establishes a boundary from which servers can be selected for a cluster. Nodes on distributed platforms and nodes on the IBM i platform can be members of the same node group, but they cannot be members of a node group that contains a node on a z/OS platform.

Node groups versus groups of nodes: Do not confuse node groups with “groups of nodes” in the job manager. These are two different concepts.

A default node group called `DefaultNodeGroup` is automatically created for you when the deployment manager is created, based on the deployment manager platform. New nodes on similar platforms are automatically added to the default group. A node must belong to at least one node group but can belong to more than one.

As long as you have nodes in a cell with similar platforms, you do not need to do anything with node groups. New nodes are automatically added to the node group. However, before adding

a node on a platform that does not have the same capabilities as the deployment manager platform, you must create the new node group.

Working with node groups

You can display the default node group and its members by clicking **System Administration** → **Node Groups**, as shown in Figure 7-46.

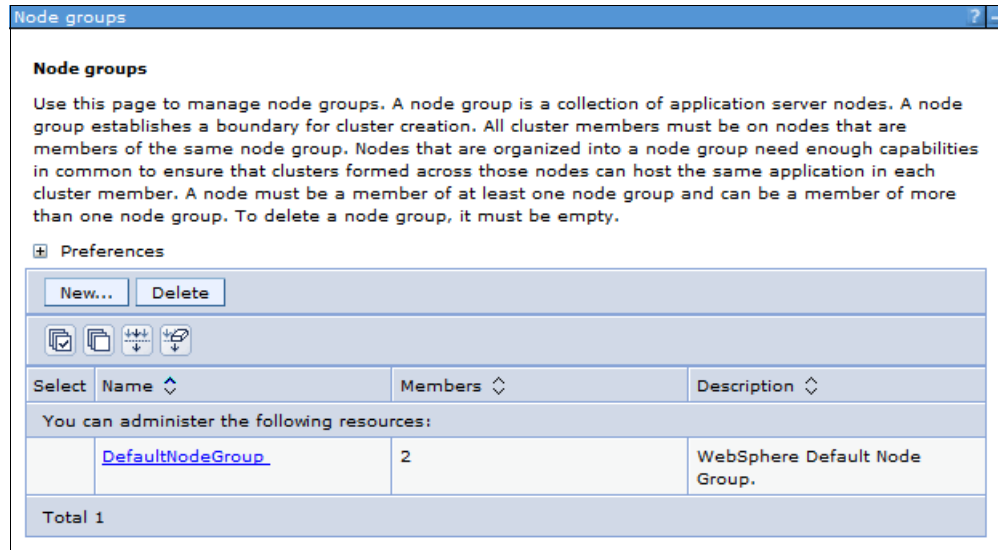


Figure 7-46 Display a list of node groups

Using Figure 7-46, you can perform a number of actions:

- ▶ To create a new node group, click **New**. The only thing that you need to enter is the name of the new node group. Click **OK**.
- ▶ To delete a node group, select the box to the left of the node group name, and click **Delete**.
- ▶ To display a node group, click the node group name. Figure 7-47 shows the DefaultNodeGroup.

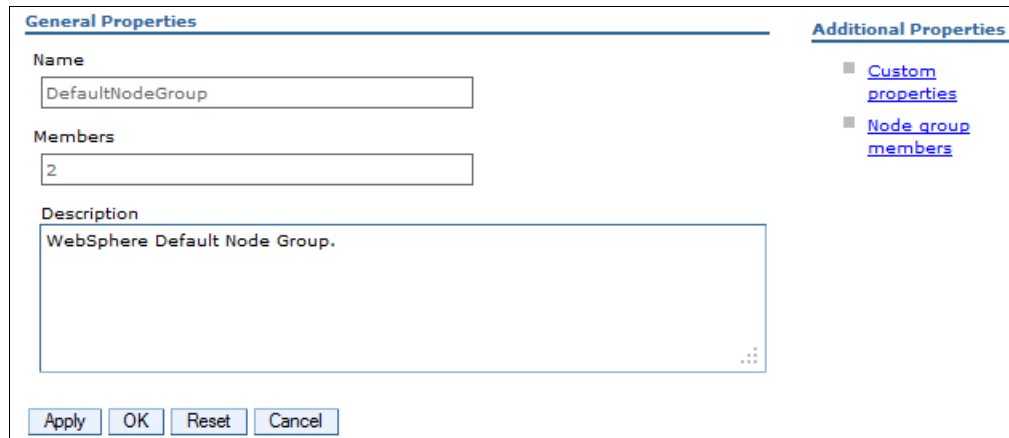


Figure 7-47 Node group general properties

- ▶ To add a node to a node group, display the node group, and click **Node group members** in the Additional Properties section. When the list appears, click **Add**. You can select from a list of nodes (Figure 7-48).

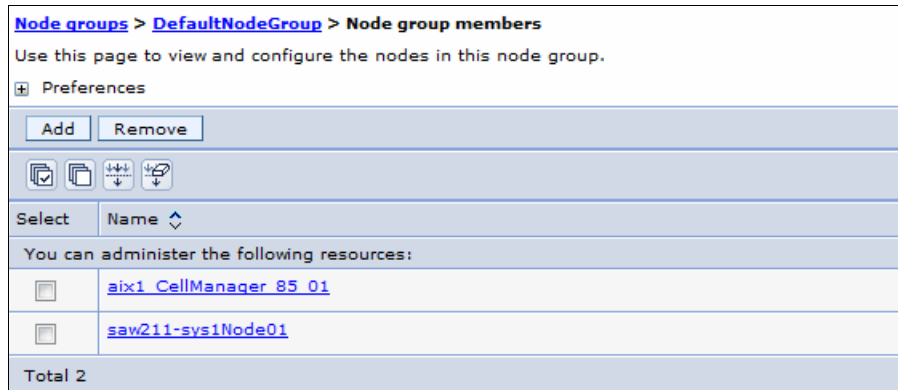


Figure 7-48 Displaying node group members

7.6 Working with clusters

This section provides information about creating, configuring, and managing application server clusters using the administrative console.

Clusters consist of one or more application servers that run the same applications. The configuration of each server can be unique.

Before creating a cluster, consider the number of servers you want to add to the cluster, the nodes on which they will be created, and how the workload is distributed across the servers.

Work is distributed across the servers in the cluster based on weights assigned to each application server. If all cluster members have identical weights, work is distributed among the cluster members equally. Servers with higher weight values are given more work. An example formula for determining routing preference is:

$$\% \text{ routed to Server1} = \text{weight}_1 / (\text{weight}_1 + \text{weight}_2 + \dots + \text{weight}_n)$$

In the formula, n represents the number of cluster members in the cluster. Consider the capacity of the system that hosts the application server.

7.6.1 Creating application server clusters

When you create a cluster, you have the option to create an empty cluster (no servers) or to create the cluster with one or more servers. The first application server added to the cluster acts as a template for subsequent servers. You can create the first server during the cluster creation process, or you can convert an existing application server. The rest of the servers must be new and can be created when you create the cluster or added later.

Tip: When creating a cluster, it is possible to select the template of an existing application server for the cluster without adding that application server into the new cluster. If you need to change the attributes of the servers in your cluster after the cluster is created, you must change each server individually. For this reason, consider creating an application server with the server properties that you want as a standard in the cluster first, then use that server as a template or as the first server in the cluster.

Cluster and cluster member options

When you create a new cluster, you have the following options to consider:

- ▶ Prefer local

This setting indicates that a request to an EJB must be routed to an EJB on the local node if available. This is the default setting and generally results in better performance.

- ▶ Configure HTTP session memory-to-memory replication (create a replication domain)

WebSphere Application Server supports session replication to another WebSphere Application Server instance. In this mode, sessions can replicate to one or more WebSphere Application Server instances to address HTTP Session single point of failure.

When you create a cluster, you can elect whether to create a replication domain for the cluster. The replication domain is given the same name as the cluster and is configured with the default settings for a replication domain. When the default settings are in effect, a single replica is created for each piece of data, and encryption is disabled. Also, the web container for each cluster member is configured for memory-to-memory replication.

For more information about replication domains, refer to 28.3.2, “Persistent sessions management” on page 973.

When you create a new cluster member, you have the following options to consider:

- ▶ Basis for first cluster member

You can add application servers to the cluster when you create the cluster or later.

The first cluster member can be a new application server or you can convert an existing application server so that it becomes the first cluster member.

Subsequent application servers in the cluster must be created new. The first application server in the cluster acts as a template for the subsequent servers.

The options you have depend on how you create the cluster.

When you use the job manager, you have the option to convert an existing server to use as the first cluster member, or create an empty cluster and run additional jobs to add cluster members.

When you use the deployment manager, you can convert an existing server, create one or more new servers, or create an empty cluster.

Note: The option to use an existing application server does not appear in the deployment manager administrative console if you create an empty cluster and then add a member later. If you want to convert an existing application server as the first server, specify that option when you create the cluster or use the job manager to create the cluster member.

Tip: To remove a server from a cluster, you must delete the server. Take this situation into consideration when you are determining whether to convert an existing server to a cluster.

- ▶ Server weight for each cluster member

The weight value controls the amount of work that is directed to the application server. If the weight value for this server is greater than the weight values that are assigned to other servers in the cluster, this server receives a larger share of the workload. The weight value represents a relative proportion of the workload that is assigned to a particular application server. The value can range from 0 to 20.

Member weight: Specify the relative weight of this server in the cluster. Values are from 0 to 20. A 0 indicates that work is to be routed to this server only in the event that no other servers are available.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system:

- For HTTP requests, weights are used to distribute HTTP traffic between the web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that is to receive the HTTP traffic.
- For web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Using the deployment manager administrative console

To create a new cluster:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click **New**.
3. Enter the information for the new cluster (see Figure 7-49):
 - Enter a cluster name of your choice.
 - On z/OS, you are also asked for the short name for the cluster.

The screenshot shows a window titled "Create a new cluster" with a progress indicator on the left. The progress indicator has four steps: "Step 1: Enter basic cluster information" (highlighted with a blue arrow), "Step 2: Create first cluster member", "Step 3: Create additional cluster members", and "Step 4: Summary". The main content area is titled "Enter basic cluster information" and contains a text input field for "Cluster name" with the value "WAS_85_aix_cluster". Below the text field are two checkboxes: "Prefer local. Specifies whether enterprise bean requests will be routed to the node on which the client resides when possible." (checked) and "Configure HTTP session memory-to-memory replication" (unchecked). At the bottom of the dialog are "Next" and "Cancel" buttons.

Figure 7-49 Creating a new cluster

4. Create first cluster member: The first cluster member determines the server settings for the cluster members (Figure 7-50).

Figure 7-50 First cluster member

The fields are:

- Member Name: Enter the name of the new server to be added to the cluster. On z/OS, you are also asked for the short name for the server.
- Select Node: Specifies the node on which this new cluster member is created.
- Server weight: Assigns the weight for this server.
- Generate unique HTTP ports: Generates unique port numbers for every transport that is defined in the source server, so that the resulting server that is created does not have transports that conflict with the original server or any other servers defined on the same node.
- Core Group: Because multiple core groups exist, you must select the core group for the cluster members to join. This field only displays if you have multiple core groups defined.

- Select how the server resources are promoted in the cluster: Specifies how resources, such as data sources, are initially created in the cluster. This option is only available for the first cluster member. All other cluster members are based on the cluster member template, which is created from the first cluster member. You can select from the following options:
 - Cluster, which indicates the resources defined can be used across all cluster members. This setting reduces the amount of configuration and management of resources. The cluster option is the default setting.
 - Server, which indicates that the resources are defined at the cluster member level. This setting is useful if you want to have different configuration settings for resources defined on each cluster member.
 - Both, which copies the resources of the cluster member (server) level to the cluster level.
- Select basis for first cluster member:
 - If you click **Create the member using an application server template**, the settings for the new application server are identical to the settings of the application server template you select from the list of available templates.
 - If you click **Create the member using an existing application server as a template**, the settings for the new application server are identical to the settings of the application server you select from the list of existing application servers. However, applications that are installed on the template server are not installed on the new servers.
 - If you click **Create the member by converting an existing application server**, the application server you select from the list of available application servers becomes a member of this cluster.

Applications that are installed on the existing server are automatically installed as new members of the cluster.

Note that the only way to remove a server from a cluster is to delete the server. If you delete the cluster, all servers in the cluster are deleted.
 - If you click **None. Create an empty cluster**, a new cluster is created, but it does not contain any cluster members.

Click **Next**.

5. Create additional cluster members: Use this window to create additional members for a cluster. You can add a member to a cluster when you create the cluster or after you create the cluster. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

To add a member, enter a new server name, select the node, and click **Add Member**. The new member is added to the list, as shown in Figure 7-51.

Figure 7-51 Additional cluster members

6. When all the servers are entered, click **Next**. A summary window shows you what will be created.
7. Click **Finish** to create the cluster and new servers.
8. Save the configuration.

Adding additional servers to the cluster

To add a server using the administrative console:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click the cluster name.
3. Under the Additional Properties sections, click **Cluster members**.
4. Click **New**. This action opens the same configuration window that was used when you created the cluster (Figure 7-48 on page 292).
5. Enter the name of the new server to create, select the node, and select the options to use. Click **Add Member**.
6. Click **Next** and then click **Finish**.

When an application server is added as a member to a server cluster, the modules installed on other members are also installed on the new member. You do not need to re-install or upgrade the application.

Using cluster member templates

When you created your cluster's servers, a server template was created for the cluster by copying the first cluster member's configuration. This template is then used when you create additional servers for that cluster. This situation is important to understand to get the results you expect when working with clusters.

To view a cluster's member templates using the administrative console, complete the following steps:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click the cluster name.
3. Under the Additional Properties sections, click **Cluster members**.
4. Click **Templates**

These steps open a configuration window that lists the templates in this cluster. Typically, you only have one template, but you have additional templates if the cluster includes servers that are at different versions of WebSphere Application Server (versions 8.5, 8, or 7).

From this configuration window, you can view and modify the server attributes of the template. If you modify the attributes, it is important to understand that existing cluster members are not affected. The template is only used for creating new cluster members.

To modify the attributes of a cluster's member using the administrative console:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click the cluster name.
3. Under the Additional Properties sections, click **Cluster members**.
4. Click the cluster member to open the server configuration window where you can make your change.

If you want to make the same change to multiple cluster members, you must repeat these steps. Also, modify the same attributes in the cluster member template because new cluster members are created based on the template. If you do not change the cluster's template(s), additional cluster members do not match the existing members.

If you want to modify the same server attribute(s) across all of a cluster's members, and you have several cluster members, one way to accomplish this task is by using the administrative console and completing the following steps:

1. Navigate to the cluster's templates.
2. Click the cluster template and make your desired changes.
3. Save your changes.
4. Delete all of the members in the cluster.
5. Recreate the members.

The new members are created from the updated cluster member template, and all of the cluster members have the same configuration.

Using the job manager

When you create an application server cluster from a job manager, you can either create an empty cluster and run subsequent jobs to add cluster members, or you can convert an existing application server as your first cluster member. If you want to create a cluster with one or more new application servers in one step, use the administrative console instead of the job manager.

Building a cluster using the job manager is done in two major steps:

1. Create the cluster.
2. Create the cluster members.

Creating the cluster

To create an application server cluster from the job manager:

1. Click **Jobs** → **Submit**.
2. Click the **Create cluster** job type.
3. Select the **deployment manager** as the job target.

Enter the user ID and password with administrative authority on the deployment manager.

4. Specify the job parameters, as shown in Figure 7-52:
 - Specify the name of the new cluster.

Step 1: Choose a job type

Step 2: Choose job targets

→ Step 3: Specify job parameters

Step 4: Schedule the job

Step 5: Review the summary and submit the job

Specify job parameters

Job type: Create cluster

* Cluster name
Cluster_85_3

Prefer local

Cluster type
Application Server

Short name

Additional job parameters

Replication domain

Create domain

Convert server

Server node

Server name

Member weight

Node group

Replication entry

Previous Next Cancel

Figure 7-52 Specify the options for the new server

Optionally:

- Prefer local: Selected, which is true (the default), or clear the check for false
- Cluster type: The options are:
 - APPLICATION_SERVER (the default)
 - PROXY_SERVER
 - ONDEMAND_ROUTER
- Leave this field blank to create an application server cluster.
- Short name: Cluster short name on z/OS platforms.
- Create domain: For creating true or false (the default).
- Convert server settings: If you want to use an existing server as the first member of the cluster, complete the server node and server name fields. The other fields are optional.

If you specify the cluster name, and take all the other defaults, you create an empty cluster. When you create an empty cluster, it does not appear in the deployment manager console until you submit a job to add a member to it.

5. Schedule the job. Take the defaults for the job schedule. The defaults execute the job once. Click **Next**.
6. Review the summary, and click **Finish**. Monitor the status of the job and ensure it completes successfully.

Creating the cluster members

To create new application server cluster members from the job manager:

1. Click **Jobs** → **Submit**.
2. Click the **Create cluster member** job type.
3. Select the **deployment manager** as the job target.

Enter the user ID and password with administrative authority on the deployment manager.
4. Specify the job parameters, as shown in Figure 7-53 on page 301:
 - a. Specify the name of the cluster.
 - b. Specify the node where the cluster member will be created.
 - c. Specify the name for the new cluster member.

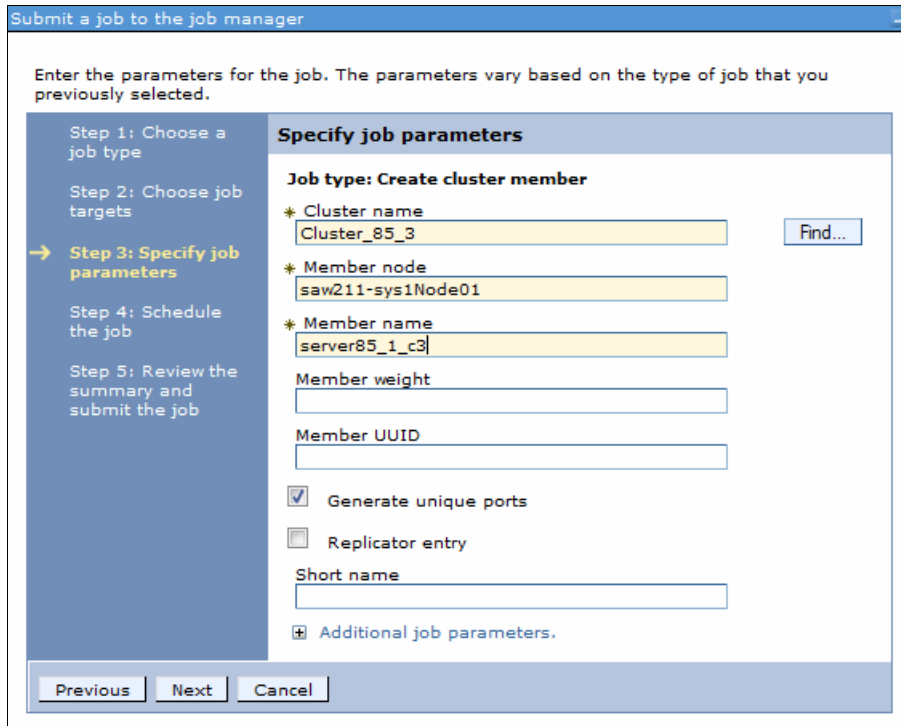


Figure 7-53 Specify the options for the new server

Optionally:

- Member weight: Specify the relative weight of this server in the cluster. Values are from 0 to 20. A 0 indicates that work is to be routed to this server only in the event that no other servers are available.
- Member UUID.
- Generate unique ports: The default is selected for this option, which generates unique ports.
- Replicator entry: Selecting this option adds a replicator entry for this server in the cluster replication domain. The default is not selected, which is a false value.
- Short name: The short name for the server on z/OS.

If this is the first server in the cluster, or if you want to specify a different node or core group, expand the additional job parameters section. It contains settings to specify the template information and the option to specify a node and core group other than the default.

5. Schedule the job. Use the defaults for the job schedule. The defaults execute the job once. Click **Next**.
6. Review the summary, and click **Finish**. Monitor the status of the job and ensure it completes successfully.

7.6.2 Viewing the cluster topology

The deployment manager administrative console provides a graphical view of the existing clusters and their members. To see the view, complete the following steps:

1. Click **Servers** → **Clusters** → **Cluster Topology**.
2. Expand each category, as shown in Figure 7-54 on page 302.

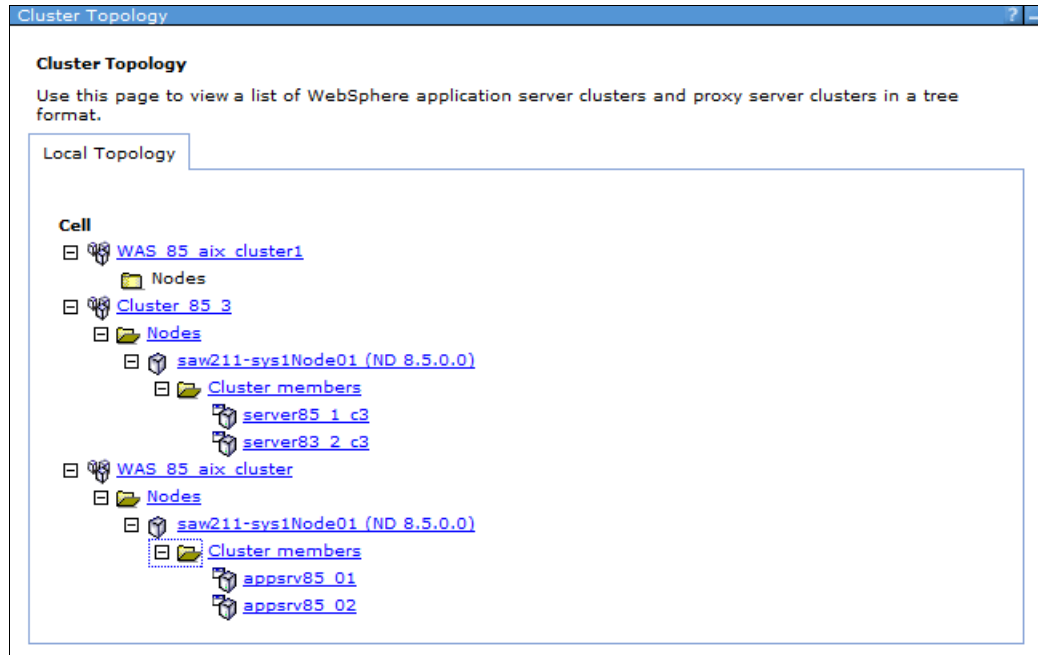


Figure 7-54 Cluster topology view

3. Select a server to get to the configuration window for the application server.

7.6.3 Managing clusters

Application servers within a cluster can be managed as independent servers. A second option is to manage all of the servers in the cluster as a single entity.

Using the administrative console

To display and manage the application server clusters:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Select each cluster you want to work with, and select one of the following options:
 - Start: Use this option to start all servers in the cluster.
 - Stop: Use this option to stop all servers in the cluster. This option allows the server to finish existing requests and allows failover to another member of the cluster.
 - Ripplestart: Use this option to stop and then start all servers in the cluster one at a time.
 - ImmediateStop: Stop all servers immediately.
 - Delete: Deletes the cluster and all servers in the cluster.

Using the job manager

The job manager provides several job types that help you manage clusters:

- ▶ Delete cluster
- ▶ Delete cluster member
- ▶ Start cluster: Use this option to start all servers in a cluster:
 - You can specify that a ripplestart be used (specify true or false). The default is that a ripplestart is not used. A ripplestart stops and then restart each server.

- You can also specify a timeout value. If the timeout expires and all servers are not started, the state of the cluster is reported without waiting any longer for the servers to start.
- ▶ Stop cluster

You can also specify a timeout value. If the timeout expires and all servers are not started, the state of the cluster is reported without waiting any longer for the servers to start.

7.7 Working with virtual hosts

Note: For many users, creating virtual hosts is unnecessary because the `default_host` that is provided is sufficient.

For an example of defining and using a new virtual host, see 23.7, “Deploying the application” on page 855.

A *virtual host* is a configuration enabling a single host machine to resemble multiple host machines. It consists of a host alias or aliases, which consist of a host name and a port number. If you specify an asterisk (*) as a host name, all host names and IP addresses that the web server can receive are mapped to that virtual host.

The following virtual hosts are defined during installation:

- ▶ The *default_host* virtual host is intended for access to user applications, either through the HTTP transport or through a web server.

Host aliases in this virtual host generally include the ports required to access applications from the web server and directly to the application server. Examples are the `wc_defaulthost`, `wc_defaulthost_secure`, `sip_defaulthost`, and `sip_defaulthost_secure` ports for application servers and ports 80 and 443 for requests through the web server.
- ▶ The *admin_host* virtual host is used for access to the WebSphere administrative console.

At installation time, the host is configured to match requests on the `wc_adminhost` and `wc_adminhost_secure` ports for the stand-alone server or deployment manager.
- ▶ The *proxy_host* virtual host includes default port definitions, port 80 and 443, which are typically initialized as part of the proxy server initialization. Use this proxy host as appropriate with routing rules associated with the proxy server.

When you install an application, you associate a virtual host with each web module in the application. By associating a virtual host with a web module, requests that match the host aliases for the virtual host must be processed by servlets in this web module. The web server plug-in also checks the URI of the request against the URIs for the web module to determine whether the web module can handle them or not. You can view or modify the virtual host to which a web module is assigned by clicking **Applications** → **Application Types** → **WebSphere enterprise applications** → *app_name* → **[Web Module Properties] Virtual hosts**.

A single virtual host can be associated with multiple web modules as long as each application has unique URIs. If there are duplicate URIs among applications, different virtual hosts must be created and associated with each of the applications.

A default virtual host is associated with a web container when you create the application server. To find the default virtual host, click **Servers** → **Server Types** → **WebSphere**

application servers, and click the server name to open the configuration page. In the Container settings section, expand **Web Container Settings**, and click **Web container**.

7.8 Creating and updating virtual hosts

By default, default_host is associated with all user application requests. The following examples show cases in which multiple virtual hosts must be created:

- ▶ Applications with conflicting URIs
- ▶ Special support for extra ports
- ▶ Providing independence of each virtual host for applications and servers

To create a new virtual host:

1. Click **Environment** → **Virtual hosts**, and then click **New**.
2. Enter a name for the virtual host, and click **Apply**. Note that two links become active: Host Aliases and MIME Types.
3. Click **Host Aliases** in the Additional Properties pane.
4. Click **New**.
5. Enter values for the Host Name and Port fields, and click **OK**.

The host aliases are not necessarily the same as the host name and port number of the WebSphere Application Server servers. They are the host names and port numbers that the web server plug-in is expecting to receive from the browser. The web server plug-in sends the request to the application server using the host name and port number in the transport setting for that server. If the web server is running on a separate machine from WebSphere, the host aliases are for web server machines.

Mapping HTTP requests to host aliases is case sensitive and the match must be alphabetically exact. Also, different port numbers are treated as different aliases. For example, the request `http://www.myhost.com/myservlet` does *not* map to any of the following sites:

- `http://myhost/myservlet`
- `http://www.myhost.com/MyServlet`
- `http://www.myhost.com:9876/myservlet`

If the web server plug-in receives a request that does not match one of the virtual hosts, it passes the request to the web server. The web server looks in the `web_server_root/htdocs` directory for the content. If it finds the content, it serves the page to the client. If it does not find the content, an HTTP 404 response is returned to the client.

Simple wild cards can be used on the host aliases. A `*` can be used for the host name, the port, or both. It means that any request will match this rule.

Note: If the virtual host is used in a cluster environment, all host aliases used by servers in the cluster must be registered in the virtual host.

6. Save your changes.

Host aliases can also be updated for virtual hosts through the administrative console. To update, complete the following steps:

1. Click **Environment** → **Virtual hosts**.
2. Click the virtual host name to open the configuration page.
3. Click **Host Aliases** in the Additional Properties pane.

4. Click **New**.
5. Enter values for the Host Name and Port fields and click **OK**.

Important: If you create, delete, or update virtual hosts, you need to regenerate the web server plug-in.

One situation that requires creating a new virtual host is the need to secure communication between an unmanaged HTTP server and the application server. This communication can be secured so that it accessed only one specific application from all the applications hosted by that server. To accomplish this, follow these general steps:

1. Add a new secure port to the HTTP server configuration (other than 443, which is the default HTTP server secure port).
2. Generate and configure new certificates for the secure port of the HTTP server.
3. Add a new web container transport chain with a new secure port to the application server.
4. Generate and configure new certificates for the secure port of the application server.
5. Create a new virtual host.
6. Add the new port of the application server and the new port of the HTTP server to the new virtual host.
7. Restart the application server.
8. Re-generate the plug-in for the application server.
9. Manually update the plug-in and configuration of the HTTP server, and map the application on the new virtual host.
10. Update the HTTP server plug-in key stores with the certificate for the new secure port of the application server.
11. Restart the HTTP server.

7.9 Managing applications

WebSphere Application Server V8.5 supports J2EE 1.3, J2EE 1.4, Java EE 5, and Java EE 6, which we refer to as *enterprise applications*. WebSphere Application Server V8.5 can run the following types of applications:

- ▶ Java EE applications
- ▶ Portlet applications
- ▶ Session Initiation Protocol applications
- ▶ Business-level applications
- ▶ OSGi applications (New in Version 8)

OSGi applications are built on an architecture for developing and deploying modular applications and libraries. OSGi applications are built using the OSGi API and deploying it into an OSGi container. WebSphere Application Server provides an OSGi container as part of its basic architecture. For more information about OSGi applications, see Chapter 26, “Working with OSGi applications” on page 921.

For more information about the ways to install enterprise applications and modules, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/crun_app_install.html

7.9.1 Managing enterprise applications: Administrative console

To view and manage applications using the administrative console, click **Applications** → **Application Types** → **WebSphere enterprise Applications**.

In the window, you see the list of installed applications and options for performing application management tasks. Select one or more applications by selecting the box to the left of the application name and then click an action to perform. The exception to this action is the Install option, which installs a new application and requires no existing application to be selected, as shown in Figure 7-55.

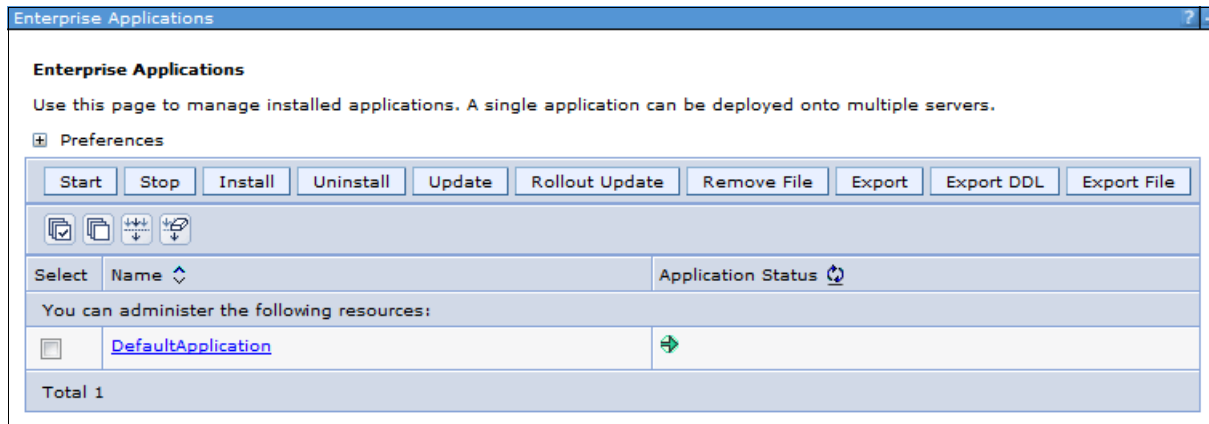


Figure 7-55 Working with enterprise applications

The following list describes the actions you can choose on this window after selecting an application:

- ▶ **Start:** Applications normally start when the server to which they are mapped starts. Exceptions to this situation include when the application is initially installed and when the application is stopped manually. To start an application, the application server that contains the application must be started. If not, the application displays in the administrative console as unavailable and you cannot start it.

An application can also be started from the job manager console using the **Start application** job type.

Note: Starting an application server starts the applications that are mapped to that server. The order in which the applications start depends on the startup order you assigned to each of them. The application with the lowest startup order value is started first. Applications that have the same order designation are started in no particular order. Enabling the parallel start option for the application server means to start applications with the same weight in parallel.

To view or change the application startup order, click **Applications** → **Application Types** → **WebSphere enterprise applications**. Open the configuration window for the application by clicking the application name and then click **Startup behavior**.

- ▶ **Stop:** You can stop an application manually without affecting the rest of the application server processes. This situation is common when you update an application or want to make it unavailable to users. You can also use the job manager console to stop an application by using the **Stop application** job type.

- ▶ **Install:** The install option takes you through the process of installing a new enterprise application EAR file. Deploying an application, using the administrative console and the job manager, is covered in Chapter 23, “Packaging and deploying Java EE applications” on page 813.
- ▶ **Uninstall:** Use this option to uninstall an application. This action removes it from the application servers and from the configuration repository. This action can be performed from the job manager using the **Uninstall application** job type.

Note: When you uninstall an application and it is the only composite unit in a business-level application, the BLA is also deleted.

- ▶ **Update or Rollout Update:** Applications can be updated in several ways. The update options include full application, single module, single file, and partial application.
- ▶ **Remove file:** With this option, you can remove a single file from an application.
- ▶ **Export:** Use this option to export an EAR file of the application.
- ▶ **Export DDL:** Use this option to export DDL files found in the application.
- ▶ **Export File:** Use this option to export individual files found in the application.

7.9.2 Preventing an enterprise application from starting on a server

By default, an application starts when the server starts. The only way to prevent this occurrence is to disable the application from running on the server.

To prevent an enterprise application from starting on a server:

1. From the administrative console, click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name to open the configuration.
3. In the Detail Properties section, select **Target specific application status**.
4. Select the server for which you want to disable the application.
5. Click **Disable Auto Start**.
6. Save the configuration.

7.9.3 Viewing application details

The administrative console does not display the deployed servlets, JSPs, or EJBs directly on the administrative console. However, you can use the administrative console to display XML deployment descriptors for the enterprise application, web modules, and EJB modules.

To view the application deployment descriptor for an application:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name in which you are interested.
3. In the Detail Properties section, click the **Configuration** tab → **View Deployment Descriptor**.

Figure 7-56 on page 308 shows the deployment descriptor window for the DefaultApplication enterprise application. The Configuration tab shows you the structure defined by the deployment descriptor:

- ▶ The name of the enterprise application

- ▶ The web modules and their context roots
- ▶ The EJB modules and their associated JAR files
- ▶ The security roles associated with the enterprise application

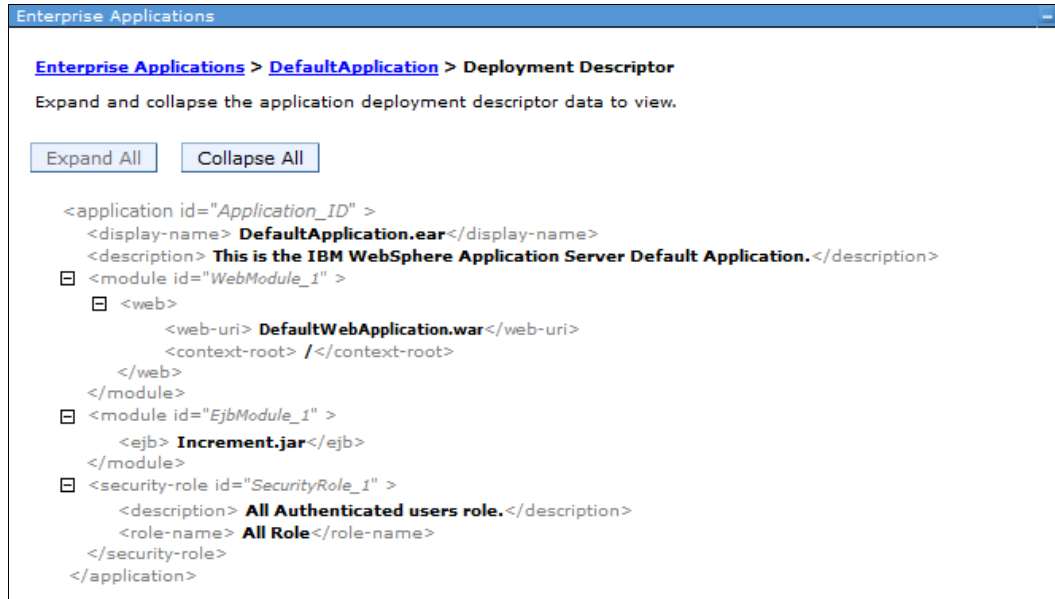


Figure 7-56 Enterprise application deployment descriptor

Viewing EJB modules

To see the EJBs that are part of an enterprise application:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name in which you are interested.
3. Click **Manage Modules** under the Modules Items section.
4. Click the EJB module name that you want to view (Figure 7-57).

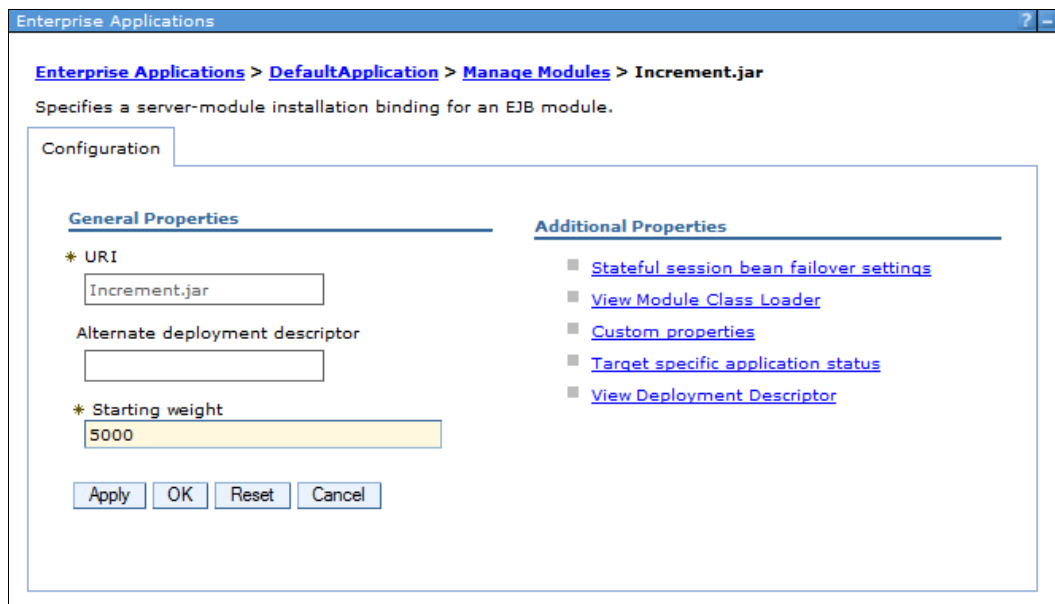


Figure 7-57 Viewing an EJB module configuration

5. Under Additional Properties, click **View Deployment Descriptor** to see the EJB deployment descriptor.

Viewing web modules

To see the servlets and JSPs that are part of an enterprise application:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application name in which you are interested.
3. Under the Modules Items section, click **Manage Modules**.
4. Click the web module name you want to view (Figure 7-58).

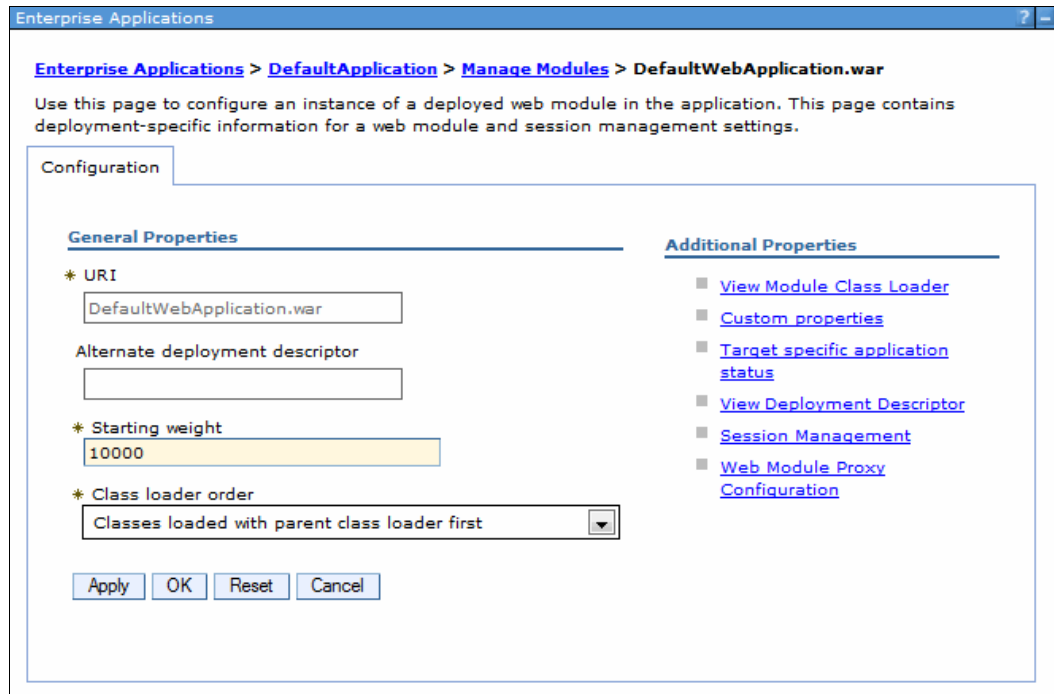


Figure 7-58 View a web module

5. Click **View Deployment Descriptor** to see the details of the web module content.

7.9.4 Finding a URL for a servlet or JSP

The URL for a servlet or JSP is the path used to access it from a browser. The URL is partly defined in the deployment descriptor provided in the EAR file and partly defined in the deployment descriptor for the web module containing the servlet or JSP.

To find the URL for a servlet or JSP:

1. Find the context root of the web module containing the servlet.
2. Find the URL for the servlet.
3. Find the virtual host where the web module is installed.
4. Find the server or cluster on which the application is installed.
5. Find the aliases by which the virtual host is known.
6. Combine the virtual host alias, context root, and URL pattern to form the URL request of the servlet/JSP.

For example, to look up the URL for the snoop servlet:

1. Find the context root of the web module DefaultWebApplication of the DefaultApplication enterprise application. This web module contains the snoop servlet:
 - a. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
 - b. Click the application in which you are interested, which in our case is **DefaultApplication.ear**.
 - c. On the Configuration tab, click **Context Root for Web Modules** (Figure 7-59) in the Web Module Properties section. Notice the following items:
 - There is only one web module in this application, that is, the Default Web Application.
 - The context root is “/”. We will use this later.

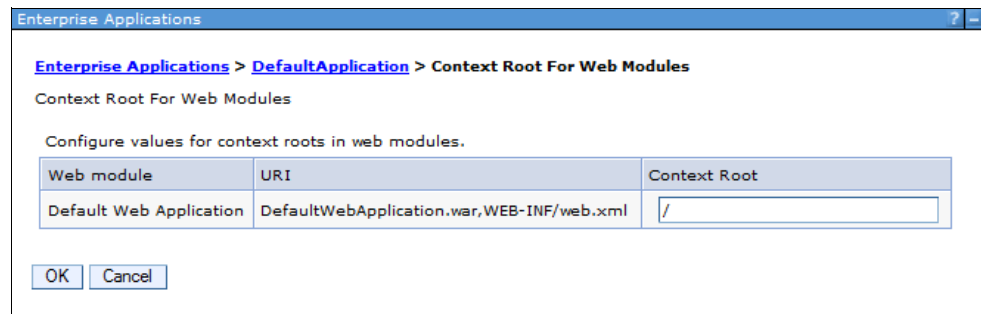


Figure 7-59 Context root for the web modules in DefaultApplication

- d. Click **OK** to return to the DefaultApplication configuration.
2. Find the URL for the snoop servlet:
 - a. Locate the DefaultApplication configuration window, and in the Modules section, select **Manage Modules**.
 - b. Click the **Default Web Application** web module to see the general properties.
 - c. Under Additional Properties, click **View Deployment Descriptor** to display the web module properties window, as shown in Figure 7-60 on page 311. Note that the URL

pattern for the snoop servlet starting from the web module context root is “/snoop/*”. The web module context root is “/”.

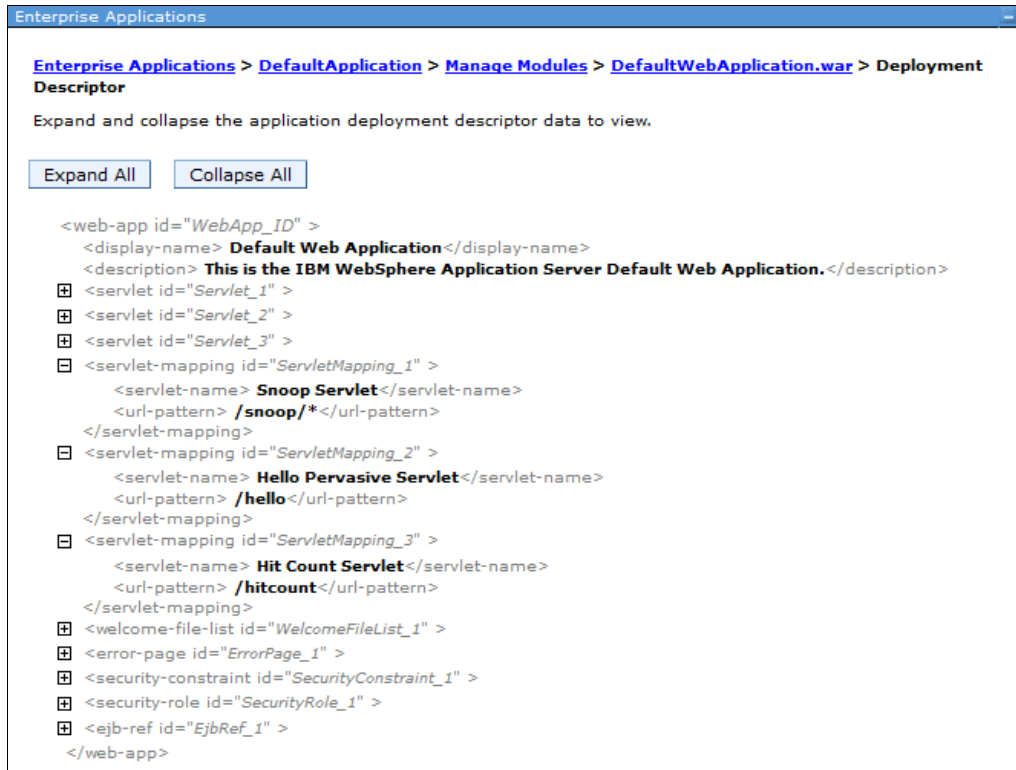


Figure 7-60 DefaultWebApplication web module deployment descriptor

- d. As you navigate through the windows, a navigation path is displayed underneath the Messages area, which is the bread-crum trail. Click **DefaultApplication.ear** to return to the application configuration page.
3. Find the virtual host where the web module is installed:

In the DefaultApplication configuration window, click **Virtual hosts** under the Web Module Properties section. This action displays all of the web modules contained in the enterprise application and the virtual hosts in which they were installed, as shown in Figure 7-61 on page 312. Note that the Default Web Application Web module is installed on the default_host virtual host.

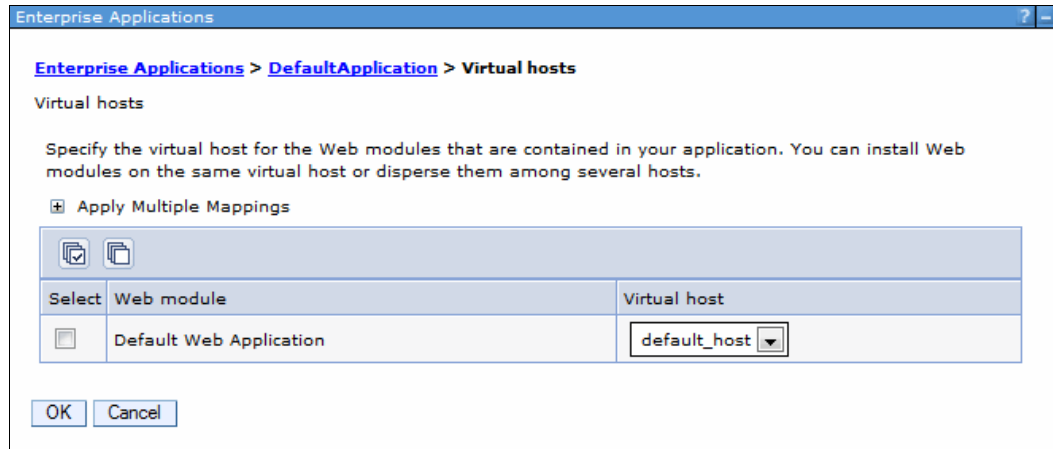


Figure 7-61 Virtual hosts

4. Find the server or cluster on which the application is installed:
 - a. In the DefaultApplication configuration window, click **Manage Modules** under the Modules section.
 - b. Click the Default Web Application module name.
 - c. Under the Additional Properties section, click **Target specific application status**.
 - d. Click the server name or cluster name listed as the target.
 - e. Look for the WC_defaulthost and WC_defaulthost_secure port values for the server or for each server in case of a cluster configuration.
5. Find the host aliases for the default_host virtual host:
 - a. From the administration console navigation tree, click **Environment** → **Virtual Hosts**.
 - b. Click **default_host**.
 - c. Under Additional Properties, click **Host Aliases**. This action shows the list of aliases by which the default_host virtual host is known, as shown in Figure 7-62 on page 313.

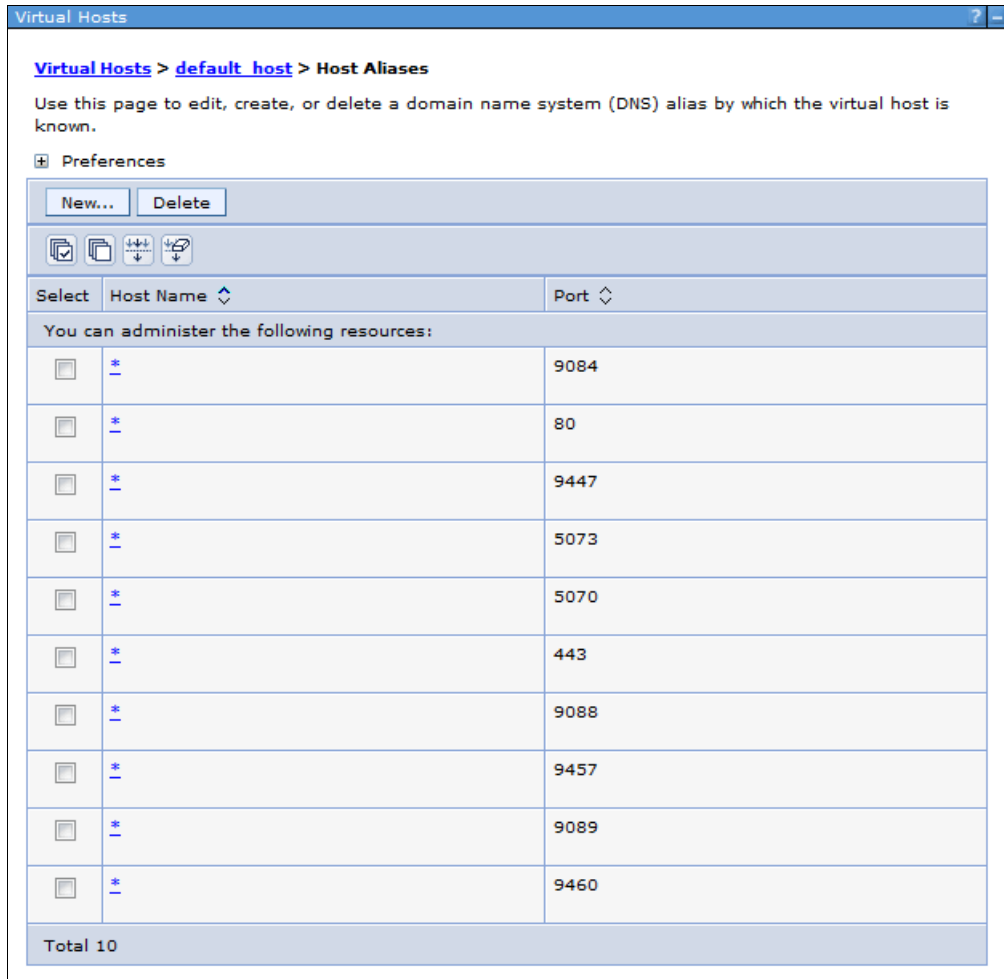


Figure 7-62 Default_host virtual host aliases

Note that the aliases are composed of a DNS host name and a port number. Combine aliases ports and the WC_defaulthost and WC_defaulthost_secure ports of the server. In the case of a cluster, you combine the ports of several servers. After ports are combined, an accurate list of aliases, based from where the applications are mapped to, can be generated. In our case, the host aliases for the default_host virtual host are *:80, *:443, *:9084, and *:9447, where “*” stands for any host name. Ports 80 and 443 are used for HTTP server (non-secure and secure port).

6. Combine the virtual host alias, context root, and URL pattern to form the URL request of the snoop servlet. Requests for the servlet with any of the following URLs map to the default_host virtual host:
 - http://hostname:80/snoop
 - https://hostname:443/snoop
 - http://hostname:9084/snoop
 - https://hostname:9447/snoop

7.10 Enabling process restart on failure

In a distributed environment, you can use the health management feature to monitor the status of application servers, nodes, clusters, dynamic clusters, on demand routers, and cells

so that you can sense and respond to problem areas before an outage occurs. You can manage the health of an application serving environment with a policy-driven approach that enables specific actions to occur when monitored criteria is met. For example, for an application server, when memory usage exceeds a percentage of the heap size for a specified time, health policy actions can run to correct the situation. The following list shows some of the predefined health policy actions that are applicable to excessive memory usage:

- ▶ Take thread dumps
- ▶ Take JVM heap dumps
- ▶ Generate a SNMP trap
- ▶ Place server in maintenance mode
- ▶ Place server in maintenance mode and break affinity
- ▶ Place server out of maintenance mode
- ▶ Restart server

All of the listed actions can be grouped and used in a custom sequence to help you detect and correct the problem. You can use the administrative console to set health policies by clicking **Operational policies** → **Health policies**. Figure 7-63 describes a sequence of actions that you might set in case your server exceeds 90 percent of the JVM heap size for a period of two minutes.

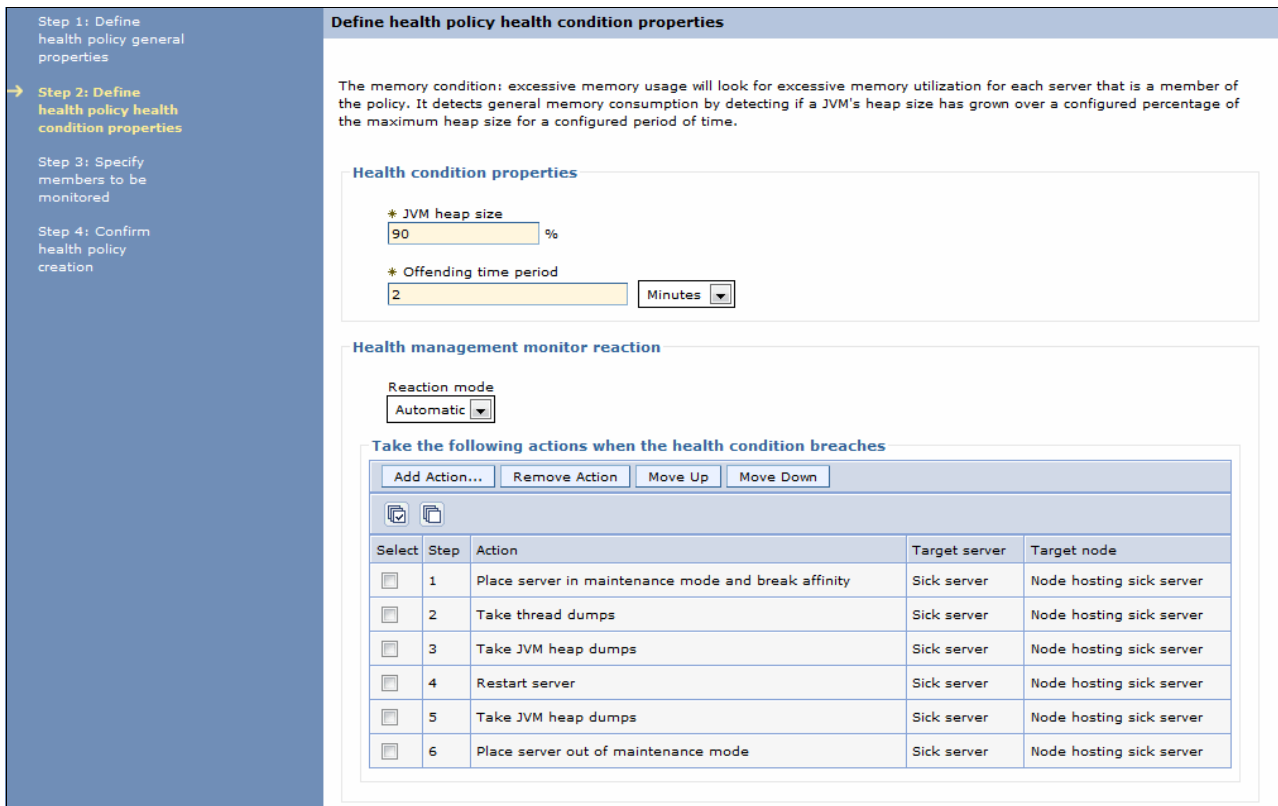


Figure 7-63 Health policy condition actions for excessive memory usage

The two reaction modes for the health management monitor are:

- ▶ Supervise: When the health condition is reached, a task is submitted with a suggested plan of action that is automatically carried out if the task is approved.
- ▶ Automatic: When the health condition is reached, the actions are automatically carried out in the order you previously defined.

You can define a large number of custom health conditions and actions for when the health conditions breach. Operational policies are described in Chapter 13, “Intelligent management” on page 469. Intelligent management features help you recover from the most common operational issues, and there is a more general way to restart your server processes. You can use the native operating system functionality to restart a failed process. The following sections provide more information about how to set your operating system.

7.10.1 Windows

The administrator can choose to register one or more of the WebSphere Application Server processes on a machine as a Windows service during profile creation. It can also be done after profile creation using the **WASService** command. With this command, Windows automatically attempts to restart the service if it fails during use.

Syntax

Enter `WASService.exe` with no arguments to get a list of the valid formats, as shown in Example 7-20.

Example 7-20 WASService command format

```
Usage: WASService.exe (with no arguments displays this help)
|| -add <service name>
   -serverName <Server>
   -profilePath <Server's Profile Directory>
       [-wasHome <WebSphere Install Directory>]
       [-configRoot <Config Repository Directory>]
       [-startArgs <additional start arguments>]
       [-stopArgs <additional stop arguments>]
       [-userid <execution id> -password <password>]
       [-logFile <service log file>]
       [-logRoot <server's log directory>]
       [-encodeParams]
       [-restart <true | false>]
       [-startType <automatic | manual | disabled>]
|| -remove <service name>
|| -start <service name> [optional startServer.bat parameters]
|| -stop <service name> [optional stopServer.bat parameters]
|| -status <service name>
|| -encodeParams <service name>
```

Be aware of the following considerations when using the **WASService** command:

- ▶ When adding a new service, the **-serverName** argument is mandatory. The `serverName` is the process name. If in doubt, use the **serverstatus -all** command to display the processes. For a deployment manager, the `serverName` is `dmgr`. For a node agent, the `serverName` is `nodeagent`, and for a server, it is the server name.
- ▶ The **-profilePath** argument is mandatory. It specifies the home directory for the profile.
- ▶ Use unique service names. The services are listed in the Windows Services control window as:

```
IBM WebSphere Application Server V8.0 - <service name>
```

The convention used by the Profile Management Tool is to use the node name as the service name for a node agent. For a deployment manager, it uses the node name of the deployment manager node concatenated with `dmgr` as the service name.

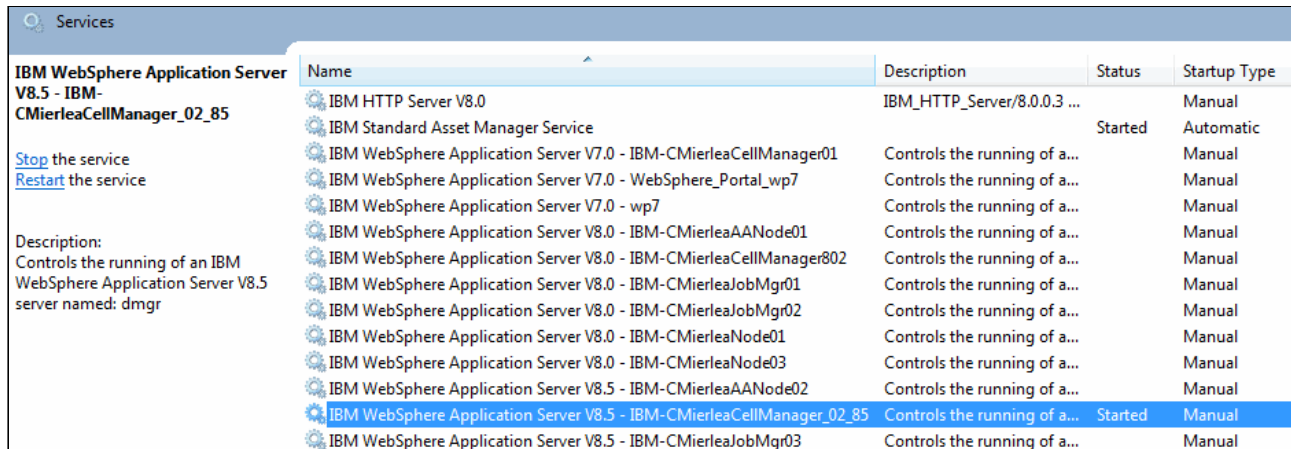
Examples

Example 7-21 shows how to use the **WASService** command to add a node agent as a Windows service.

Example 7-21 Registering a deployment manager as a Windows 7 service

```
D:\was85\IBM\WebSphere\AppServer\bin>runas /user:IBM-CMierlea\admin
"D:\was85\IBM\WebSphere\AppServer\bin\WASService -add "dmgr" -servername dmgr -profilePath
"D:\was85\IBM\WebSphere\AppServer_85_01" -restart true"
Enter the password for IBM-CMierlea\admin:
Attempting to start D:\was85\IBM\WebSphere\AppServer\bin\WASService -add dmgr -servername
dmgr -profilePath D:\was85\IBM\WebSphere\AppServer\profiles\Dmgr_85_01 -restart true as
user "IBM-CM
..
D:\was85\IBM\WebSphere\AppServer\bin>
```

Note that the service name added in Figure 7-64 will be IBM WebSphere Application Server V8.5, concatenated with the name you specified for the service name. You can set recovery actions in case of failure using the Recovery tab under the Properties of the new service.



| Name | Description | Status | Startup Type |
|---|------------------------------|---------|--------------|
| IBM HTTP Server V8.0 | IBM_HTTP_Server/8.0.0.3 ... | | Manual |
| IBM Standard Asset Manager Service | | Started | Automatic |
| IBM WebSphere Application Server V7.0 - IBM-CMierleaCellManager01 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V7.0 - WebSphere_Portal_wp7 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V7.0 - wp7 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V8.0 - IBM-CMierleaAANode01 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V8.0 - IBM-CMierleaCellManager802 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V8.0 - IBM-CMierleaJobMgr01 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V8.0 - IBM-CMierleaJobMgr02 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V8.0 - IBM-CMierleaNode01 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V8.0 - IBM-CMierleaNode03 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V8.5 - IBM-CMierleaAANode02 | Controls the running of a... | | Manual |
| IBM WebSphere Application Server V8.5 - IBM-CMierleaCellManager_02_85 | Controls the running of a... | Started | Manual |
| IBM WebSphere Application Server V8.5 - IBM-CMierleaJobMgr03 | Controls the running of a... | | Manual |

Figure 7-64 New services

If you remove the service using the **WASService -remove** command, specify only the latter portion of the name, as shown in Example 7-22.

Example 7-22 Removing a service

```
D:\was85\IBM\WebSphere\AppServer\bin>runas /user:IBM-CMierlea\admin
"D:\was85\IBM\WebSphere\AppServer\bin\WASService -remove "dmgr""
Enter the password for IBM-CMierlea\admin:
Attempting to start D:\was85\IBM\WebSphere\AppServer\bin\WASService -remove dmgr
as user "IBM-CMierlea\admin" ...
D:\was85\IBM\WebSphere\AppServer\bin>
```

7.10.2 UNIX and Linux

The administrator can choose to include entries in *inittab* for one or more of the WebSphere Application Server processes on a machine, as shown in Example 7-23 on page 317. Each such process is then automatically restarted if it has failed.

Example 7-23 Inittab contents for process restart

On deployment manager machine:

```
ws1:23:respawn:/usr/WebSphere/DeploymentManager/bin/startManager.sh
```

On node machine:

```
ws1:23:respawn:/usr/WebSphere/AppServer/bin/startNode.sh
ws2:23:respawn:/usr/WebSphere/AppServer/bin/startServer.sh nodename_server1
ws3:23:respawn:/usr/WebSphere/AppServer/bin/startServer.sh nodename_server2
ws4:23:respawn:/usr/WebSphere/AppServer/bin/startServer.sh nodename_server2
```

Note: When setting the action for `startServer.sh` to respawn in `/etc/inittab`, be aware that `init` always restarts the process, even if you intended for it to remain stopped. As an alternative, you can use the `rc.was` script located in `${WAS_HOME}/bin`, which allows you to limit the number of retries.

The best solution is to use a monitoring product that implements notification of outages and logic for automatic restart.

7.10.3 z/OS

WebSphere for z/OS takes advantage of the z/OS Automatic Restart Management (ARM) to recover application servers. Each application server running on a z/OS system (including servers you create for your business applications) are automatically registered with an ARM group. Each registration uses a special element type called SYSCB. ARM treats SYSCB as restart level 3, ensuring that RRS (a z/OS facility that provides two-phase sync point support across participating resource managers) restarts before any application server.

Note: If you have an application that is critical for your business, you need facilities to manage failures. z/OS provides rich automation interfaces, such as automatic restart management, which you can use to detect and recover from failures. The automatic restart management handles the restarting of servers when failures occur.

Some important things to consider when using automatic restart management:

- ▶ If you have automatic restart management (ARM) enabled on your system, you might want to disable ARM for the WebSphere Application Server for z/OS address spaces before you install and customize WebSphere Application Server for z/OS. During customization, job errors might cause unnecessary restarts of the WebSphere Application Server for z/OS address spaces. After installation and customization, consider enabling ARM.
- ▶ If you are ARM-enabled and you cancel or stop a server, it will restart in place using the `armrestart` command.
- ▶ It is a good idea to set up an ARM policy for your deployment manager and node agents. For more information about how to change the ARM policies, refer to the following information center website:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.zseries.doc/ae/cins_changearm.html
- ▶ If you start the location service daemon on a system that already has one, it will terminate.
- ▶ Every other server comes up on a dynamic port unless the configuration has a fixed port. Therefore, the fixed ports must be unique in a sysplex.
- ▶ If you issue STOP, CANCEL, or MODIFY commands against server instances, be aware of how automatic restart management behaves regarding WebSphere Application Server

for z/OS server instances. Table 7-1 depicts ARM behavior regarding WebSphere Application Server for z/OS server instances.

Table 7-1 ARM Behavior and WebSphere Application Server for z/OS server instances

| When you issue | ARM behavior |
|--|--|
| STOP <i>address_space</i> | It does not restart the address space. |
| CANCEL <i>address_space</i> | It does not restart the address space. |
| CANCEL <i>address_space</i> , ARMRESTART | It does restart the address space. |
| MODIFY <i>address_space</i> , CANCEL | It does not restart the address space. |
| MODIFY <i>address_space</i> , CANCEL, ARMRESTART | It restarts the address space. |

For more information about how to activate the ARM, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.zseries.doc/ae/tins_activearm.html

If you activated ARM and want to check the status of address spaces registered for automatic restart management:

1. Initialize all servers.
2. Issue one or both of the commands shown in Example 7-24.

Example 7-24 Displaying the status of address spaces registered for ARM

To display all registered address spaces (including the address spaces of server instances), issue the command:

```
d xcf,armstatus,detail
```

To display the status of a particular server instance, use the display command and identify the job name. For example, to display the status of the Daemon server instance (job BBODMN), issue the following command:

```
d xcf,armstatus,jobname=bbodmn,detail
```

For more information about how to use the **display** command, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_mvdisplay.html



Administration with scripting

The administrative console addresses tasks that are non-repetitive, have a minimal number of administrative steps, and are relatively simple. For administration that requires many steps, that can be repetitive, and time consuming to configure, `wsadmin` combined with scripts is an ideal tool.

In this chapter, we introduce the `wsadmin` scripting solution and describe how you can use it to perform basic tasks.

This chapter contains the following topics:

- ▶ Overview of WebSphere scripting
- ▶ Launching `wsadmin`
- ▶ Command and script invocation
- ▶ The `wsadmin` tool management objects
- ▶ Properties file based configuration
- ▶ Managing WebSphere using script libraries
- ▶ Assistance with scripting
- ▶ Example: Using scripts with the job manager
- ▶ Online resources

8.1 Overview of WebSphere scripting

WebSphere Application Server provides a scripting interface based on the *Bean Scripting Framework (BSF)*. This interface is called **wsadmin**. BSF is an open source project that is used to implement an architecture for incorporating scripting into Java applications and applets. The BSF architecture works as an interface between Java applications and scripting languages. Using this framework allows scripting languages to complete the following tasks:

- ▶ Look up a preregistered bean and access a predeclared bean.
- ▶ Register a newly created bean.
- ▶ Perform all bean operations.
- ▶ Bind events to scripts in the scripting language.

Because **wsadmin** uses BSF, it can make various Java objects available through language-specific interfaces to scripts. Figure 8-1 shows the major components that are involved in the **wsadmin** scripting solution.

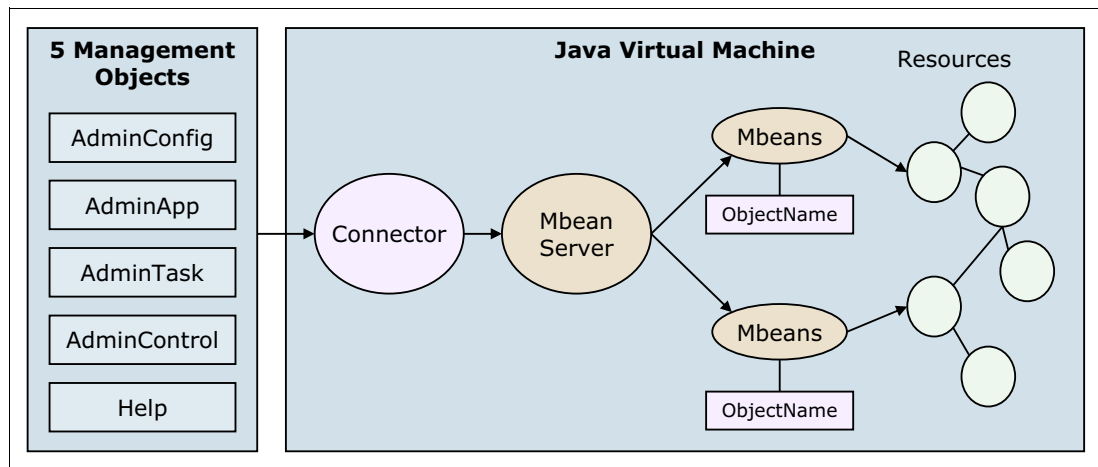


Figure 8-1 Scripting in wsadmin

You can use the following programming languages to write wsadmin scripts:

- ▶ Jython
- ▶ Jacl

With WebSphere Application Server V7, the deprecation process for the Jacl syntax began. The script library and the command assistance on the administrative console support only Jython.

If you have existing Jacl scripts and want to start migrating to Jython, you can use the Jacl-to-Jython conversion utility to convert the scripts. This conversion assistant typically achieves 95-98% of a preliminary conversion. In most cases, the resulting conversion is syntactically and runtime equivalent.

However, verify each line to ensure that the code functions as you originally intended. When Jacl and Jython language differences result in lines of code that are difficult to convert automatically, the converted lines are flagged with a `#?PROBLEM?` tag. This tag provides assistance in finding areas that are most likely in need of manual conversion.

8.2 Launching wsadmin

The **wsadmin** command file resides in the bin directory of every profile. Start **wsadmin** from a command prompt with the following command (as appropriate):

- ▶ UNIX: `profile_root/bin/wsadmin.sh`
- ▶ Windows: `profile_root\bin\wsadmin`

Note that the **wsadmin** command also exists in the bin directory of the `install_root` directory. If you start **wsadmin** from this location, you must be careful to specify the profile to work within the command. If you do not specify the profile, the default profile is chosen.

Example 8-1 shows how to start **wsadmin**. In this example, the **wsadmin** command is used to connect to the job manager. It is issued from the bin directory of the job manager profile. The profile does not need to be specified. The **-lang** argument indicates Jython is used (Jacl is the default).

Example 8-1 Flexible management: The wsadmin command line

```
C:\WebSphereV8.5\AppServer\profiles\jmgr40\bin>wsadmin -lang jython
WASX7209I: Connected to process "jobmgr" on node jmgr40node using SOAP connector
; The type of process is: JobManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>
```

To get syntax-related help, use **wsadmin -?** or **-help** (see Example 8-2).

Example 8-2 The wsadmin syntax

```
wsadmin
[ -h(elp) ]
[ -? ]
[ -c <command> ]
[ -p <properties_file_name>]
[ -profile <profile_script_name>]
[ -f <script_file_name>]
[ -javaoption java_option]
[ -lang language]
[ -wsadmin_classpath class_path]
[ -profileName profile]
[ -conntype
    SOAP
        [-host host_name]
        [-port port_number]
        [-user userid]
        [-password password] |
    RMI
        [-host host_name]
        [-port port_number]
        [-user userid]
        [-password password] |
    JSR160RMI
        [-host host_name]
        [-port port_number]
        [-user userid]
        [-password password] |
    IPC
```

```

        [-ipchost host_name]
        [-port port_number]
        [-user userid]
        [-password password] |
    NONE
]
[ -jobid <jobid_string>]
[ -tracefile <trace_file>]
[ -appendtrace <true/false>]
[ script parameters ]

```

8.2.1 Scripting environment properties file

You can set the properties that determine the scripting environment for **wsadmin** using either the command line or a properties file. Modifying the properties file can be useful when you want to change a default setting, for example, changing the language from Jacl to Jython.

You can set properties in the following locations:

- ▶ The installation default properties file for the profile, which is located in the following directory:
profile_root/properties/wsadmin.properties
- ▶ A user default properties file, which is located in the Java user `.home` property.
- ▶ A customized properties file placed in the location that is pointed to by the `WSADMIN_PROPERTIES` environment variable.
- ▶ A customized properties file, which is pointed to using the `-p` argument to the **wsadmin** command.

When **wsadmin** is started, properties are loaded from these files in the order listed in Table 8-1. The properties file that is loaded last overrides any property files loaded earlier.

Table 8-1 The *wsadmin* properties

| Property | Value |
|--|---|
| <code>com.ibm.ws.scripting.connectionType</code> | SOAP, RMI or NONE |
| <code>com.ibm.scripting.port</code> | TCP port of target system |
| <code>com.ibm.scripting.host</code> | Host name of target system |
| <code>com.ibm.ws.scripting.defaultLang</code> | Jython or Jacl |
| <code>com.ibm.ws.scripting.echoparams</code> | Determines whether parameters or arguments are output to STDOUT or to the wsadmin trace file |
| <code>com.ibm.ws.scripting.traceFile</code> | File for trace information |
| <code>com.ibm.ws.scripting.validationOutput</code> | Location of validation reports |
| <code>com.ibm.ws.scripting.traceString</code> | <code>=com.ibm.*=all=enabled</code> |
| <code>com.ibm.ws.scripting.appendTrace</code> | Appends to the end of the existing log file |
| <code>com.ibm.ws.scripting.profiles</code> | List of profiles to be run before running user commands, scripts, or an interactive shell |
| <code>com.ibm.ws.scripting.emitWarningForCustomSecurityPolicy</code> | Controls whether message WASX7207W is emitted when custom permissions are found |

| Property | Value |
|---|--|
| com.ibm.ws.scripting.tempdir | Stores temporary files when installing applications |
| com.ibm.ws.scripting.validationLevel | Level of validation to use when configuration changes are made from the scripting interface |
| com.ibm.ws.scripting.crossDocumentValidationEnabled | Determines whether the validation mechanism examines other documents when changes are made to one document |
| com.ibm.ws.scripting.classpath | List of paths to search for classes and resources |

Some of the listed properties in the `wsadmin.properties` file are commented out by default. An example is `com.ibm.ws.scripting.traceString`. If you want to trace `wsadmin` execution, remove the comment sign (#) from the properties file.

Some of the properties contain default values, for example, `com.ibm.ws.scripting.connectionType` has a default value of SOAP. Thus, when a scripting process is invoked, a SOAP connector is used to communicate with the server. The `com.ibm.ws.scripting.defaultLang` property is set to Jacl.

Use the `-p` option to specify a customized properties file. Example 8-3 shows sample coding for invoking `wsadmin` to execute a script file using a specific properties file.

Example 8-3 Specifying properties file on the command line

```
C:\WebSphereV8.5\AppServer\profiles\dmgr40\bin>wsadmin -p
C:\WebSphereV8.5\AppServer\profiles\dmgr40\properties\wsadmin_custom.properties
WASX7209I: Connected to process "dmgr" on node dmgr40node using SOAP connector;
The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
```

8.2.2 Script profile file

A *script profile* is a script that is invoked before the main script or before invoking `wsadmin` in interactive mode. The purpose of the script profile is to customize the environment on which the script runs. For example, a script profile can be set for the Jacl scripting language that makes Jacl-specific variables or procedures available to the interactive session or main script.

Use the `-profile` command-line option to specify a profile script. Several `-profile` options can be used on the command line and are invoked in the order given.

8.2.3 Connected versus local mode

The `wsadmin` command can operate in either connected or local mode. In connected mode, all operations are performed by method invocations on running JMX MBeans. In local mode, the application server (MBeans server) is not started and the `wsadmin` objects are limited to configuring the server by means of directly manipulating XML configuration documents.

When operating in local mode, be sure that you are operating on the correct profile, either using the `-profileName` argument or starting `wsadmin` from the `profile/bin` directory.

When performing configuration changes in local mode in a distributed server environment, make configuration changes at the deployment manager level. Changes made directly to the node configuration are lost at server startup or at configuration replication.

Use the `-conntype NONE` option to run in local mode.

8.3 Command and script invocation

The `wsadmin` commands can be invoked in three different ways. This section describes how to invoke commands the following ways:

- ▶ Invoking a single command
- ▶ Running script files
- ▶ Invoking commands interactively

Note: For simplicity, the examples in this chapter assume the following facts:

- ▶ `wsadmin` is executed from the `profile_root/bin` directory. It is not necessary to specify the profile name, host, and port.
- ▶ Administrative security is disabled. In reality, you need to specify the user name and password when you invoke `wsadmin`.

8.3.1 Invoking a single command (-c)

You can use the `-c` option to execute a single command using `wsadmin`, as shown in Example 8-4. In this example, we use the `AdminControl` object to query the node name of the WebSphere server process.

Example 8-4 Running a single command in wsadmin

```
C:\WebSphereV8.5\AppServer\profiles\jmgr40\bin>wsadmin -lang jython -c
AdminControl.getNode()
WASX7209I: Connected to process "jobmgr" on node jmgr40node using SOAP connector
; The type of process is: JobManager
'jmgr40node'

C:\WebSphereV8.5\AppServer\profiles\jmgr40\bin>
```

8.3.2 Running script files (-f)

You can use the `-f` option to execute a script file. Example 8-5 shows a two-line Jython script named `myScript.py`. The script has a `.py` extension to reflect the Jython language syntax of the script. The extension plays no significance in `wsadmin`. The `com.ibm.ws.scripting.defaultLang` property or `-lang` parameter is used to determine the language used. If the property setting is not correct, use the `-lang` option to identify the scripting language because the default is `Jacl`.

Example 8-5 Jython script

```
print "This is an example Jython script"
print ""+ AdminControl.getNode()+"
```

Example 8-6 shows how to execute the script.

Example 8-6 Running a Jython script in wsadmin

```
C:\WebSphereV8.5\AppServer\profiles\dmgr40\bin>wsadmin -f myScript.py -lang jython

WASX7209I: Connected to process "dmgr" on node dmgr40node using SOAP connector;
  The type of process is: DeploymentManager
This is an example Jython script
dmgr40node
```

8.3.3 Invoking commands interactively

You can run the command execution environment using interactive mode, so that you can invoke multiple commands without incurring the impact of starting and stopping the **wsadmin** environment for every single command. Run the **wsadmin** command without the command (-c) or script file (-f) options to start the interactive command execution environment, as shown in Example 8-7.

Example 8-7 Starting the wsadmin interactive command execution environment

```
C:\WebSphereV8.5\AppServer\profiles\dmgr40\bin>wsadmin -lang jython
WASX7209I: Connected to process "dmgr" on node dmgr40node using SOAP connector;
  The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>
```

From the **wsadmin>** prompt, you can invoke the WebSphere administrative objects and built-in language objects, as shown in Example 8-8. Enter the commands at the **wsadmin>** prompt.

Example 8-8 Interactive command invocation

```
wsadmin>AdminControl.getNode()
'dmgr40node'
wsadmin>
```

End the interactive execution environment by typing **quit** and then pressing Enter.

8.4 The wsadmin tool management objects

The **wsadmin** tool has the following administrative objects that provide server configuration and management capabilities:

- ▶ Help
- ▶ AdminControl
- ▶ AdminConfig
- ▶ AdminApp
- ▶ AdminTask

This section provides information about how to use the script libraries and command assist. For information about how to use management objects, refer to the information center at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=txml_launchscript%20File%20name:%20txml_launchscript.html

8.4.1 Help

The Help object provides a quick way to get information about methods, operations, and attributes using scripting. For example, to get a list of the public methods that are available for the AdminControl object, enter the following command:

```
wsadmin>print Help.AdminConfig()
```

To get a detailed description of a specific object method and the parameters that it requires, invoke the help method of the target object with the method name as the option to the help method, as shown in Example 8-9.

Example 8-9 AdminConfig.help scripting

```
wsadmin>print AdminConfig.help("createClusterMember")
```

```
WASX7284I: Method: createClusterMember  
Arguments: cluster id, node id, member attributes
```

```
Description: Creates a new Server object on the node specified  
by "node id." This Server is created as a new member of the existing  
cluster specified by "cluster id," and has attributes specified in  
"member attributes." One attribute is required: "memberName."  
The Server is created using the default template for  
Server objects, and has the name and specified by the  
"memberName" attribute.  
attribute.
```

```
Method: createClusterMember
```

```
Arguments: cluster id, node id, member attributes, template id
```

```
Description: Creates a new Server object on the node specified  
by "node id." This Server is created as a new member of the existing  
cluster specified by "cluster id," and has attributes specified in  
"member attributes." One attribute is required: "memberName."  
The Server is created using the Server template  
specified by "template id," and has the name specified by  
the "memberName" attribute.
```

The AdminTask object also supports searching for the specific command by using a wildcard character under help. For example, Example 8-10 shows the command to get a list of the command names that start with the term *create*.

Example 8-10 AdminTask.help scripting

```
wsadmin>print AdminTask.help("-commands", "create*")
```

```
WASX8004I: Available admin commands:
```

```
createAllActivePolicy - Create a policy that automatically activates all group members.  
createApplicationServer - Command that creates a server
```

createApplicationServerTemplate - creates a server Template based on a server configuration
createAuditEncryptionConfig - Configures audit record encryption.
createAuditEventFactory - Creates an entry in the audit.xml to reference the configuration of a Factory interface.
createAuditFilter - Creates an entry in the audit.xml to reference an Audit Specification. Enable
createAuditKeyStore - Creates a new Key Store.
createAuditNotification - Configures an audit notification.
createAuditNotificationMonitor - Configures an audit notification monitor.
createAuditSelfSignedCertificate - Create a new self-signed certificate and store it in a keys
....

8.4.2 AdminControl

The AdminControl object is used for operational control. It communicates with MBeans that represent live objects running a WebSphere server process. It includes commands to query existing running objects and their attributes and to invoke operations on the objects. In addition to the operational commands, the AdminControl object supports commands to query information about the connected server, to trace clients, to reconnect to a server, and to start and stop a server.

Note that because the AdminControl object operates on live MBeans, you cannot use it to start a deployment manager, node agent, or stand-alone application server.

8.4.3 AdminConfig

The AdminConfig object is used to manage the configuration information that is stored in the repository. This object communicates with the WebSphere Application Server configuration service component to make configuration inquiries and changes. You can use it to query existing configuration objects, create configuration objects, modify existing objects, and remove configuration objects. In a distributed server environment, the **AdminConfig** commands are available only if a scripting client is connected to the deployment manager. When connected to a node agent or a managed application server, the **AdminConfig** commands are not available because the configuration for these server processes are copies of the master configuration that resides in the deployment manager.

8.4.4 AdminApp

The AdminApp object can update application metadata, map virtual hosts to web modules, and map servers to modules for applications already installed. Changes to an application, such as specifying a library for the application to use or setting session management configuration properties, are performed using the AdminConfig object.

8.4.5 AdminTask

The AdminTask object is used to access a set of task-oriented administrative commands that provide an alternative way to access the configuration commands and the running object management commands. The administrative commands run simple and complex commands. The administrative commands are discovered dynamically when the scripting client is started. The set of available administrative commands depends on the edition of WebSphere Application Server that you install. You can use the AdminTask object commands to access these commands.

Two run modes are always available for each administrative command, namely the batch and interactive mode. When you use an administrative command in interactive mode, you go through a series of steps to collect your input interactively. This process provides users a text-based wizard and a similar user experience to the wizard in the administrative console. You can also use the `help` command to obtain help for any of the administrative commands and the AdminTask object.

8.5 Properties file based configuration

Using complex scripts requires knowledge of the Jacl or Jython scripting languages and the public MBean interfaces. The use of a properties file-based configuration provides a way to simplify administrative tasks using `wsadmin`. WebSphere Application Server configuration can be extracted into a single file and any configuration attribute can be located in that file in the form of name/value pair properties.

The following commands in the PropertiesBasedConfiguration command group implement this type of configuration:

- ▶ `extractConfigProperties`
- ▶ `validateConfigProperties`
- ▶ `applyConfigProperties`
- ▶ `deleteConfigProperties`
- ▶ `createPropertiesFileTemplates`

Properties file-based configuration extracts configuration data to one file that is easy to read using any editor tool. The configuration attributes are provided as key/value pairs, as shown in Figure 8-2.

```
#
# SubSection 1.0 # JDBCProvider attributes
#
ResourceType=JDBCProvider
ImplementingResourceType=JDBCProvider
ResourceId=Cell=!{cellName}:JDBCProvider=ID#builtin_jdbcprovider
#

#
#Properties
#
classpath=${DERBY_JDBC_DRIVER_PATH}/derby.jar
name=Derby JDBC Provider (XA)
implementationClassName=org.apache.derby.jdbc.EmbeddedXADataSource
nativepath={}
description=Built-in Derby JDBC Provider (XA)
providerType=Derby JDBC Provider (XA) #readonly
xa=true #boolean
```

Figure 8-2 Properties for a JDBC provider

After the properties are extracted, you can make any necessary changes to the attributes and validate the changes before applying them to the server. You can also create or delete configuration objects.

Example 8-11 on page 329 shows samples of these commands.

Example 8-11 The wsadmin commands for properties based configuration

```
AdminTask.extractConfigProperties('-configData Node=myNode -propertiesFileName myNodeProperties.props')
```

```
AdminTask.validateConfigProperties('-propertiesFileName myNodeProperties.props -reportFile report.txt')
```

```
AdminTask.applyConfigProperties('-propertiesFileName myPropFile.props -validate true')
```

```
AdminTask.deleteConfigProperties('-propertiesFileName myPropFile.props')
```

```
AdminTask.createPropertiesFileTemplates('-propertiesFileName serverTemplate.props -configType Server')
```

Because it is not possible to modify every configuration using properties file-based configuration, do *not* use this tool for backup and recovery. Use the **BackupConfig** and **RestoreConfig** commands in the `<was_home>/bin` directory as the main backup and recovery tools.

Properties file-based configuration: You can find more information about properties file-based configuration in the WebSphere Application Server information center at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=rxml_7propbasedconfig

8.6 Managing WebSphere using script libraries

You can use script libraries to perform a higher level of wsadmin functions than when using a single **wsadmin** command. Only a single line from a library function is needed to perform complex functions. Each script is written in Jython and is often referred to as *the Jython script*. The script libraries are categorized into six types, and the types are subdivided, as listed in Table 8-2.

Python script files are supplied for each Jython class file. The Python files can be read as text files. Table 8-2 shows the scripts and the type of resources they manage. Each script has a set of procedures that perform specific functions.

Table 8-2 The types of script libraries

| TYPE | Python (Jython) script file |
|-------------|---|
| Application | AdminApplication AdminBLA |
| PerfTuning | ApplyPerfTuning |
| Resources | AdminJ2C AdminJDBC AdminJMS AdminResources |

| TYPE | Python (Jython) script file |
|-------------|---|
| Application | AdminApplication AdminBLA |
| Security | AdminAuthorizations |
| Servers | AdminClusterManagement AdminServerManagement |
| System | AdminNodeGroupManagement AdminNodeManagement |
| Utilities | AdminLibHelp AdminUtilities |

Script libraries, their procedures, and syntax: You can find more information about these script libraries in the WebSphere Application Server information center at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=welc_ref_adm_jython

8.6.1 Invoking script libraries

The script libraries are located in the *install_root* /scriptlibraries directory. These libraries are loaded when **wsadmin** starts and are readily available from the **wsadmin** command prompt or to be used from the customized scripts. You can invoke the scripts in interactive mode or script mode.

Interactive mode

Interactive mode is suitable for simple tasks and testing. Using this mode allows you to get the results directly. To invoke a script interactively, start a **wsadmin** session, and enter the script name, procedure, and arguments at the **wsadmin>** prompt (see Example 8-12).

Example 8-12 Using the Jython scripts in interactive mode

```
C:\WebSphereV8.5\AppServer\bin>wsadmin -lang jython
WASX7209I: Connected to process "dmgr" on node DmgrNode02 using SOAP connector;
The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>AdminServerManagement.checkIfServerExists("sys1Node01", "Amember01")
-----
AdminServerManagement: Check if server exists
Node name:                sys1Node01
Server name:               Amember01
Usage: AdminServerManagement.checkIfServerExists("sys1Node01", "Amember01")
-----

'true'
wsadmin>
```

Script file mode (-f)

Using a script file with **wsadmin** is useful when you want to have daily tasks performed automatically or if you need to manage multiple servers. To run in script mode, select the

script libraries to use and merge them into your own script file. Save the custom script as a Python file, and run it from the command line.

Example 8-13 shows a Python file containing two script library commands.

Example 8-13 The script for test.py

```
# Writting by Jython
# Script file name : test.py

AdminServerManagement.checkIfServerExists("sys1Node01", "Amember21")
AdminServerManagement.createApplicationServer("sys1Node01", "Amember21")
```

Example 8-14 shows how to invoke the script.

Example 8-14 Using the invoke test.py script

```
C:\WebSphereV8.5\AppServer\bin>
C:\WebSphereV8.5\AppServer\bin>wsadmin.bat -lang jython -f C:\temp\test.py
```

Customizing scripts

Customizing scripts is an advanced use of the script mode. You can add customized code written in Python or Jython to your script file (the one that calls the script libraries).

Note: Do *not* modify the script libraries. If you need to customize the scripts, you can copy parts of the library code to other files and modify the copied code to improve it or to better suit your needs.

When customizing a script:

1. Run each Jython script in interactive mode and then verify the syntax and the result.
2. Create the script file that will call the script libraries by combining all Jython scripts. Verify that the results are as you expect.
3. Add your additional **wsadmin** commands, written in Python and then verify that the customized script does the work that you intended.

8.6.2 Displaying help for script libraries

You can use the AdminLibHelp script to display each script within a script library. For example, the following command displays each script in the AdminApplication script library:

```
print AdminLibHelp.help("AdminApplication")
```

You can use the help script to display detailed descriptions, arguments, and usage information for a specific script. For example, the following command displays detailed script information for the listApplications script in the AdminApplication script library:

```
print AdminApplication.help('listApplications')
```

Example 8-15 shows sample code for displaying help information for the createApplicationServer procedure in the AdminServerManagement script.

Example 8-15 Help information for a procedure in createApplicationServer

```
wsadmin>print AdminServerManagement.help('createApplicationServer')
WASL2004I: Procedure: createApplicationServer
```

Arguments: nodeName, serverName, (Optional) templateName

Description: Create a new application server

Usage: AdminServerManagement.createApplicationServer(nodeName,
serverName, templateName)
wsadmin>

8.6.3 Application script library

The application scripts provide a set of procedures to manage and configure enterprise applications and business level applications. You can use these scripts individually, or you can combine them in a custom script file to automate application installation, configuration, and management tasks.

The library that is located in the *install_root/scriptlibraries/application/V70* directory contains the following scripts:

- ▶ AdminApplication, which provides procedures to manage enterprise applications
- ▶ AdminBLA, which provides procedures to manage business level applications

We provide information about the AdminApplication script in the next section. For information about the AdminBLA script, see the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=rxml_7libbla%20File%20name:%20rxml_7libbla.html

AdminApplication script

The AdminApplication script contains procedures that allow you to manage enterprise applications. The AdminApplication script uses the following syntax:

`AdminApplication.procedure_name(arguments)`

The AdminApplication script provides procedures to:

- ▶ Install and uninstall applications
- ▶ Update applications
- ▶ Export applications
- ▶ Configure application deployment characteristics
- ▶ Start and stop enterprise and business level applications
- ▶ Query applications

Open the AdminApplication script from *install_root/scriptlibraries/application/V70* to find the complete list of procedures for performing application management tasks.

Example: Installing an application

You can use multiple procedures to install applications. When preparing to install an application, determine the options that you need, and choose the procedure accordingly.

The most basic procedure is `installAppwithNodeAndServerOptions`. If the installation is successful, the installed application successfully message returns.

This procedure uses the following syntax:

`AdminApplication.installAppwithNodeAndServerOptions(app_name, app_location,
node_name, app_server_Name)`

Example 8-16 illustrates the `installAppWithNodeAndServerOptions` procedure.

Example 8-16 Installing the application script library

```
wsadmin>AdminApplication.installAppWithNodeAndServerOptions("IBMUTC",
"C:/IBMUTC.ear", "sys1Node01", "Amember01")
-----
AdminApplication:      Install application with -node and -server options
Application name:      IBMUTC
Ear file to deploy:    C:/IBMUTC.ear
Node name:             sys1Node01
Server name:           Amember01
Usage: AdminApplication.installAppWithNodeAndServerOptions("IBMUTC", "C:/IBMUTC
.ear", "sys1Node01", "Amember01")
-----
WASX7327I: Contents of was.policy file:grant codeBase "file:${application}"
{permission java.security.AllPermission;
};
ADMA5016I: Installation of IBMUTC started.
ADMA5058I: Application and module versions are validated with versions of deploy
ment targets.

CWSAD0040I: The application IBMUTC is configured in the Application Server repos
itory.
ADMA5113I: Activation plan created successfully.
ADMA5011I: The cleanup of the temp directory for application IBMUTC is complete.

ADMA5013I: Application IBMUTC installed successfully.
OK: installAppWithNodeAndServerOptions('IBMUTC', 'C:/IBMUTC.ear', 'sys1Node01',
'Amember01', 'false'):
```

Example: Starting an application

Multiple procedures are also available to start an application. To start the application, choose the most suitable script.

The `startApplicationSingleServer` procedure starts a single application on a single application server. This procedure uses the following syntax:

```
AdminApplication.startApplicationOnSingleServer(app_name, node_name,
app_server_name)
```

Example 8-17 illustrates the `startApplicationSingleServer` procedure.

Example 8-17 Starting an application script library

```
wsadmin>AdminApplication.startApplicationOnSingleServer("IBMUTC","sys1Node01","Ame
mber01")
-----
AdminApplication:      Start an application on a single server
Application name:      IBMUTC
Node name:             sys1Node01
Server name:           Amember01
Usage: AdminApplication.startApplicationOnSingleServer("IBMUTC",
"sys1Node01", "Amember01")
-----
OK: startApplicationOnSingleServer('IBMUTC', 'sys1Node01', 'Amember01', 'false')
```

8.6.4 Resource script library

The Resource script library provides a set of scripts to manage WebSphere resources. The library provides script functions for J2C resources, JDBC providers, and JMS resources at the server scope. If you need to configure resources at the cell, node, or cluster level, you can customize the scripts for this purpose.

The script library is located in the *install_root/scriptlibraries/resources/V70* directory. It contains the following scripts:

- ▶ AdminJ2C script: Provides procedures to configure and query J2C resources.
- ▶ AdminJDBC script: Provides procedures to configure and query JDBC resources.
- ▶ AdminJMS script: Provides procedures to configure and query JMSresources.
- ▶ AdminResources script: Provides procedures to configure mail, resource environment settings, URL provider settings and additional Java Enterprise Edition (JEE) resources.

Open the Resource scripts from *install_root/scriptlibraries/resources* to find the complete list of procedures for managing WebSphere resources

Example: Listing JDBC resources

You can use the `listDataSources` and `listJDBCProviders` procedures of the AdminJDBC script to display a list of configuration IDs for the JDBC providers and data sources in your environment.

The syntax to use each procedure is:

- ▶ `AdminJDBC.listDataSources(ds_name)`
- ▶ `AdminJDBC.listJDBCProviders(jdbc_name)`

Example 8-18 shows the use of the `listDataSources` and `listJDBCProviders` procedures.

Example 8-18 List of JDBC resources

```
wsadmin>AdminJDBC.listDataSources("PLANTSDB")
-----
AdminJDBC:          listDataSources
Optional parameter:
DataSource name:    PLANTSDB
Usage: AdminJDBC.listDataSources("PLANTSDB")
-----
['PLANTSDB(cells/Cell02/nodes/sys1Node01/servers/server1|resources.xml#DataSource_
1183122165968)']
wsadmin>
wsadmin>AdminJDBC.listJDBCProviders("Derby JDBC Provider")
-----
AdminJDBC:          listJDBCProviders
Optional parameter:
JDBC provider name:  Derby JDBC Provider
Usage: AdminJDBC.listJDBCProvider("Derby JDBC Provider")
-----
['"Derby JDBC Provider(cells/Cell02/nodes/sys1Node01/servers/server1|resources.x
```

```
m1#JDBCProvider_1183122153343)"]
```

Example: Creating a J2C connection factory

The createJ2CConnectionFactory procedure in the AdminJ2C script creates a new J2C connection factory in the environment. The result is the configuration ID of the new J2C connection factory.

The arguments are the resource adapter, connection factory name, the connection factory interface, and the Java Naming and Directory Interface (JNDI) name arguments. This procedure uses the following syntax:

```
AdminJ2C.createJ2CConnectionFactory(resource_adapterID, connfactory_name,  
connFactory_interface, jndi_name, attributes)
```

Example 8-19 shows sample coding for creating a J2C connection factory.

Example 8-19 The createJ2CConnectionFactory procedure

```
wsadmin>AdminJ2C.createJ2CActivationSpec("WebSphere MQ Resource  
Adapter(cells/Ce1102/nodes/DmgrNode02/servers/dmgr|resources.xml#J2CResourceAdapte  
r_1234298429000)", "WebSphere MQ Resource Adapter", "javax.jms.MessageListener",  
"jdbc/PlantsByWebSphereDataSourceNONJTA")
```

```
-----  
AdminJ2C:                createJ2CActivationSpec  
J2CResourceAdapter configID:  WebSphere MQ Resource  
Adapter(cells/Ce1102/nodes/DmgrNode02/servers/dmgr|resources.xml#J2CResourceAdapte  
r_1234298429000)  
J2CActivationSpec name:      WebSphere MQ Resource Adapter  
Message listener type:       javax.jms.MessageListener  
jndi name:                   jdbc/PlantsByWebSphereDataSourceNONJTA  
Optional attributes:  
    otherAttributesList  []  
Usage: AdminJ2C.createJ2CActivationSpec("WebSphere MQ Resource  
Adapter(cells/Ce1102/nodes/DmgrNode02/servers/dmgr|resources.xml#J2CResourceAdapte  
r_1234298429000)", "WebSphere MQ Resource Adapter", "javax.jms.MessageListener",  
"jdbc/PlantsByWebSphereDataSourceNONJTA")  
-----
```

```
'"WebSphere MQ Resource Adapter(cells/Ce1102/nodes/DmgrNode02/servers/dmgr|resou  
rces.xml#J2CActivationSpec_1236206121468)'  
wsadmin>
```

8.6.5 Security script library

The security script library provides a script to manage the security. This library is located in the *install_root/scriptlibraries/security/V70* directory.

The AdminAuthorizations script provides procedures to:

- ▶ Configure authorization groups.
- ▶ Remove users and groups from the security authorization settings.
- ▶ Query your security authorization group configuration.

Open the AdminAuthorizations script from *install_root/scriptlibraries/security/V70* to find the complete list of procedures for managing WebSphere security.

Example: Listing the authorization groups

The `listAuthorizationGroups` procedure displays each authorization group in the security configuration. This script does not require arguments.

```
AdminAuthorizations.listAuthorizationGroups()
```

Example 8-20 shows sample coding for listing the authorization groups.

Example 8-20 Listing authorization groups

```
wsadmin>AdminAuthorizations.listAuthorizationGroups()
```

```
-----  
AdminAuthorizations: List authorization groups  
Usage: AdminAuthorizations.listAuthorizationGroups()  
-----
```

```
['sec_group1']
```

8.6.6 Server script library

The server script library provides a set of scripts and their procedures to manage the server and the cluster component.

This library is located in the `install_root/scriptlibraries/servers/V70` directory.

The `AdminServerManagement` script provides procedures to:

- ▶ Start and stop servers
- ▶ Configure servers
- ▶ Query the server configuration
- ▶ Manage server settings

The `AdminClusterManagement` script provides procedures to:

- ▶ Start cluster processes
- ▶ Stop cluster processes
- ▶ Configure clusters
- ▶ Remove clusters and cluster members
- ▶ Query a cluster configuration

Open the Server scripts from `install_root/scriptlibraries/servers/V70` to find the complete list of procedures for managing server and cluster components in WebSphere.

Example: Creating an application server

The `CreateApplicationServer` procedure of the `AdminServerManagement` script creates a new application server. The script requires the node to be running. This procedure uses the following syntax:

```
AdminServerManagement.createApplicationServer(node_name, server_name, Template)
```

Example 8-21 illustrates sample coding for creating an application server.

Example 8-21 Creating an application server

```
wsadmin>AdminServerManagement.createApplicationServer("sys1Node01", "Amember01", "default")
```

```
-----  
AdminServerManagement: Create an application server on a given node
```



```

Node name:          sys1Node01
New Server name:    Amember01
Optional parameter:
Template name:      default
Usage: AdminServerManagement.createApplicationServer("sys1Node01", "Amember01",
"default")
-----
'Amember01(cells/Ce1102/nodes/sys1Node01/servers/Amember01|server.xml#Server_1235061945890)
'

```

Example: Starting an application server

The StartAllServers procedure of the AdminServerManagement script starts all application servers on a node. StartSingleServer starts one server. These procedures use the following syntax:

- ▶ AdminServerManagement.startAllServers(*node_name*)
- ▶ AdminServerManagement.startSingleSever(*node_name*, *server_name*)

Example 8-22 shows sample coding for starting an application server.

Example 8-22 Starting the application server using a single script library

```

wsadmin>AdminServerManagement.startAllServers("sys1Node01")
-----
AdminServerManagement:  Start all servers
Node name:              sys1Node01
Usage: AdminServerManagement.startAllServers("sys1Node01")
-----
Start server: Amember01
Start server: server1
OK: startAllServers('sys1Node01', 'false'):1

wsadmin>AdminServerManagement.startSingleServer("sys1Node01", "Amember01")
-----
AdminServerManagement:  Start single server
Node name:              sys1Node01
Server name:            Amember01
Usage: AdminServerManagement.startSingleServer("sys1Node01", "Amember01")
-----
Start server: Amember01
OK: startSingleServer('sys1Node01', 'Amember01', 'false'):1

```

Example: Stopping application servers

The StopAllServers procedure stops all application servers on a node. The StopSingleServer stops one server. These procedures use the following syntax:

- ▶ AdminServerManagement.stopAllServers(*node_name*)
- ▶ AdminServerManagement.stopSingleSever(*node_name*, *server_name*)

Example 8-23 shows sample coding for stopping application servers.

Example 8-23 Stopping the application server using a single script library

```

wsadmin>AdminServerManagement.stopAllServers("sys1Node01")
-----
AdminServerManagement:  Stop all servers

```

```

Node name:                sys1Node01
Usage: AdminServerManagement.stopAllServers("sys1Node01")
-----
Stop server: Amember01
WASX7337I: Invoked stop for server "Amember01" Waiting for stop completion.
Stop server: server1
WASX7337I: Invoked stop for server "server1" Waiting for stop completion.
OK: stopAllServers('sys1Node01', 'false'):1

wsadmin>AdminServerManagement.stopSingleServer("sys1Node01", "Amember01")
-----
AdminServerManagement: Stop single server
Node name:                sys1Node01
Server name:                Amember01
Usage: AdminServerManagement.stopSingleServer("sys1Node01", "Amember01")
-----
Stop server: Amember01
WASX7337I: Invoked stop for server "Amember01" Waiting for stop completion.
OK: stopSingleServer('sys1Node01', 'Amember01', 'false'):1

```

8.6.7 System management script library

The system management script library provides a set of scripts that manage nodes and node groups. This library is located in the *install_root/scriptlibraries/system/V70* directory. It contains scripts to:

- ▶ AdminNodeManagement
- ▶ AdminNodeGroupManagement

Open the Server scripts from *install_root/scriptlibraries/system/V70* to find the complete list of procedures for managing nodes and node groups in WebSphere.

Example: Querying node group members

The `listNodeGroupMembers` procedure lists the name of each node that is configured within a specific node group. This procedure uses the following syntax:

```
AdminNodeGroupManagement.listNodeGroupMembers(node_group_name)
```

Example 8-24 shows sample coding for querying node group members.

Example 8-24 Listing node group members using the script library

```

wsadmin>AdminNodeGroupManagement.listNodeGroupMembers("DefaultNodeGroup")
-----
AdminNodeGroupManagement: List nodes for a given node group
Optional parameter:
Node group name:          DefaultNodeGroup
Usage: AdminNodeGroupManagement.listNodeGroupMembers("DefaultNodeGroup")
-----

```

```
['DmgrNode02', 'sys1Node01']
```

Example: Synchronizing a node

The `synActiveNodes` and `syncNode` procedures propagate a configuration change. These commands use the following syntax:

- ▶ `AdminNodeManagement.syncActiveNodes()`
- ▶ `AdminNodeManagement.syncNode(node_name)`

Example 8-25 shows sample coding for synchronizing a node.

Example 8-25 Synchronizing the node using the script library

```
wsadmin>AdminNodeManagement.syncNode("sys1Node01")
```

```
-----  
AdminNodeManagement:      syncNode  
nodeName:                  sys1Node01  
Usage: AdminNodeManagement.syncNode("sys1Node01")  
-----  
true  
1
```

8.6.8 Applying performance tuning

Pre-defined performance tuning templates can be applied to an application server or cluster using a python-based tuning script, `applyPerfTuning.py`.

The script is present in `install_root/scriptlibraries/perfTuning/V70` along with three predefined templates: default, peak, and development.

For information about the `applyPerfTuning` script, see the following information center website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=trpf_tuneappserv_script%20File%20name:%20trpf_tuneappserv_script.htm

8.7 Assistance with scripting

When you perform an action in the administrative console, you can use the command assistance feature to show the corresponding scripting commands. This feature allows you to capture and copy scripting commands for use in `wsadmin` scripts. You also have the option to send these as notifications to Rational Application Developer, where you can use the Jython editor to build scripts.

8.7.1 Enabling command assistance

The command assistance feature can help you view `wsadmin` scripting commands in the Jython language for the last action run in the administrative console.

When you perform an action in the administrative console, you can select the **View administrative scripting command for last action** option in the Help area of the window to

display the command equivalent. You can copy and paste this command into a script or command window.

You can also enable additional features, as follows:

1. Click **System administration** → **Console Preferences**. Select the command assistance features that you want to use (see Figure 8-3):
 - Enable command assistance notifications:
Use this option in non-production environments to send notifications containing command assist data. Enabling the notifications allows integration with Rational Application Developer.
 - Log command assistance commands:
This option sends the commands to a log.

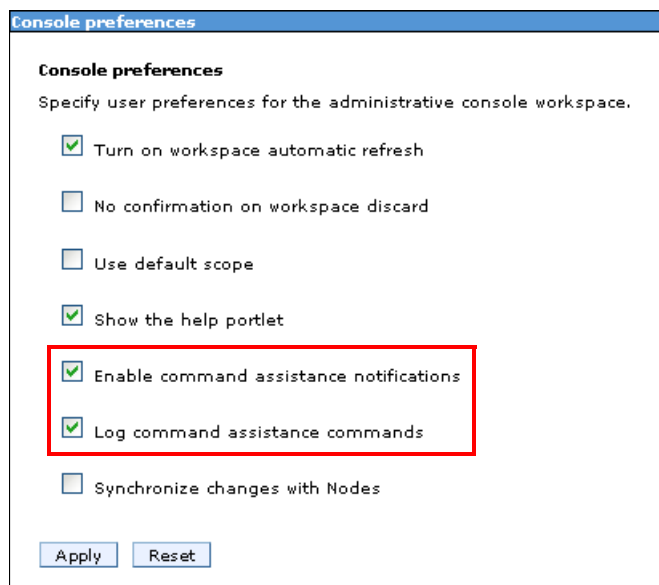


Figure 8-3 Administrative scripting command features that map to actions

2. Click **Apply**.

When you select the **option to log** commands, they are stored in the following location:

`profile_root/logs/AssistanceJythonCommands_user_name.log`

See Example 8-26 for some sample log location.

Example 8-26 Log location

```
C:\WebSphereV8.5\AppServer\profiles\Dmgr01\logs\dmgr  
commandAssistanceJythonCommands_wasadmin.log
```

The first line of each log entry consists of a time stamp and the location within the console where the command was generated. Below the time stamp is the command information. Example 8-27 shows a sample of the log.

Example 8-27 The command assistance - Log content

```
# [2/24/09 12:15:42:890 EST] Business-level applications > New  
AdminTask.createEmptyBLA('[-name sample -description Sample ]')
```

```

# [2/24/09 12:15:42:906 EST] Business-level applications > New
AdminTask.listBLAs('[-blaID WebSphere:blaname=sample -includeDescription true ]')

# [2/24/09 12:15:42:906 EST] Business-level applications > New
AdminTask.listCompUnits('[-blaID WebSphere:blaname=sample -includeDescription true
-includeType true ]')

# [2/24/09 12:15:47:500 EST] Business-level applications > sample
AdminTask.listAssets('[-includeDescription true ]')
# [2/24/09 12:15:47:531 EST] Business-level applications > sample
AdminTask.listBLAs('[-includeDescription true ]')

# [2/24/09 12:15:50:531 EST] Business-level applications > sample > Add
AdminTask.addCompUnit('[-blaID WebSphere:blaname=sample -cuSourceID
WebSphere:blaname=IBMUTC ]')

# [2/24/09 12:15:53:562 EST] Business-level applications > sample > Add
AdminTask.addCompUnit('[-blaID WebSphere:blaname=sample -cuSourceID
WebSphere:blaname=IBMUTC -CUOptions [[WebSphere:blaname=sample WebSphere:blaname=IBMUTC
IBMUTC_0001 "" 1]]]')

# [2/24/09 12:15:57:625 EST] Adding composition unit to the business-level application
AdminConfig.save()

# [2/24/09 12:15:57:890 EST] BLAManagement
AdminTask.listBLAs('[-includeDescription true ]')

# [2/24/09 12:16:01:421 EST] Business-level applications
AdminTask.startBLA('[-blaID WebSphere:blaname=sample ]')

```

8.7.2 Building script files using command assist

The command assist features provide several methods to build scripts. You can copy commands from the Help area of the console or from the log into Jython scripts.

The command assist notifications also provide an integration point with Rational Application Developer. This integration provides tools that allow you to monitor commands as they are created and to insert the monitored commands into a script.

Working with Jython scripts

To work with Jython scripts in Rational Application Developer, you create a Jython project and Jython script files in the project from any perspective. When you open a new Jython script, it opens with the Jython editor.

You can use the Jython editor in Rational Application Developer to perform a variety of tasks. The following list notes some of those tasks:

- ▶ View the administrative console
- ▶ Develop and edit Jython script files
- ▶ Import existing Jython files for structured viewing
- ▶ Set breakpoints for debugging your scripts

The Jython editor has many text editing features, such as syntax highlighting, unlimited undo or redo, and automatic tab indentation.

When you tag a comment in a Jython script with the "#TODO" tag, the editor automatically creates a corresponding task as a reminder in the Tasks view. Then, if you open the task later, the editor synchronizes automatically to that TODO entry in the script source.

Other helpful features are content assist and tips that provide a list of acceptable continuations. The continuation information is dependant on where the cursor is located in a Jython script file or what you just typed. The Jython editor is not integrated to a compiler. As a result, the Jython editor does not perform syntax verification on your scripts.

Using the command assist notifications

The command assistance in the administrative console sends JMX notifications containing command data. You can monitor these notifications from Rational Application Developer. This monitoring requires that you define the server that is producing the notifications as a server in the workspace.

To monitor the commands that are produced as actions, which are taken on the administrative console of the server:

1. In the Servers view, right-click the server, and click **Administration** → **WebSphere administration command assist**. The WebSphere Administration Command view opens.
2. In the Select Server to Monitor list, select the servers that you want the tool to monitor as you interact with its administrative console. The Select Server to Monitor list is available in the toolbar of the WebSphere Administration Command view (see Figure 8-4).

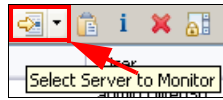


Figure 8-4 Select Server to Monitor icon

Important: You need to start the server that you want to monitor in profile or debug mode. The server is disabled for selection in the Select Server to Monitor list when the server is stopped.

As commands are generated by the console, they display in the WebSphere Application Command Assist view. The commands are shown in WebSphere Administration Command view.

With the Jython script open in the Jython editor and with the monitored command data in the WebSphere Administration Command view, you can insert the commands directly into the script file. Place the cursor in the script file where you want the command to go. Right-click the command, and select **Insert**, as shown in Figure 8-5 on page 343. Double-clicking the command also inserts it into the script.

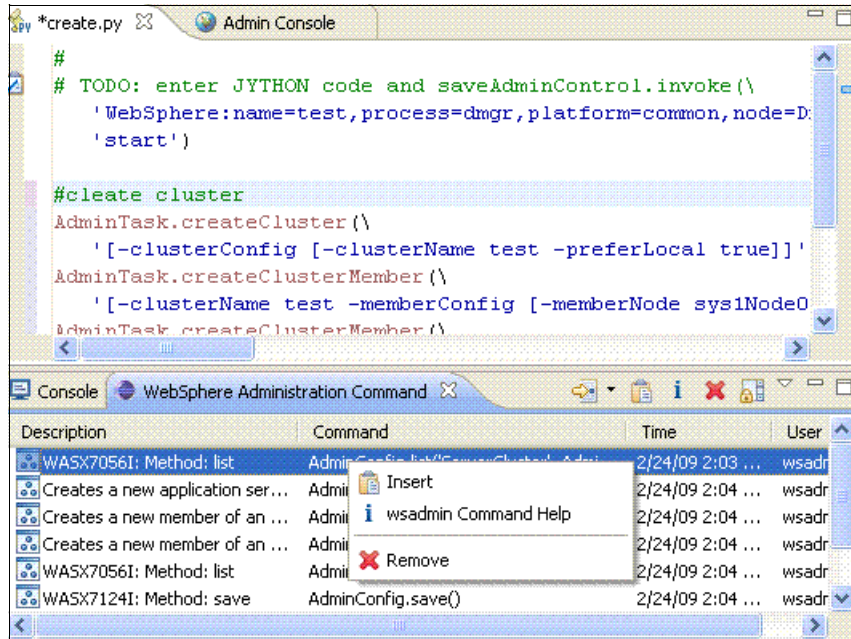


Figure 8-5 Jython editor running the command output automatically

8.8 Example: Using scripts with the job manager

This section provides an example of how to use scripting to automate a WebSphere installation that uses a flexible management environment.

Most companies have routine tasks that occur in different phases of the software development lifecycle. In an environment with multiple WebSphere Application Server installations, automation of these tasks can save a lot of time. The combination of wsadmin scripts, script libraries, and the automated management provided in a flexible management environment provides an automation solution.

8.8.1 Introduction

The scenario that we describe here uses a simple WebSphere environment to illustrate how to automate tasks. You can use these techniques in more complex environments. This scenario contains the following steps:

1. Write a customized script to automate the tasks.
2. Configure the job manager.
3. Verify the results.

Figure 8-6 shows the scenario environment. A single application server is configured on Node B. The deployment manager for the cell is registered to a job manager on Node A.

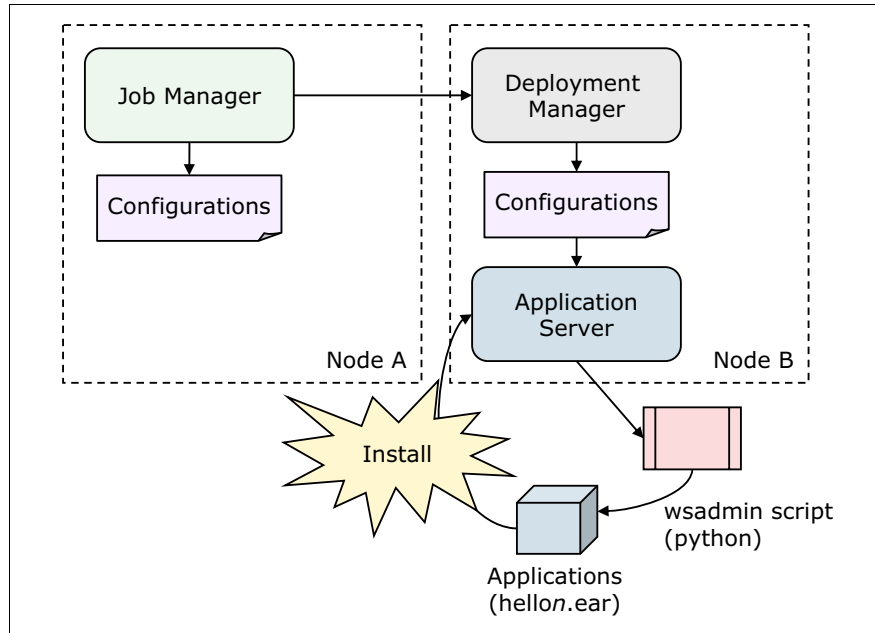


Figure 8-6 The environment details

Applications in this environment are installed frequently, and the administrator needs a quick way to install these applications, which can be accomplished by completing the following steps:

1. A wsadmin script is prepared to install an application. The script makes use of the script libraries.
2. A job is scheduled to run the script at regular intervals.
3. The script checks a text file that lists the new applications to be installed. The new application information stored in a text file includes the application name, application location, node name, and server name (see Figure 8-7).

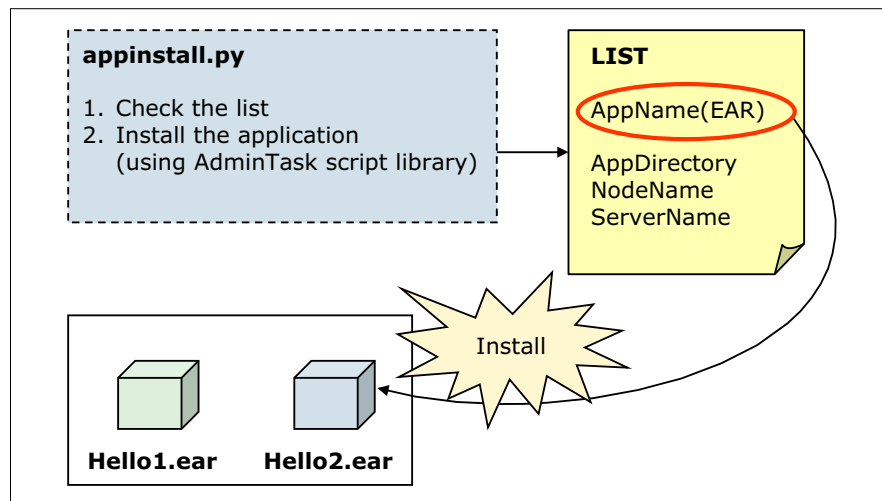


Figure 8-7 Scripting details

4. If the job runs and finds that the application is not installed, the application is then installed. If the application is already installed, it is uninstalled and then re-installed.
5. The text file is renamed *filename.txt.old* when the job is executed.
6. If the job executes and no text file exists, no actions are taken. It is only when you distribute a new text file and application that the job performs the install.

To test, we used the following applications:

- ▶ hello1.ear
- ▶ hello2.ear

8.8.2 Creating the customized script

The AdminApplication script in the script libraries includes procedures that accomplish the installation and update of applications. In this scenario, we used the following procedures from the script libraries:

- ▶ AdminApplication.uninstallApplication
This procedure is used to uninstall an application. The arguments specify the application name.
- ▶ AdminApplication.installAppWithNodeAndServerOptions
This procedure is used to install an application. This procedure is selected because there is only a single server in this environment. In an environment with clustered application servers, use the installAppWithClusterOption procedure instead.

The script file is written in Python and is called `appInstall.py` (see Example 8-28).

Example 8-28 The appInstall.py script

```
#Check the list and install/update the applications.
#

import sys
import java
import os
import re
import time
from java.lang import *

dir = "C:/WebSphereV8.5/AppServer/profiles/dmgr40/downloadedContent/inputfile"
#
sep = System.getProperty("line.separator")
for fname in os.listdir(dir):
    print fname
    path = os.path.join(dir, fname)
    if (os.path.isfile(path) and (not re.match(".*old$",path) and (not
re.match("appInstall.py",fname))):
        print "procesing %s" % (path)
        inp = open(path, 'r')
        for line in inp.readlines():

            itemList = line[:-1].split(',')
            appName = itemList[0]
            earFile = itemList[1]
            nodeName= itemList[2]
```

```

serverName = itemList[3]

# application existence check
print "application existence check"
existAppList = AdminApp.list().split(sep)
isExist = 0
for existApp in existAppList:
    if(appName == existApp):
        isExist = 1
        break
# acquire application manager
print "acquire application manager"

    appMgrID =
AdminControl.queryNames("type=ApplicationManager,node="+nodeName+",process="+serverName+",*" )

# if exists, uninstall application
print "app exists - uninstall"
if( isExist == 1 ):
    appId = ""
    try:
        _excp_ = 0
        appId =
AdminControl.completeObjectName("type=Application,node="+nodeName+",Server="+serverName+",name="
+appName+",*" )
    except:
        _type_, _value_, _tbck_ = sys.exc_info()
        _excp_ = 1
    #endTry
# if running, stop application
print "appID is %s" % (appId)
if(len(appID) > 0):
    print "stopping %s" % (appName)
    stopped = AdminControl.invoke(appMgrID, "stopApplication", appName)
    sleep(1)
# uninstall application
print "Uninstalling %s" % (appName)
AdminApplication.uninstallApplication(appName)

# install application
print "Installing %s" % (appName)
AdminApplication.installAppWithNodeAndServerOptions(appName, earFile, nodeName,
serverName)
print "Starting %s" % (appName)
started = AdminControl.invoke(appMgrID, "startApplication", appName)
sleep(1)

inp.close()
os.rename(fname, fname + ".old")
#endIf
#endFor

```

Example 8-29 shows the hello.txt input file.

Example 8-29 The hello.txt input file

```
hello,/webspherev8.5/appserver/profiles/dmgr40/downloadedContent/appl/hello1.ear,node40a,server40a1
```

The sample script is first tested using **wsadmin** running in script mode. Example 8-30 shows the result of the sample script.

Example 8-30 Testing the sample script

```
C:\WebSphereV8.5\AppServer\profiles\dmgr40\bin>wsadmin -lang jython -f c:\webspherev8.5\appserver\profiles\dmgr40\downloadedContent\appinstall.py -username admin -password admin
WASX7209I: Connected to process "dmgr" on node dmgr40node using SOAP connector;
The type of process is: DeploymentManager
hello.txt
processing C:\WebSphereV8.5\AppServer\profiles\dmgr40\downloadedContent\inputfile\hello.txt
application existence check
acquire application manager
app exists - uninstall
Installing hello
-----
AdminApplication:      Install application with -node and -server options
Application name:      hello
Ear file to deploy:    /webspherev8.5/appserver/profiles/dmgr40/downloadedContent/appl/hello1.ear
Node name:             node40a
Server name:           server40a1
Usage: AdminApplication.installAppWithNodeAndServerOptions("hello", "/webspherev8.5/appserver/profiles/dmgr40/downloadedContent/appl/hello1.ear", "node40a", "server40a1")
-----
ADMA5016I: Installation of hello started.
ADMA5058I: Application and module versions are validated with versions of deployment targets.
ADMA5005I: The application hello is configured in the WebSphere Application Server repository.
ADMA5053I: The library references for the installed optional package are created
.
ADMA5005I: The application hello is configured in the WebSphere Application Server repository.
ADMA5001I: The application binaries are saved in C:\WebSphereV8.5\AppServer\profiles\dmgr40\wstemp\Script120f8e64a06\workspace\cells\Cell140\applications\hello.ear\hello.ear
ADMA5005I: The application hello is configured in the WebSphere Application Server repository.
SECJ0400I: Successfully updated the application hello with the appContextIDForSecurity information.
ADMA5005I: The application hello is configured in the WebSphere Application Server repository.
CWSAD0040I: The application hello is configured in the Application Server repository.
```

```
tory.  
ADMA5113I: Activation plan created successfully.  
ADMA5011I: The cleanup of the temp directory for application hello is complete.  
ADMA5013I: Application hello installed successfully.  
OK: installAppWithNodeAndServerOptions('hello', '/webspherev8.5/appserver/profiles  
/dmgr40/downloadedContent/appl/hello1.ear', 'node40a', 'server40a1', 'false'):
```

8.8.3 Submitting the job

To use the job manager to execute the script:

1. Before running the `appInstall.py` script, you must transfer it along with the `hello.txt` file and the `hello1.ear` file from the job manager to the deployment manager using the `distributeFile` job. To define the directories that are required for this task, see 6.3.2, “Distributing files using the job manager” on page 225.

In Rational Application Developer, export the `appInstall.py` script file and application EAR file to the `jmgr_profile_root/config/temp/JobManager` directory.

Manually copy the `hello.txt` file to the same directory.

2. In the Job manager console, click **Job** → **Submit** to launch the Job properties wizard.
3. Use the Distribute file job to transfer each file. In each case, select the **admin agent** as the target node. The source location refers to the file in the `jmgr_profile_root/config/temp/JobManager` directory. The format of the file is:

`file:/file_name`

4. Distribute the files as follows:
 - Distribute the `hello.txt` file to the following directory:
`dmgr_profile_root/downloadedContent/inputfile`
 - Distribute the `appInstall.py` file to the following directory:
`dmgr_profile_root/downloadedContent`
 - Distribute the `appInstall.py` file to the following directory:
`dmgr_profile_root/downloadedContent/appl`

5. Submit the wsadmin script for execution. Click **Job** → **Submit** to launch the Job properties wizard and then complete the following steps:
 - a. Click **Run wsadmin script** as the job type and then enter a description. Click **Next** (Figure 8-8).

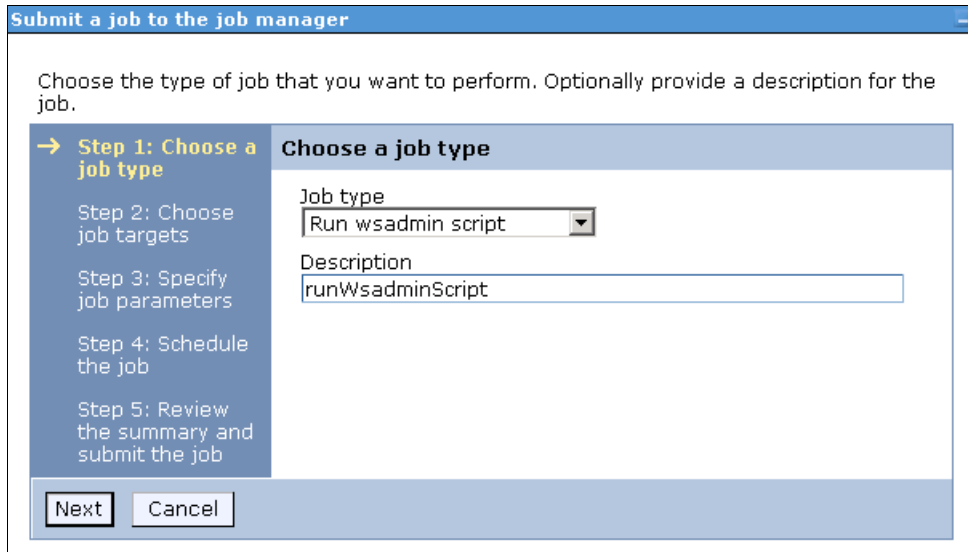


Figure 8-8 Step 1 - Choose the job type

- b. Select the job target. In this case, the **deployment manager node** is selected. Enter the user ID and password that are required for administration tasks on the deployment manager. Click **Next**.
- c. Specify the script location. This location is the same location that you used when you distributed the file. The current directory is the *dmgr_profile_root/downloadedContent* directory. Click **Next** (Figure 8-9).

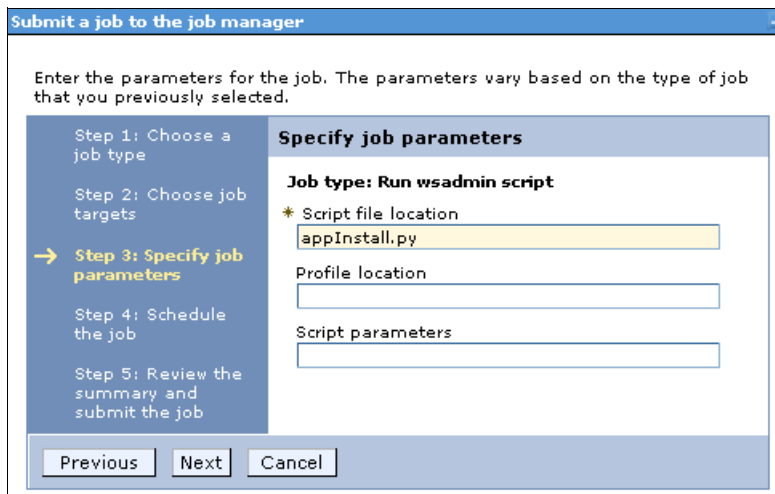


Figure 8-9 Step 3 - Specify job parameters

- d. Next, you can schedule the job to run once or to run multiple times at regular intervals. You can also define when the job is available for execution and when it expires.

In this example, the execution of the script from the job manager is tested first using the “Run once” option in the Availability interval field. After the job is tested, you can repeat the job submit process and set an interval so that the job runs automatically.

Click **Next**.

- e. Review the summary and then click **Finish**.
6. Monitor the job status to ensure that it completes successfully. If the job does not complete successfully, click the **Job ID** to see the messages produced.

8.8.4 Verifying the results

You can verify the results by displaying the new application in the administrative console, starting the application and then accessing the application from a web browser. Use the following steps:

1. Access the deployment manager console, expand **Applications** from the navigation tree, and click **Application Types** → **WebSphere enterprise** application.
2. Verify that the new application is in the list.

8.9 Online resources

The following websites are also relevant as further information sources:

- ▶ Command assistance simplifies administrative scripting in WebSphere Application Server:
http://www.ibm.com/developerworks/websphere/library/techarticles/0812_rhodes/0812_rhodes.html
- ▶ Sample Scripts for WebSphere Application Server:
<http://www.ibm.com/developerworks/websphere/library/samples/SampleScripts.html>
- ▶ Properties based configuration:
http://www.ibm.com/developerworks/websphere/techjournal/0904_chang/0904_chang.html



Accessing relational databases from WebSphere

When an application or WebSphere component requires access to a database, that database must be defined to WebSphere as a data source. Two basic definitions are required:

- ▶ A JDBC provider definition describes a vendor-provided JDBC driver, including the type of database access that it provides and the location of the files that provide the implementation.
- ▶ A data source definition defines which JDBC provider to use, the name and location of the database, and other connection properties.

In this chapter, we provide information about the various considerations for accessing databases from WebSphere.

We cover the following topics:

- ▶ JDBC resources
- ▶ Steps to define access to a database
- ▶ Connecting to an IBM DB2 database
- ▶ Connecting to an Oracle database
- ▶ Connecting to an SQL Server database
- ▶ Configuring connection pooling properties
- ▶ Shared and unshared connections
- ▶ Troubleshooting database access problems

9.1 JDBC resources

The JDBC API provides a programming interface for data access of relational databases from the Java programming language. WebSphere Application Server V8.5 supports the following JDBC APIs:

- ▶ JDBC 4.0
- ▶ JDBC 3.0
- ▶ JDBC 2.1 and Optional Package API (2.0)

In the following sections, we explain how to create and configure data source objects for use by JDBC applications. This is the only method to connect to a database if you intend to use connection pooling and distributed transactions.

The following database platforms are supported for JDBC:

- ▶ DB2
- ▶ Oracle
- ▶ Sybase
- ▶ IBM Informix®
- ▶ SQL Server
- ▶ Apache Derby (test and development only)
- ▶ Third-party vendor JDBC data source using SQL99 standards

9.1.1 JDBC providers and data sources

A *data source* represents a real-world source of data, such as a relational database. When a data source object is registered with a JNDI naming service, an application can retrieve it from the naming service and use it to make a connection to the associated database.

Information about the data source and how to locate it, such as its name, the server on which it resides, its port number, and so on, is stored in the form of *properties* on the DataSource object. Storing this information in this manner makes an application more portable because it does not need to hard code a driver name, which often includes the name of a particular vendor. It also makes maintaining the code easier because if, for example, the data source is moved to a different server, all that needs to be done is to update the relevant property in the data source. None of the code using that data source needs to be touched.

To increase application performance and reduce workload on the database, connections to it are typically pooled. In other words, when the application closes the connection, the connection is returned to a connection pool, rather than being destroyed.

Data source *classes* and JDBC *drivers* are implemented by the data source vendor. By configuring a JDBC provider, you provide information about the set of classes that are used to implement the data source and the database driver. Also, you provide the environment settings for the DataSource object. A driver can be written purely in the Java programming language or in a mixture of the Java programming language and the Java Native Interface (JNI) native methods.

In the next sections, we describe how to create and configure data source objects and how to configure the connection pools used to serve connections from the data source.

9.1.2 WebSphere support for data sources

The programming model for accessing a data source is:

1. An application retrieves a DataSource object from the JNDI naming space.
2. After the DataSource object is obtained, the application code calls the `getConnection()` method on the data source to get a Connection object. The connection is obtained from a pool of connections.
3. After the connection is acquired, the application sends SQL queries or updates to the database.

In addition to the data source support for Java EE6, Java EE 5, J2EE 1.4, and J2EE 1.3 applications, support is also provided for J2EE 1.2 data sources. The two types of support differ in how connections are handled. However, from an application point of view, they look the same.

Note: The `@Resource` annotation can be used to declare a reference to a datasoure. The container injects the data source referred to by `@Resource` into the component either at runtime or when the component is initialized, depending on whether field/method injection or class injection is used.

Data source support

The primary data source support is intended for J2EE 1.3 and J2EE 1.4 and Java EE 5 and Java EE6 applications. Connection pooling is provided by two components: a JCA Connection Manager and a relational resource adapter. See Figure 9-1.

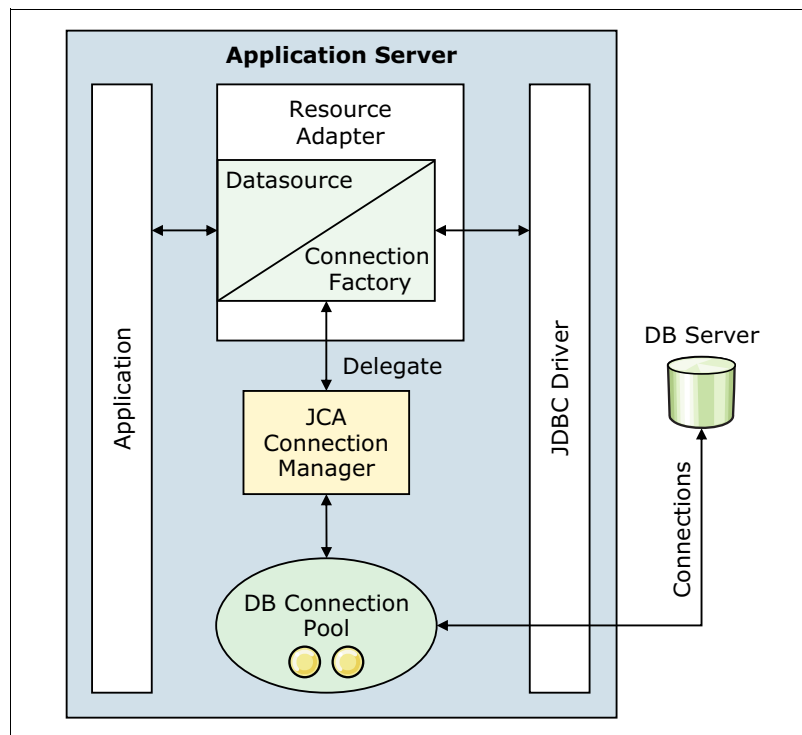


Figure 9-1 Resource adapter in J2EE connector architecture

The JCA Connection Manager provides connection pooling, local transactions, and security support.

The relational resource adapter provides JDBC wrappers and the JCA common client interface (CCI) implementation that allows Bean Managed Persistence (BMP), JDBC applications, and Container Managed Persistence (CMP) beans to access the database.

Figure 9-2 shows the relational resource adapter model.

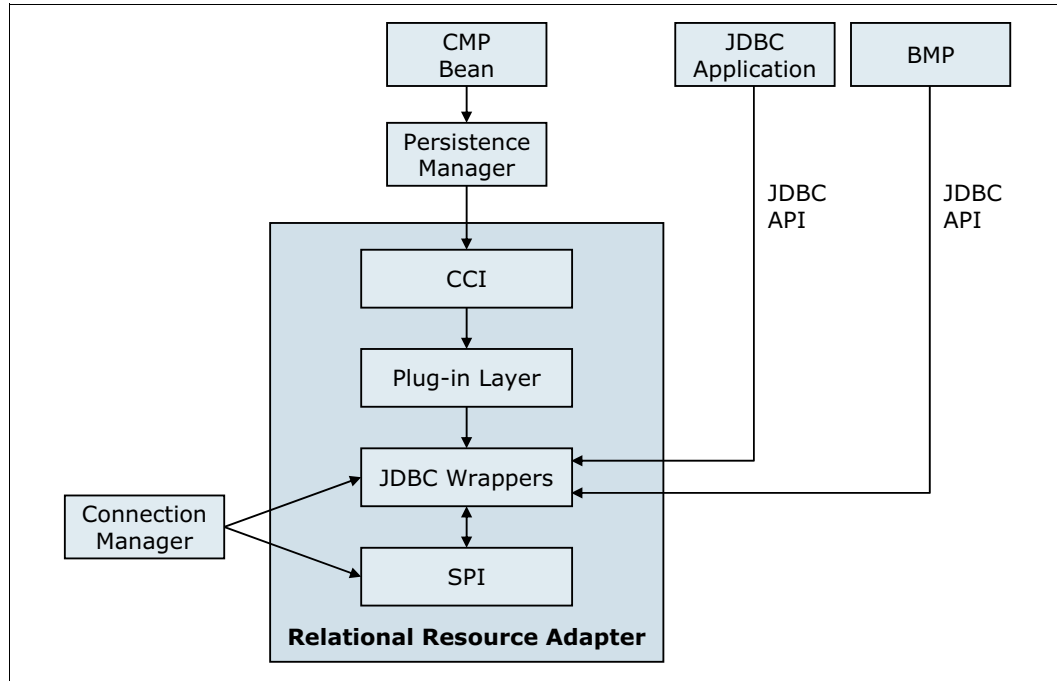


Figure 9-2 Persistence resource adapter model

WebSphere Application Server has a Persistence Resource Adapter that provides relational persistence services to EJB beans and provides database access to BMP and JDBC applications. The Persistence Resource Adapter has two components:

- ▶ The Persistence Manager, which supports the EJB CMP persistence model
- ▶ The Relational Resource Adapter

The Persistence Resource Adapter code is included in the following Java packages:

- ▶ The `com.ibm.ws.rsadapter.cci` package contains the CCI implementation and JDBC wrappers.
- ▶ The `com.ibm.ws.rsadapter.spi` package contains the service provider interface (SPI) implementation.
- ▶ The `com.ibm.ws.rsadapter.jdbc` package contains all of the JDBC wrappers.
- ▶ The `com.ibm.websphere.rsadapter` package contains `DataStoreHelper`, `WSCallerHelper`, and `DataAccessFunctionSet`.

The Relational Resource Adapter is the Persistence Manager's vehicle to handle data access to and from the back-end store, providing relational persistence services to EJB beans. The implementation is based on the J2EE Connector Architecture (JCA) specification and implements the JCA CCI and SPI interfaces.

When an application uses a data source, the data source uses the JCA connector architecture to get to the relational database.

For an EJB, the sequence is:

1. An EJB performs a JNDI lookup of a data source connection factory and issues a `getConnection()` request.
2. The connection factory delegates the request to a connection manager.
3. The connection manager looks for an instance of a connection pool in the application server. If no connection pool is available, the manager uses the `ManagedConnectionFactory` to create a physical, or nonpooled, connection.

Version 4 data source

WebSphere Application Server V4 provided its own JDBC connection manager to handle connection pooling and JDBC access. This support is included with WebSphere Application Server V8.5 to provide support for J2EE 1.2 applications. If an application chooses to use a Version 4 data source, the WebSphere Application Server V8.5 application has the same connection behavior as in Version 4 of the application server.

Use the Version 4 data source for the following purposes:

- ▶ J2EE 1.2 applications: All EJB beans, JDBC applications, or Version 2.2 servlets must use the Version 4 data source.
- ▶ EJB 1.x modules with 1.1 deployment descriptor: All of these modules must use the Version 4 data source.

9.2 Steps to define access to a database

The following steps are involved in creating a data source:

1. Verify that connection to the database server is supported by WebSphere Application Server. Refer to the following website for more information:
<http://www-304.ibm.com/support/docview.wss?rs=180&uid=swg27006921#8.5>
2. Ensure that the database is created and can be accessed by the systems that will use it.
3. Ensure that the JDBC provider classes are available on the systems that will access the database. If you are not sure which classes are required, consult the documentation for the provider.
4. Create an authentication alias that contains the user ID and password that will be used to access the database.
5. Create a JDBC provider. The JDBC provider gives the class path of the data source implementation class and the supporting classes for database connectivity. This is vendor-specific.

The information center provides information about JDBC driver support and requirements. To determine if your provider is supported, refer to the JDBC Provider Summary article at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=/com.ibm.websphere.nd.doc/ae/udat_minreq.html

6. Create a data source. The JDBC data source encapsulates the database-specific connection settings. You can create many data sources that use the same JDBC provider.
7. Save the changes to the master repository and, in case it is a Network Deployment environment, synchronize with the nodes involved.
8. Test the connection to the data source.
9. Review and adjust the connection pool settings (do this on a periodic basis).

z/OS note: Always run a test connection with server scope when using the DB2 Universal JDBC Driver Provider Type 2 in WebSphere Application Server Network Deployment for z/OS environment. The runtime resource manager does not run in node agent. The test will therefore display a failure to load the type 2 native driver library.

9.3 Creating an authentication alias

An authentication alias is a feature that encrypts the password used by the adapter to access the database. After an authentication alias is created, you can use it when you configure the adapter (instead of directly typing the user ID and password). Using an authentication alias eliminates the need to store the password in clear text in an adapter configuration property, where it might be visible to others.

The examples in this chapter assume that the database is password protected and that the user ID and password are defined at run time.

To create a J2C authentication alias that contains the user ID and password that is required to access the database:

1. Open the Administration Console for WebSphere Application Server.
2. Logon to Administration Console, and click **Security** → **Global security**.
3. In the Authentication area, expand **Java Authentication and Authorization Service**, and click **J2C authentication data**.
4. Click **New**.
5. Enter an alias name, user ID, and password, as shown in Figure 9-3. The alias name is used later as the unique identifier when creating an application resource reference. The user ID and password must have authority to access the database.

The screenshot shows a dialog box titled "Global security > JAAS - J2C authentication data > New". Below the title bar, it says "Specifies a list of user identities and passwords for Java(TM)". Under the "General Properties" section, there are four fields: "Alias" with the value "TESTdbAlias", "User ID" with the value "testdb", "Password" which is masked with six dots, and an empty "Description" field. At the bottom of the dialog are four buttons: "Apply", "OK", "Reset", and "Cancel".

Figure 9-3 Define an authentication alias

6. Click **OK**.

9.4 Connecting to an IBM DB2 database

In this section, we illustrate how to configure a JDBC provider using a DB2 provider as an example.

9.4.1 Creating the JDBC provider

To create a JDBC provider, complete the following steps from the administrative console:

1. Ensure that the implementation classes for the provider are available to the system. The class files will need to be located on each system where the application servers will run.
2. In the administrative console, expand **Resources** → **JDBC** in the navigation tree.
3. Click **JDBC Providers**.
4. Select the scope. (Although you can click **All scopes** to view all resources, you must select a specific scope to create a resource.)

Note: JDBC resources are created at a specific scope level. The data source scope level is inherited from the JDBC provider. For example, if we create a JDBC provider at the node level and then create a data source using that JDBC provider, the data source inherits:

- ▶ The JDBC provider settings, such as class path, implementation class, and so on
- ▶ The JDBC provider scope level

In this example, if the scope were set to node-level, all application servers running on that node register the data source in their name space.

The administrative console now shows all of the JDBC providers that are created at that scope level.

5. Click **New** to start the wizard and to create a new JDBC provider.

6. In Step 1 of the wizard, define the type of provider you will use. See Figure 9-4.

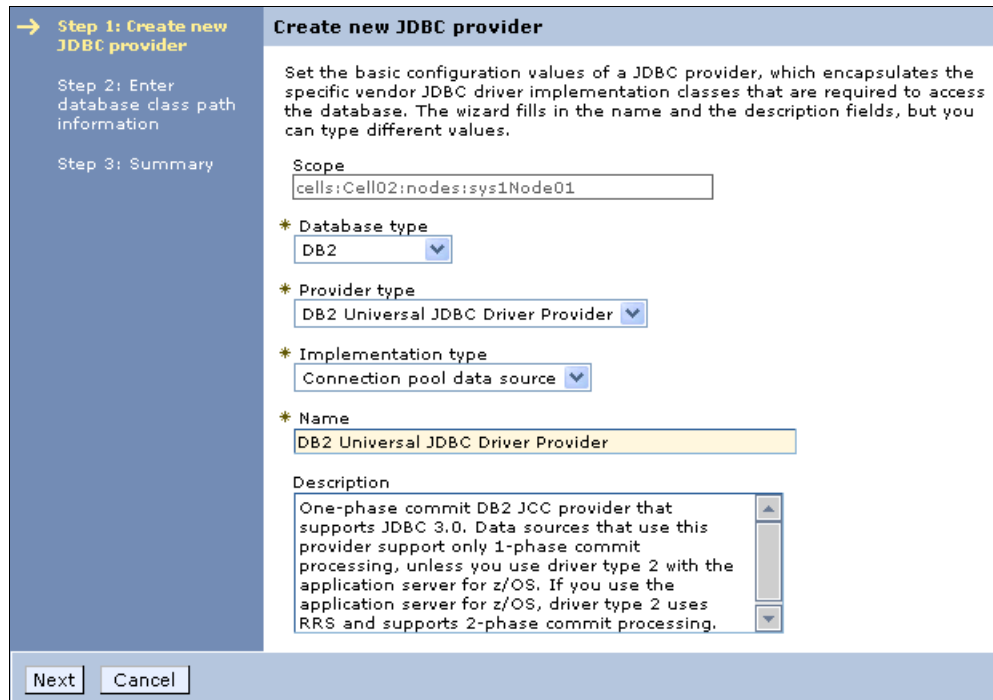


Figure 9-4 Define a new JDBC provider: Window 1

Specify the following information:

- Database type

Select the vendor-specific database type. If the database type you need is not in the list, click **User-defined**, and consult the vendor documentation for the specific properties that are required.

- Provider type

Select from a predefined list of supported provider types, based on the database type that you select.

- Implementation type

Select from the implementation types for the provider type that you selected.

- Name

Specify a name for this driver.

Click **Next**.

7. The settings window for your JDBC database class path opens. Figure 9-5 on page 359 shows the configuration window for the DB2 Universal JDBC Provider.

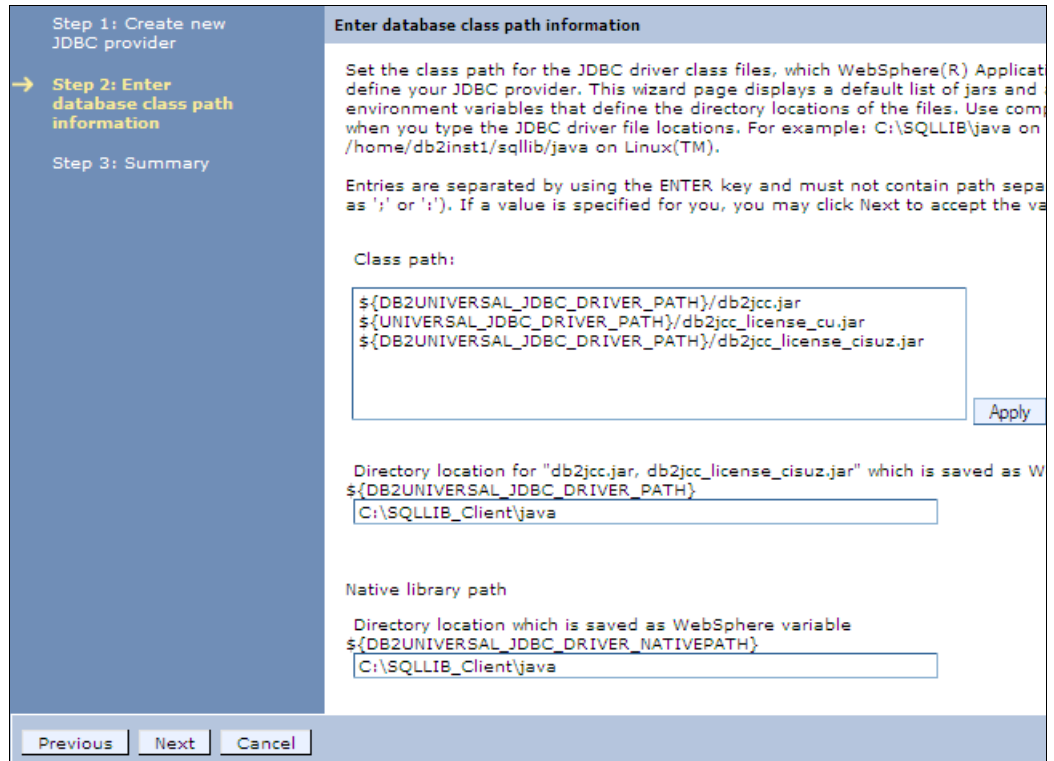


Figure 9-5 Define a new JDBC provider - Window 2

Enter the JDBC provider properties:

- Class path

This field is a list of paths or JAR file names that together form the location for the resource provider classes. This field is pre-set using variable names that are specific to each type of provider. If you are creating a user-defined provider, specify the entries by pressing **Enter** between each entry.

The remaining properties are dependent on the type of provider. They represent the variables that are used in the class path and their value. If you enter a value for a variable on this window, the corresponding variables are populated automatically with these values. Conversely, if the variables are already defined, these fields are populated with the variables.

You can view or modify the variables by clicking **Environment** → **WebSphere Variables** in the navigation menu.

Because this example is for DB2, the following fields are available:

- Path to db2jcc.jar, db2jcc_license_cisuz.jar

This field specifies the values for the global variable DB2UNIVERSAL_JDBC_DRIVER_PATH, which indicates the class path jar's location.

- Native Library Path

This field is an optional path to any native libraries. Entries are required if the JDBC provider chosen uses non-Java, or native, libraries. The global variable for this is UNIVERSAL_JDBC_DRIVER_NATIVEPATH.

Click **Next**.

8. After verifying the settings, click **Finish**.

9. After the JDBC providers collection window is displayed, click the name of the just created JDBC provider and then click Data Sources under the Additional Properties section.

Tip: To make a data source available on multiple nodes using different directory structures, complete the following steps using the administrative console:

1. Define the JDBC provider and data source at the cell scope. Use WebSphere environment variables for the class path and native path.
2. Define the variables at the node scope for each node to specify the driver location for the node.

For example, `${DRIVER_PATH}` can be used for the class path in the provider definition. You can then define a variable called `${DRIVER_PATH}` at the cell scope to act as a default driver location. Next, you can override that variable on any node by defining `${DRIVER_PATH}` at the node scope. The node-level definition takes precedence over the cell-level definition.

9.4.2 Creating the data source

Data sources are associated with a specific JDBC provider and can be viewed or created from the JDBC provider configuration window. You have two options when creating a data source, depending on the J2EE support of the application:

- ▶ J2EE 1.2 application: All EJB 1.1 enterprise beans, JDBC applications, or Servlet 2.2 components must use the 4.0 data source.
- ▶ J2EE 1.3 (and subsequent releases) application:
 - EJB 1.1 module: All EJB 1.x beans must use the 4.0 data source.
 - EJB 2.0 (and subsequent releases) module: Enterprise beans that include container-managed persistence (CMP) Version 1.x, 2.0 and beyond must use the new data source.
 - JDBC applications and Servlet 2.3+ components: Must use the new data source.

In this section, we provide information about creating or modifying data sources for Java EE6, Java EE5, J2EE 1.4, and J2EE 1.3 applications. For information about using data sources with J2EE 1.2 applications, see the topic “Data sources (Version 4)” in the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=/com.ibm.websphere.nd.doc/ae/udat_minreq.html

The administrative console provides a wizard that helps you create a data source. Keep in mind, however, that although the wizard provides a good way to establish connections quickly, it also establishes default-sized connection pool settings that you need to tune properly before production.

Complete the following steps to create a data source:

1. Expand **Resources** → **JDBC** in the navigation tree, and click **Data sources**.
2. Select the scope. Although you can select **All** to view all resources, you must select a specific scope to create a resource.

The scope determines which applications can use this data source. Select the narrowest scope that is required, while also ensuring that the applications that require the resource can access it. For information about selecting a scope, see “Selecting a scope” on page 199.

3. Click **New** to start the wizard and create a new data source. See Figure 9-6.

The screenshot shows a wizard window titled "Enter basic data source information". On the left, a vertical sidebar lists five steps: "Step 1: Enter basic data source information" (highlighted with a yellow arrow), "Step 2: Select JDBC provider", "Step 3: Enter database specific properties for the data source", "Step 4: Setup security aliases", and "Step 5: Summary". The main area contains the following text: "Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database." Below this is a requirement: "Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification." There are three input fields: "Scope" with the value "cells:Cell02:nodes:sys1Node01", "* Data source name" with the value "TDSDB2", and "* JNDI name" with the value "jdbc/testdb". At the bottom are "Next" and "Cancel" buttons.

Figure 9-6 Data source general properties

Specify the following information:

- Data source name: This field is a name by which to administer the data source. Use a name that is suggestive of the database name or function.
- JNDI name: This field refers to the data source's name as registered in the application server's name space.

When installing an application that contains modules with JDBC resource references, the resources need to be bound to the JNDI name of the resources, for example, `jdbc/<database_name>`.

Click **Next**.

4. Select an existing JDBC provider or create a new one, as shown in Figure 9-7.

The screenshot shows a wizard window titled "Select JDBC provider". On the left, a vertical sidebar lists four steps: "Step 1: Enter basic data source information", "Step 2: Select JDBC provider" (highlighted with a yellow arrow), "Step 3: Enter database specific properties for the data source", and "Step 4: Summary". The main area contains the text: "Specify a JDBC provider to support this data source." There are two radio buttons: "Create new JDBC provider" (unselected) and "Select an existing JDBC provider" (selected). Below the radio buttons is a dropdown menu showing "DB2 Universal JDBC Driver Provider". At the bottom are "Previous", "Next", and "Cancel" buttons.

Figure 9-7 Select a JDBC provider

In Figure 9-7, you can select a JDBC provider or create a new one. If you create a new JDBC provider, you are routed through the windows shown in 9.4.1, "Creating the JDBC provider" on page 357. If you select an existing JDBC provider, continue with the next step.

In this case, select an existing JDBC provider, and click **Next**.

The entries shown in Figure 9-8 on page 362 are specific to the JDBC driver and data source type, which show the properties for the Universal data source.

| Step 1: Enter basic data source information Step 2: Select JDBC provider → Step 3: Enter database specific properties for the data source Step 4: Setup security aliases Step 5: Summary | Enter database specific properties for the data source | | | | | | | | | | |
|--|--|------|-------|---------------|---|-----------------|--------|---------------|-----|---------------|-------|
| | Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource. | | | | | | | | | | |
| | <table border="1"> <thead> <tr> <th>Name</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>* Driver type</td> <td>4</td> </tr> <tr> <td>* Database name</td> <td>TESTDB</td> </tr> <tr> <td>* Server name</td> <td>T60</td> </tr> <tr> <td>* Port number</td> <td>50000</td> </tr> </tbody> </table> | Name | Value | * Driver type | 4 | * Database name | TESTDB | * Server name | T60 | * Port number | 50000 |
| Name | Value | | | | | | | | | | |
| * Driver type | 4 | | | | | | | | | | |
| * Database name | TESTDB | | | | | | | | | | |
| * Server name | T60 | | | | | | | | | | |
| * Port number | 50000 | | | | | | | | | | |
| | <input checked="" type="checkbox"/> Use this data source in container managed persistence (CMP) | | | | | | | | | | |
| <input type="button" value="Previous"/> <input type="button" value="Next"/> <input type="button" value="Cancel"/> | | | | | | | | | | | |

Figure 9-8 Database-specific properties

Specify the following information:

- Driver type: The type of JDBC Driver (2 or 4) used to access the database. To determine the best type of driver to use for your circumstances, consult the documentation for the specific driver that you use.

In general, use type 2 for databases on the same system as the application server and type 4 for remote databases.

- Database Name: The name of the database (or the cataloged alias).
- Server name and port: The database server name and its listening port (the default for DB2 is 50000).
- Container managed persistence (CMP): This field specifies if the data source is to be used for container managed persistence of EJB beans.

Deep-dive: Selecting the **Use this data source in container managed persistence (CMP)** option causes a CMP connection factory that corresponds to this data source to be created for the relational resource adapter. The name of the connector factory that is created is `<datasourcename>_CF` and the connector factory is registered in JNDI under the entry `eis/<jndi_name>_CMP`.

To view the properties of the just created connection factory, click **Resources** → **Resource Adapters** → **Resource Adapters**. Select the **Show built-in resources** option in the preferences. Click **WebSphere Relational Resource Adapter** → **CMP Connection Factories**. Be sure to set the scope so that it is the same scope as that for the data source.

Click **Next**.

5. Select or define a new J2C authentication alias for the database. The authentication alias simply contains the user ID and password required to access the database. This window allows you to select an already created authentication value or create a new one. If you select an existing authentication alias, continue with the next step.

The page provides the following options:

- Component-managed authentication alias: This alias is used for database authentication at run time. If the database is not secured, setting database authentication is not required. This is not recommended for a production environment.

- Container-managed authentication alias: Specifies authentication data, which is a JAAS - J2C authentication data entry for container-managed sign on to the resource. Depending on the value that is selected for the Mapping-configuration alias setting, you can disable this setting.
- Mapping-configuration alias: Specifies the authentication alias for the Java Authentication and Authorization Service (JAAS) mapping configuration that is used by this connection factory. The DefaultPrincipalMapping JAAS configuration maps the authentication alias to the user ID and password.

In Figure 9-9, the already existing authentication alias, 't60Node01Cell/samples', is selected.

Figure 9-9 Specify the authentication alias

Click **Next**.

6. A summary of the options that you chose displays. Click **Next** to create the data source.

The new data source is listed in the table of resources. You can test the new connection by selecting the check box to the left of the data source and clicking **Test Connection**. You can view or modify settings for the new data source by clicking the name in the resources list.

9.5 Connecting to an Oracle database

This section illustrates a connection to an Oracle Express 11g database.

Ensure that the implementation classes for the provider are available to the system. The class files need to be located on each system where the application servers will run.

9.5.1 Creating the JDBC provider

Complete the following steps to create the JDBC provider:

1. In the administrative console, expand **Resources** → **JDBC** in the navigation tree.
2. Click **JDBC Providers**.

3. Select the scope. (Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.)
4. Click **New** to start the wizard and to create a new JDBC provider.
5. In step 1 of the wizard, define the type of provider that you will use. See Figure 9-10.

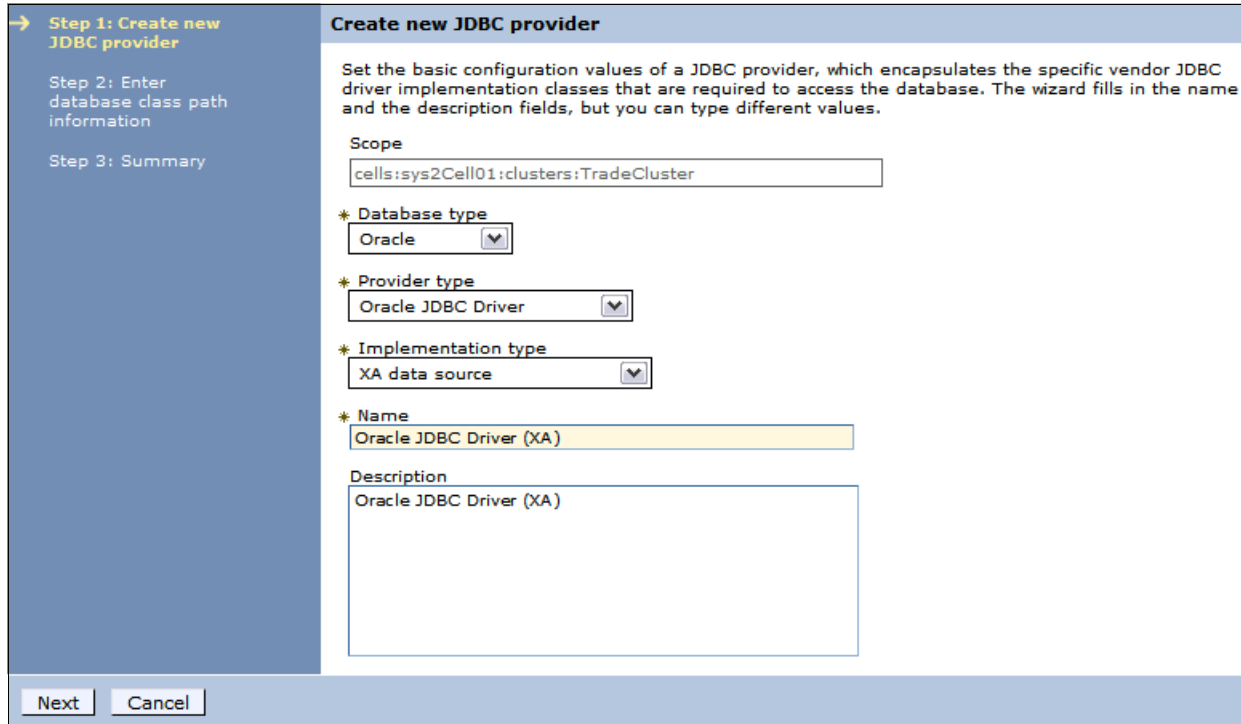


Figure 9-10 Define a new Oracle JDBC provider: Step 1

The database type is Oracle and the provider type is Oracle JDBC driver.

Options of implementation type are XA data source or connection pool data source. XA data source types support two-phase commit transactions.

Click **Next**.

6. In the next window (Figure 9-11), enter the directory location for the Oracle JDBC drivers. In this example, the `ojdbc6.jar` is selected by the wizard.

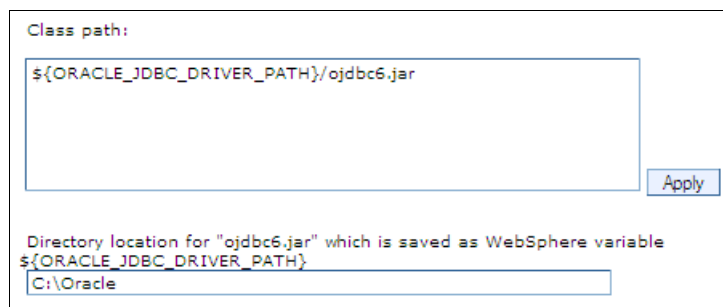


Figure 9-11 Define a new Oracle JDBC provider - Step 2

If you defined a variable named `ORACLE_JDBC_DRIVER_PATH` and set its value, that value is displayed in the directory location field. If you enter a value here, it is saved in the variable.

Click **Next**.

7. Review the summary of the settings, and click **Finish**. The new JDBC provider displays in the list of providers.

9.5.2 Creating the data source

Complete the following steps to create a data source:

1. Expand **Resources** → **JDBC** in the navigation tree, and click **Data sources**.
2. Select the scope. Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.
3. Click **New** to create a new data source and to start a wizard. See Figure 9-12.

The screenshot shows a wizard window titled "Enter basic data source information". On the left, a navigation pane lists five steps: "Step 1: Enter basic data source information" (highlighted with a yellow arrow), "Step 2: Select JDBC provider", "Step 3: Enter database specific properties for the data source", "Step 4: Setup security aliases", and "Step 5: Summary". The main content area contains the following text: "Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database. Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification." Below this text are three input fields: "Scope" with the value "cells:sys2Cell01:clusters:TradeCluster", "* Data source name" with the value "orac-trade-ds", and "* JNDI name" with the value "jdbc/trade-1-ds". At the bottom, there are "Next" and "Cancel" buttons.

Figure 9-12 Create a data source - Step 1

Enter a name for the new data source. This is used for administrative purposes. Enter the JNDI name that will be used to access the data source, and click **Next**.

4. Select the Oracle JDBC driver, and click **Next**. See Figure 9-13.

The screenshot shows a wizard window titled "Select JDBC provider". On the left, a navigation pane lists five steps: "Step 1: Enter basic data source information", "Step 2: Select JDBC provider" (highlighted with a yellow arrow), "Step 3: Enter database specific properties for the data source", "Step 4: Setup security aliases", and "Step 5: Summary". The main content area contains the following text: "Specify a JDBC provider to support the datasource. If you choose to create a new JDBC provider, it will be created at the same scope as the datasource. If you are selecting an existing JDBC provider, only those providers at the current scope are available from the list." Below this text are two radio button options: "Create new JDBC provider" (unselected) and "Select an existing JDBC provider" (selected). Below the second option is a dropdown menu showing "Oracle JDBC Driver (XA)". At the bottom, there are "Previous", "Next", and "Cancel" buttons.

Figure 9-13 Create a data source - Step 2

5. Enter the properties for the database, as shown in Figure 9-14 on page 366.

Enter database specific properties for the data source

Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.

| Name | Value |
|--------------------------------|---|
| * URL | <code>jdbc:oracle:thin:@sys2.itso.ra</code> |
| * Data store helper class name | Oracle11g data store helper |

Use this data source in container managed persistence (CMP)

Figure 9-14 Create a data source - Step 3

Specify the following information:

- The URL for the connection to the XA database is in the following format:

`jdbc:oracle:thin:@host_name:port:service`

In this case:

`jdbc:oracle:thin:@sys2.itso.ra1.ibm.com:1521:XA`

- Select the data store helper class name.

Click **Next**.

6. Select the authentication alias that will provide the user ID and password required to access the database. This window allows you to select an already created authentication value or create a new one. If you select an existing authentication alias, continue with the next step. Here, the existing authentication alias, 'sys2CellManager/oracle_user', is selected.

The page provides the following options:

- Authentication alias for XA Recovery: This field is used to specify the authentication alias that you must use during XA recovery processing. If this alias name is changed after a server failure, the subsequent XA recovery processing uses the original setting that was in effect before the failure. The database identity for the XA recovery authentication alias on a data source must have authorization to do XA recovery. If the resource adapter does not support XA transactions, this field does not display. The default value for this field is derived from the selected alias for application authentication, if one is specified.
- Component-managed authentication alias: This alias is used for database authentication at run time. If the database is not secured, setting database authentication is not required. This is not recommended for a production environment.
- Container-managed authentication alias: Specifies authentication data, which is a JAAS - J2C authentication data entry for container-managed sign on to the resource. Depending on the value that is selected for the Mapping-configuration alias setting, you can disable this setting.
- Mapping-configuration alias: Specifies the authentication alias for the Java Authentication and Authorization Service (JAAS) mapping configuration that is used by this connection factory. The DefaultPrincipalMapping JAAS configuration maps the authentication alias to the user ID and password.

Click **Next**. See Figure 9-15 on page 367.



Figure 9-15 Create a data source: Step 4

Note: If the database does not support user ID and password, like Cloudscape, do not set the alias in the component-managed authentication alias or container-managed authentication alias fields. Otherwise, a warning message is written in the system log to indicate that the user and password are not valid properties. This message is only a warning message because the data source is still created successfully.

7. Review the summary of your selections, and click **Finish**.
8. When the data source creation is complete, save the configuration and synchronize the changes with the nodes.
9. Test the new connection by selecting the new data source and clicking **Test connection**, as shown in Figure 9-16.

| Select | Name | JNDI name | Scope | Provider | Description | Category |
|---|-------------------------------|-----------------|----------------------|-------------------------------|--------------------------------|----------|
| You can administer the following resources: | | | | | | |
| <input checked="" type="checkbox"/> | orac-trade-ds | jdbc/trade-1-ds | Cluster=TradeCluster | Oracle JDBC Driver (XA) | New JDBC Datasource | |
| <input type="checkbox"/> | trade-app-ds | jdbc/trade-ds | Cluster=TradeCluster | DB2 Using IBM JCC Driver (XA) | DB2 JCC XA-capable data source | |

Figure 9-16 Test the connection

9.6 Connecting to an SQL Server database

This section illustrates a connection to a Microsoft SQL Server Enterprise Edition 2008 database.

Ensure that the implementation classes for the provider are available to the system. The class files need to be located on each system where the application servers will run.

In general, JDBC drivers are provided by the database vendor. Information about the location and features of the JDBC provider is provided by the database vendor, not the WebSphere documentation.

9.6.1 Creating the JDBC provider

Complete the following steps to create a JDBC provider:

1. In the administrative console, expand **Resources** → **JDBC** from the navigation tree.
2. Click **JDBC Providers**.
3. Select the scope. (Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.)
4. Click **New** to start the wizard to create a new JDBC provider.
5. In Step 1 of the wizard, define the type of provider that you will use. See Figure 9-17.

The screenshot shows the 'Create new JDBC provider' wizard. On the left, a navigation pane lists three steps: 'Step 1: Create new JDBC provider' (highlighted with a yellow arrow), 'Step 2: Enter database class path information', and 'Step 3: Summary'. The main content area is titled 'Create new JDBC provider' and includes a descriptive paragraph: 'Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.' Below this are several configuration fields: 'Scope' with a text input containing 'cells:sys2Cell01:clusters:TradeCluster'; '* Database type' with a dropdown menu set to 'SQL Server'; '* Provider type' with a dropdown menu set to 'Microsoft SQL Server JDBC Driver'; '* Implementation type' with a dropdown menu set to 'XA data source'; '* Name' with a text input containing 'Microsoft SQL Server JDBC Driver (XA)'; and 'Description' with a text area containing 'Microsoft SQL Server JDBC Driver (XA). This provider is configurable in version 6.1.0.15 and later nodes.'. At the bottom of the window are 'Next' and 'Cancel' buttons.

Figure 9-17 Define a new SQL Server JDBC provider: Window1

The database type is SQL Server, and the provider type is Microsoft SQL Server JDBC driver.

The options of implementation type are XA data source or connection pool data source. XA data source types support two-phase commit transactions.

Click **Next**.

6. Enter the directory location for the SQL Server JDBC drivers. See Figure 9-18.

Class path:
\${MICROSOFT_JDBC_DRIVER_PATH}/sqljdbc4.jar
Apply

Directory location for "sqljdbc4.jar" which is saved as WebSphere variable
\${MICROSOFT_JDBC_DRIVER_PATH}
C:\msql

Native library path
Directory location which is saved as WebSphere variable
\${MICROSOFT_JDBC_DRIVER_NATIVEPATH}
C:\msql\x86\x86

Figure 9-18 Define a new SQL Server JDBC provider - Window 2

If you defined either of the variables listed in the dialog and set their values, those values are displayed in the directory location and native library path fields.

Click **Next**.

7. Review the summary of the settings, and click **Finish**. The new JDBC provider displays in the list of providers.

8. Click the JDBC provider name to open the configuration window (Figure 9-19). Verify that the correct driver is used in the class path as advised by the vendor.

JDBC providers > Microsoft SQL Server JDBC Driver (XA)

Use this page to edit properties of a Java Database Connectivity (JDBC) provider. The JDBC provider object encapsulates the specific JDBC driver implementation class for access to the specific vendor database of your environment.

Configuration

General Properties

- * Scope: cells:sys2Cell01:clusters:TradeCluster
- * Name: Microsoft SQL Server JDBC Driver (XA)
- Description: Microsoft SQL Server JDBC Driver (XA). This provider is configurable in version 6.1.0.15 and later nodes.
- Class path: \${MICROSOFT_JDBC_DRIVER_PATH}/sqljdbc4.jar
- Native library path: \${MICROSOFT_JDBC_DRIVER_NATIVEPATH}
- Isolate this resource provider
- * Implementation class name: com.microsoft.sqlserver.jdbc.SQLServerXADataSource

Additional Properties

- Data sources
- Data sources (WebSphere Application Server V4)

Apply OK Reset Cancel

Figure 9-19 Configure the class path

9.6.2 Creating the data source

To create a data source:

1. Expand **Resources** → **JDBC** in the navigation tree, and click **Data sources**.
2. Select the scope. Although you can select **All scopes** to view all resources, you must select a specific scope to create a resource.

3. Click **New** to create a new data source and to start a wizard. See Figure 9-20.

Enter basic data source information

Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database.

Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification.

Scope
cells:sys2Cell01:clusters:TradeCluster

* Data source name
ms-trade

* JNDI name
jdbc/trade-2-ds

Next Cancel

Figure 9-20 Create a data source: Step 1

Enter a name for the new data source. This name is used for administrative purposes. Enter the JNDI name that will be used to access the data source, and click **Next**.

4. Select the Microsoft SQL Server JDBC driver, and click **Next**. See Figure 9-21.

Select JDBC provider

Specify a JDBC provider to support the datasource. If you choose to create a new JDBC provider, it will be created at the same scope as the datasource. If you are selecting an existing JDBC provider, only those providers at the current scope are available from the list.

Create new JDBC provider

Select an existing JDBC provider

Microsoft SQL Server JDBC Driver (XA)

Previous Next Cancel

Figure 9-21 Create a data source: Step 2

5. Enter the properties for the database. See Figure 9-22.

Enter database specific properties for the data source

Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.

| Name | Value |
|---------------|----------------------|
| Database name | TRADE |
| Port number | 1433 |
| Server name | sys2.itso.ral.ibmcom |

Use this data source in container managed persistence (CMP)

Previous Next Cancel

Figure 9-22 Create a data source: Step 3

Specify the following information:

- Enter the database name.
- Enter the port number on which the database server listens.
- Enter the host name of the SQL Server installation.

Click **Next**.

6. Select the authentication alias that provides the user ID and password that are required to access the database. This window allows you to select an already created authentication value or create a new one. If you select an existing authentication alias, continue with the next step. In Figure 9-23, the existing authentication alias, `sys2CellManager01/trade-app-alias`, is selected.

The page provides the following options:

- Authentication alias for XA Recovery: This field is used to specify the authentication alias that you must use during XA recovery processing. If this alias name is changed after a server failure, the subsequent XA recovery processing uses the original setting that was in effect before the failure. The database identity for the XA recovery authentication alias on a data source must have authorization to do XA recovery. If the resource adapter does not support XA transactions, this field does not display. The default value for this field is derived from the selected alias for application authentication, if one is specified.
- Component-managed authentication alias: This alias is used for database authentication at run time. If the database is not secured, setting database authentication is not required. This is not recommended for a production environment.
- Container-managed authentication alias: Specifies authentication data, which is a JAAS - J2C authentication data entry, for container-managed sign-on to the resource. Depending on the value that is selected for the Mapping-configuration alias setting, you can disable this setting.
- Mapping-configuration alias: Specifies the authentication alias for the Java Authentication and Authorization Service (JAAS) mapping configuration that is used by this connection factory. The DefaultPrincipalMapping JAAS configuration maps the authentication alias to the user ID and password.

Click **Next**.



Figure 9-23 Create a data source: Step 4

7. Review the summary of your selections, and click **Finish**.
8. When the data source creation is complete, save the configuration, and synchronize the changes with the nodes when using Network Deployment environment.
9. Test the new connection by selecting the new data source and clicking **Test connection**.

9.7 Configuring connection pooling properties

Performance of an application that connects to a database can be greatly affected by the availability of connections to the database and how those connections affect the performance of the database itself. There are no simple rules that tell you how to configure the connection pool properties. Your configuration is highly dependent on application, network, and database characteristics. You must coordinate the values that you specify in WebSphere closely with the database administrator.

Remember to include all resources in capacity planning. If 10 applications all connect to a database using separate connection pools of 10 maximum connections, this means that there is a theoretical possibility of 100 concurrent connections to the database. Make sure that the database server has sufficient memory and processing capacity to support this requirement.

Complete the following steps to access the connection pool properties:

1. Navigate to **Resources** → **JDBC** → **Data sources**, and click the data source name.
2. In the Additional Properties section, click **Connection pool properties**. The window shown in Figure 9-24 opens.

Figure 9-24 Data source connection pool properties

Specify the following information:

– Connection Timeout

Specify the interval, in seconds, after which a connection request times out and a `ConnectionWaitTimeoutException` is thrown. This action can occur when the pool is at its maximum (Max Connections) and all of the connections are in use by other applications for the duration of the wait. For example, if Connection Timeout is set to 300 and the maximum number of connections is reached, the Pool Manager waits for 300 seconds for an available physical connection. If a physical connection is not available within this time, the Pool Manager throws a `ConnectionWaitTimeoutException`.

Tip: If Connection Timeout is set to 0, the pool manager waits as long as necessary until a connection is allocated.

– Max Connections

Specify the maximum number of physical connections that can be created in this pool. These connections are the physical connections to the database. After this number is reached, no new physical connections are created and the requester waits until a physical connection that is currently in use is returned to the pool or a `ConnectionWaitTimeoutException` is thrown. For example, if Max Connections is set to 5, and there are five physical connections in use, the Pool Manager waits for the amount of time specified in Connection Timeout for a physical connection to become free. If, after that time, there are still no free connections, the Pool Manager throws a `ConnectionWaitTimeoutException` to the application.

– Min Connections

Specify the minimum number of physical connections to be maintained. Until this number is reached, the pool maintenance thread does not discard any physical connections. However, no attempt is made to bring the number of connections up to this number. For example, if Min Connections is set to 3, and one physical connection is created, that connection is not discarded by the Unused Timeout thread. By the same token, the thread does not automatically create two additional physical connections to reach the Min Connections setting.

Tip: Set Min Connections to zero (0) if the following conditions are true:

- ▶ You have a firewall between the application server and database server.
- ▶ Your systems are not busy 24x7.

– Reap Time

Specify the interval, in seconds, between runs of the pool maintenance thread. For example, if Reap Time is set to 60, the pool maintenance thread runs every 60 seconds. The Reap Time interval affects the accuracy of the Unused Timeout and Aged Timeout settings. The smaller the interval you set, the greater the accuracy. When the pool maintenance thread runs, it discards any connections that are unused for longer than the time value specified in Unused Timeout, until it reaches the number of connections specified in Min Connections. The pool maintenance thread also discards any connections that remain active longer than the time value specified in Aged Timeout.

Tip: If the pool maintenance thread is enabled, set the Reap Time value less than the values of Unused Timeout and Aged Timeout.

The Reap Time interval also affects performance. Smaller intervals mean that the pool maintenance thread runs more often and degrades performance.

– Unused Timeout

Specify the interval in seconds after which an unused or idle connection is discarded.

Tips:

- ▶ Set the Unused Timeout value higher than the Reap Timeout value for optimal performance. Unused physical connections are only discarded if the current number of connections not in use exceeds the Min Connections setting.
- ▶ Make sure that the database server's timeout for connections exceeds the Unused timeout property specified here. Long lived connections are normal and desirable for performance.

For example, if the unused timeout value is set to 120, and the pool maintenance thread is enabled (Reap Time is not 0), any physical connection that remains unused for two minutes is discarded. Note that accuracy of this timeout and performance are affected by the Reap Time value.

– Aged Timeout

Specify the interval in seconds before a physical connection is discarded, regardless of recent usage activity.

Setting Aged Timeout to 0 allows active physical connections to remain in the pool indefinitely. For example, if the Aged Timeout value is set to 1200 and the Reap Time value is not 0, any physical connection that remains in existence for 1200 seconds (20 minutes) is discarded from the pool. Note that accuracy of this timeout and performance are affected by the Reap Time value.

Tip: Set the Aged Timeout value higher than the Reap Timeout value for optimal performance.

– Purge Policy

Specify how to purge connections when a stale connection or fatal connection error is detected.

Valid values are EntirePool and FailingConnectionOnly. If you choose EntirePool, all physical connections in the pool are destroyed when a stale connection is detected. If you choose FailingConnectionOnly, the pool attempts to destroy only the stale connection. The other connections remain in the pool. Final destruction of connections that are in use at the time of the error might be delayed. However, those connections are never returned to the pool.

Tip: Many applications do not handle a StaleConnectionException in the code. Test and ensure that your applications can handle them.

Clicking the **Advanced connection pool properties** link allows you to modify the additional connection pool properties. These properties require advanced knowledge of how connection pooling works and how your system performs. For information about these settings, see the "Connection pool advanced settings" topic in the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.iseries.doc/ae/udat_conpooladv.html

9.8 WebSphere Application Server data source properties

You can set the properties that apply to the WebSphere Application Server connection, rather than to the database connection. To access the connection pool properties, navigate to **Resources** → **JDBC** → **Data sources**, and click the data source name. Click **WebSphere Application Server data source properties** in the Additional Properties section. See Figure 9-24 on page 373.

Clicking the link opens the window shown in Figure 9-25.

The screenshot shows a configuration window titled "General Properties" for a data source. It contains several sections:

- Statement cache size:** A text box with the value "10" and the unit "statements".
- Checkboxes:**
 - Enable multithreaded access detection
 - Enable database reauthentication
 - Enable JMS one-phase optimization support
 - Log missing transaction context
 - Non-transactional data source
- Error detection model:** Two radio buttons:
 - Use WebSphere Application Server exception checking model
 - Use WebSphere Application Server exception mapping model
- Connection validation properties:**
 - Validate new connections
 - Number of retries:
 - Retry interval: seconds
 - Validate existing pooled connections
 - Retry interval: seconds
- Validation options:**
 - Query:

Figure 9-25 WebSphere data source custom properties

Specify the following information:

► **Statement Cache Size**

Specify the number of prepared statements that are cached per connection. A prepared statement is a precompiled SQL statement that is stored in a prepared statement object. This object is used to execute the given SQL statement multiple times. The WebSphere Application Server data source optimizes the processing of prepared statements.

In general, the more statements your application has, the larger the cache must be. For example, if the application has five SQL statements, set the statement cache size to 5 so that each connection has five statements.

Statement Cache Size: This setting is vital to performance of the database and most likely requires tuning to suit the specific application. In general, the default is not high enough for best performance.

- ▶ Enable multi-threaded access detection

If you enable this feature, the application server detects the existence of access by multiple threads.

- ▶ Enable database reauthentication

Connection pool searches do not include the user name and password. If you enable this feature, a connection can still be retrieved from the pool, but you must extend the `DataStoreHelper` class to provide implementation of the `doConnectionSetupPerTransaction()` method where the reauthentication takes place.

Connection reauthentication can help improve performance by reducing the impact of opening and closing connections, particularly for applications that always request connections with different user names and passwords.

- ▶ Enable JMS one-phase optimization support

Activating this support enables the Java Message Service (JMS) to get optimized connections from the data source. Activating this support also prevents JDBC applications from obtaining connections from the data source. For further explanation of JMS one-phase support, refer to the article entitled “Sharing connections to benefit from one-phase commit optimization” at the following website.

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Fae%2Ftjm0280_.html

- ▶ Log missing transaction context

Specifies whether the container issues an entry to the activity log when an application obtains a connection without a transaction context.

- ▶ Non-transactional data source

Setting the flag to true causes the Application Server to never enlist the connections from the data source in global or local transactions. Applications must explicitly call `setAutoCommit(false)` on the connection if they want to start a local transaction on the connection, and they must commit or roll back the transaction that they started. This property is rarely set to true.

- ▶ Error detection model

The error detection model was expanded and the data source has a configuration option that you can use to select the exception mapping model or the exception checking model for error detection.

- ▶ Connection validation properties

There are two properties, and you can choose both. If you select the **Validate new connections** option, the connection manager attempts to communicate to the database using the new connection allocated, before returning the connection to the application for use. If you select this property, you can specify how often, in seconds (interval), the connection attempt will be retried, and how many attempts are made.

If you select the **Validate existing pooled connections** option, when the connection manager reuses an existing connection, it attempts to communicate to the database using that connection before returning it to the application for use. If you select this property, you can specify how often, in seconds (interval), the connection attempt will be retried. The pretest SQL string is sent to the database to test the connection.

Note: Connection validation by SQL query is deprecated in WebSphere Application Server V8.0.

- ▶ Advanced DB2 features:
 - Optimize for get/use/close/connection pattern with heterogeneous pooling

If you check this property, the heterogeneous pooling feature allows you to extend the data source definition.
 - DB2 automatic client reroute options

Client reroute for DB2 allows you to provide an alternate server location in case the connection to the database server fails. If you decide to use client reroute with the persistence option, the alternate server information persists across Java Virtual Machines (JVMs). In the event of an application server crash, the alternate server information is not lost when the application server is restored and attempts to connect to the database. You can specify the retry interval for client reroute, how often to retry, alternate server name or names for the DB2 server, port number, and JNDI name.

9.9 Shared and unshared connections

The WebSphere Application Server V8.5 connection manager supports both unshareable and shareable connections. It also provides local transaction containment (LTC) in an unspecified transaction context:

- ▶ An unshareable connection cannot be shared with other components in an application. The component using this connection has full control over it. Access to a resource marked as unshareable means that there is a one-to-one relationship between the connection handle that a component is using and the physical connection with which the handle is associated. This access implies that every call to the `getConnection()` method returns a connection handle solely for the requesting user.
- ▶ The use of a shareable connection means that, if conditions allow it, different `getConnection()` requests by an application actually receive a handle for the same physical connection to the resource. The physical connection is shared through multiple connection handles instead of retrieving a new physical connection from the connection pool for every `getConnection()` invocation.

More information about shared and unshared connections are in the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=/com.ibm.websphere.base.doc/ae/welcome_base.html

9.9.1 Factors that determine sharing

This section explains the factors that determine sharing, and the listing here is not an exhaustive one. The product might or might not share connections under different circumstances. The factors are:

- ▶ Each `getConnection()` request must have the same connection properties.
- ▶ Each `getConnection()` request must be made within the same sharing scope.

Connection sharing conditions are such that a connection can be shared only within a sharing scope. The most common sharing scope is a transaction scope, where multiple active connections share the same physical connection.

There are two transaction scopes in WebSphere Application Server:

- ▶ Global transaction
- ▶ Local transaction containment (LTC)

9.9.2 Configuring Shared and Unshared Connections

This section provides information about configuring shared and unshared connections:

- ▶ Resource Reference: The resource reference can be used to configure connection sharing for connection factory or data source. Example 9-1 shows how to configure shared connections for a data source using the resource reference.

Example 9-1 Shared connections for a data source

```
<resource-ref>
  <jndi-name>jdbc/Account</jndi-name>
  <authentication-alias>Alias1</authentication-alias>
  <interface>javax.sql.DataSource</interface>
  <authentication>Container</authentication>
  <sharing-scope>Shareable</sharing-scope>
  <id>resourceRef</id>
</resource-ref>
```

- ▶ Connection pool Custom properties: Custom properties, `defaultConnectionTypeOverride` and `globalConnectionTypeOverride` can be used to control connection sharing for a particular connection factory or data source:
 - `defaultConnectionTypeOverride`
Changes the default sharing value for a connection pool. The value configured through resource references takes precedence over this property.
 - `globalConnectionTypeOverride`:
The value takes precedence over all of the other connection sharing settings for connection factory or data source.

More information about Connection pool Custom properties is in the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.express.iseries.doc%2Fae%2Frdat_conpoolcustprops.html

9.10 Troubleshooting database access problems

This section describes ways to troubleshoot database access problems with WebSphere Application server. The following topics are covered:

- ▶ Enabling JDBC tracing for database problems
- ▶ Enabling ConnLeakLogic
- ▶ Dumping connection pool information using wsadmin
- ▶ Tool to debug Database Access problems

9.10.1 Enabling JDBC tracing for database problems

If the problem can be reproduced easily, enable a WebSphere Application Server trace. To enable the tracing:

1. Click **Troubleshooting** → **Logs and Trace** in the Application Server administrative console.
2. In the **Logging and Tracing**, select your **Server** → **Diagnostic Trace**.
3. Go to **Trace Output** → **File**. Accept the defaults.
4. Click **OK**.
5. Select **change the Log Detail Levels**.
6. Enter the following strings depending on your connection type:
 - Connecting to a database enter:
***info:**
WAS.j2c=all:
RRA=all:
Transaction=all
 - Connecting to an enterprise information system enter:
***info:**
WAS.j2c=all:
com.ibm.connector2.*all:
Transaction=all
 - Connecting to a messaging system enter:
***info:**
WAS.j2c=all:
Messaging=all:
JMSApi=all:
Transaction=all
7. Save your configuration and then click **OK**.
8. Restart the Application Server.
9. Reproduce the problem.
10. Send the resulting trace output file to IBM support for further analysis.

9.10.2 Enabling ConnLeakLogic

Connection pools get exhausted due to a variety of reasons, and ConnLeakLogic can be enabled to identify the application code holding the connections for long durations. It is recommended to enable a 'Runtime' trace instead of a 'Configuration' trace, which allows the trace to be disabled after the diagnostic data is retrieved from the server. To enable the ConnLeakLogic:

1. Start the application server.
2. Enable a Runtime trace immediately after starting the server:
 - a. Click **Troubleshooting** → **Logs and Trace** in the WebSphere Application Server administrative console.
 - b. Select the **link for your server**.
 - c. Click **Diagnostic Trace**.

- d. Click **Runtime tab**.
- e. Click **Change Log Level Details**.
- f. Click the **Runtime tab**.
- g. In the Trace Specification field, enter **ConnLeakLogic=a11**.
- h. Click **OK**.

Note: Ensure that you enabled the trace immediately *after* the server is started before any connections are obtained from the connection pool.

A trace.log is created that contains similar content as the SystemOut.log. It does not log ongoing messages, such as a WAS.j2c trace, and it causes only slight performance impact.

9.10.3 Dumping connection pool information using wsadmin

For collecting diagnostic data you can look at the SystemOut.log while the system is running. If you see the J2CA0045E error message in SystemOut.log, invoke **wsadmin** to dump the pool contents of the data source.

Use one of the following commands to dump the content of the connection pool:

- ▶ C:\IBM\WebSphere\bin>wsadmin -c “\$AdminControl invoke [\$AdminControl queryNames \”*:name=<INSERT DISPLAY NAME OF DATASOURCE HERE>,process=<SERVER NAME>,node=<NODE NAME>,j2eeType=JDBCDataSource,*\”] showPoolContents” -user <adminuserid> -password <adminpw>
- ▶ C:\IBM\WebSphere\bin>wsadmin>set ds [\$AdminControl queryNames “*:name=<INSERT DISPLAY NAME OF DATASOURCE HERE>,process=<SERVER NAME>, node=<NODE NAME>,j2eeType=JDBCDataSource,*”]
wsadmin>\$AdminControl invoke \$ds showPoolContents

9.10.4 Tool to debug Database Access problems

For debugging database access problems, you can use the IBM Database Connection Pool Analyzer. This tool finds JDBC connection leaks and helps to resolve JDBC connection pool problems. The tool performs the following functions:

- ▶ Analysis of JDBC data source
- ▶ Java stack trace view of getConnection method
- ▶ JDBC connection chart view
- ▶ Analysis of JDBC connection pool configuration

You can download the IBM Database Connection Pool Analyzer from the following website:

<https://www.ibm.com/developerworks/mydeveloperworks/groups/service/html/communityview?communityUid=f324412c-747c-42b9-8a70-a4d54e5f0e03>



Accessing EIS applications from WebSphere

The JEE Connector Architecture (JCA) defines a standard architecture for connecting the JEE platform to heterogeneous enterprise information systems (EIS). Some connectivity examples are ERP, mainframe transaction processing, database systems, and existing applications not written in the Java programming language. The architecture includes two parts: an EIS vendor-provided resource adapter and a JEE application server that supports the ability to plug in this resource adapter.

WebSphere Application Server V8.5 supports JCA versions 1.0, 1.5, and 1.6, including additional configurable features for JCA 1.5 and 1.6.

In this chapter, we provide information about the various considerations for accessing EIS applications from WebSphere.

We cover the following topics:

- ▶ JCA resource adapters
- ▶ Installing and configuring resource adapters
- ▶ Configuring J2C connection factories
- ▶ Resource authentication

10.1 JCA resource adapters

The Java EE Connector Architecture defines a set of secure, scalable, and transactional mechanisms that enable the communication of Java Enterprise Applications and EISs.

The JCA Resource Adapter is a system-level software driver supplied by EIS vendors or other third-party vendors. The adapter provides the following functionality:

- ▶ Connectivity between JEE components, such as an application server or an application client and an EIS
- ▶ Plugs into an application server
- ▶ Collaborates with the application server to provide important services, such as connection pooling, transaction, and security services

JCA 1.6 defines a broad set of system-level contracts between an application server and EIS. The resource adapter implements the EIS-side of these same system-level contracts:

- A *connection management contract* allows an application server pool to connect to an underlying EIS. This contract also allows application components to connect to an EIS. The contract leads to a scalable application environment that can support a large number of clients requiring access to EISs.
- A *transaction management contract* between the transaction manager and an EIS supports transactional access to EIS resource managers. This contract lets an application server use a transaction manager to manage transactions across multiple resource managers. This contract also supports transactions that are managed internally to an EIS resource manager without the necessity of involving an external transaction manager.
- A *security contract* enables secure access to an EIS. This contract provides support for a secure application environment, reducing security threats to the EIS, and protecting valuable information resources managed by the EIS.
- A *lifecycle management contract* allows an application server to manage the lifecycle of a resource adapter. This contract provides a mechanism for the application server to initialize a resource adapter instance during the adapter's deployment or at startup of the application server. The application server can also notify a resource adapter instance during its undeployment or a requested shutdown of the application server.
- A *work management contract* that allows a resource adapter to monitor endpoints, call application components, and other work. This can be achieved by submitting Work instances to an application server for execution. The application server creates threads to execute the submitted Work instances to allow better control of the application server's runtime environment.
- A *generic work context contract* enables a resource adapter to control the execution context of a Work instance that was submitted to the application server for execution.
- A *transaction inflow contract* allows a resource manager to propagate an imported transaction to an application server. This contract also allows a resource adapter to ensure that the ACID properties of the imported transaction are preserved.
- A *security work context* enables a resource adapter to establish security information while submitting a Work instance for execution to a WorkManager.
- A *message inflow contract* allows a resource adapter to asynchronously deliver messages to message endpoints residing in the application server.

- ▶ **Common Client Interface (CCI) for EIS access**
The CCI defines a standard client API through which a JEE component accesses the EIS. This simplifies writing code to connect to an EIS data store.
The resource adapter provides connectivity between the EIS, the application server, and the enterprise application through the CCI.
- ▶ **Implements the standard Service Provider Interface (SPI)**
The SPI integrates the transaction, security, and connection management facilities of an application server (JCA Connection Manager) with those of a transactional resource manager.

Multiple resource adapters (one resource adapter per type of EIS) can be plugged into an application server. This capability enables application components deployed on the application server to access the underlying EISs, as shown in Figure 10-1.

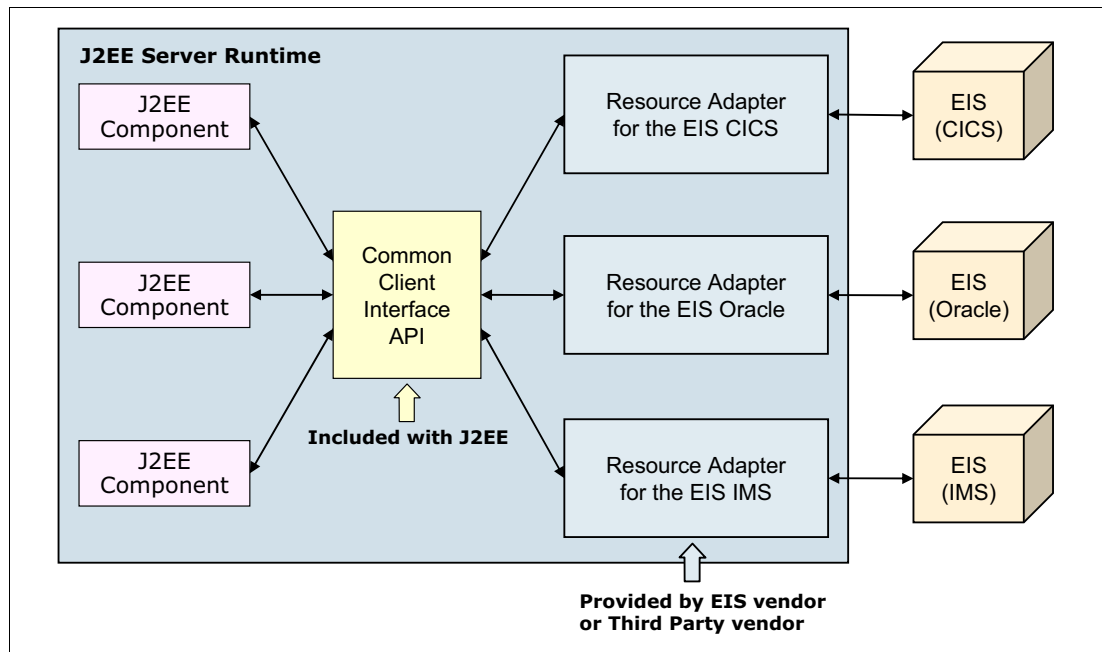


Figure 10-1 Common Client Interface API

10.2 WebSphere Application Server JCA support

In WebSphere Application Server, two types of objects are configured for JCA support:

- ▶ Resource adapters
- ▶ Connection factories

The application using the resource adapter requests a connection from the connection factory through a JNDI lookup. The application then uses the connection factory to get a connection to the underlying EIS.

The role of the WebSphere administrator is to:

- ▶ Install and define the resource adapter
- ▶ Define one or more connection factories associated with the resource adapter

10.2.1 Resource adapters

A WebSphere resource adapter administrative object represents the library that supplies implementation code for connecting applications to a specific EIS, such as IBM CICS® or SAP. Resource adapters are stored in a resource adapter archive (RAR) file, which is a Java archive (JAR) file used to package a resource adapter for the connector architecture. The file has a standard file extension of .rar.

A RAR file can contain the following elements:

- ▶ EIS-supplied resource adapter implementation code in the form of JAR files or other executables, such as DLLs
- ▶ Utility classes
- ▶ Static documents, such as HTML files for developer documentation, which are not used for run time
- ▶ J2C common client interfaces, such as `cci.jar`
- ▶ A deployment descriptor (`ra.xml`)

This deployment descriptor instructs the application server about how to use the resource adapter in an application server environment. The deployment descriptor contains information about the resource adapter, including security and transactional capabilities, and the `ManagedConnectionFactory` class name. In version 1.0 and 1.5, this deployment descriptor is *mandatory*.

Prior to JCA 1.6, metadata was specified only in the deployment descriptor, but now you can specify metadata using either a deployment descriptor or annotations. Metadata that is specified in annotations is merged into the deployment descriptor of a RAR module when it is updated. Annotation metadata is *not* merged if the module is marked `metadata-complete` in the deployment descriptor or if the module version is earlier than JCA 1.6.

The RAR file or JCA resource adapter is provided by your EIS vendor.

Registering the resource adapter with the high-availability manager specifies that the high-availability (HA) manager will manage the lifecycle of a JCA 1.5 or later resource adapter in a cluster. This manager ensures that applications using resource adapters for inbound communication remain highly available. Appropriate use of the HA capability options enable you to set up an environment that can implement failover for inbound activity when a server goes down.

WebSphere provides three JCA resource adapters:

- ▶ The WebSphere Relational Resource Adapter: Used to connect to relational databases using JDBC. The WebSphere Relational Resource Adapter is installed and runs as part of WebSphere Application Server and needs no further administration.
- ▶ The Resource Adapter for Java Message Service (JMS): Used by applications that perform JMS or JCA messaging with the default messaging provider
- ▶ The WebSphere MQ resource adapter: Used by applications that perform JMS or JCA messaging with the WebSphere MQ messaging provider.

10.2.2 Connection factories

The WebSphere connection factory administrative object represents the configuration of a specific connection to the EIS supported by the resource adapter. The connection factory can be thought of as a list holder for connection configuration properties.

Application components, such as CMP enterprise beans, have `cmpConnectionFactory` descriptors that refer to a specific connection factory, not to the resource adapter.

10.3 Installing and configuring resource adapters

To use a resource adapter, you must install the resource adapter code and create connection factories that use the adapter. The resource adapter configuration is stored in the `resources.xml` file.

There are two ways to make a resource adapter (.rar file) available to applications. One way is to install the adapter into WebSphere Application Server. The other way is to install the adapter in the application (embedded adapter). For example, Rational Application Developer embeds resource adapters when you create a JCA application. This chapter describes installing the adapter into WebSphere Application Server, which can be useful if different applications use the same resource adapter.

To install an adapter:

1. From the administrative console, select **Resources** → **Resource Adapters** → **Resource adapters**, and select a scope (Figure 10-2 on page 388).

Note: You can see all of the WebSphere built-in resources by selecting the **Show built-in resources** preference, also shown in Figure 10-2.

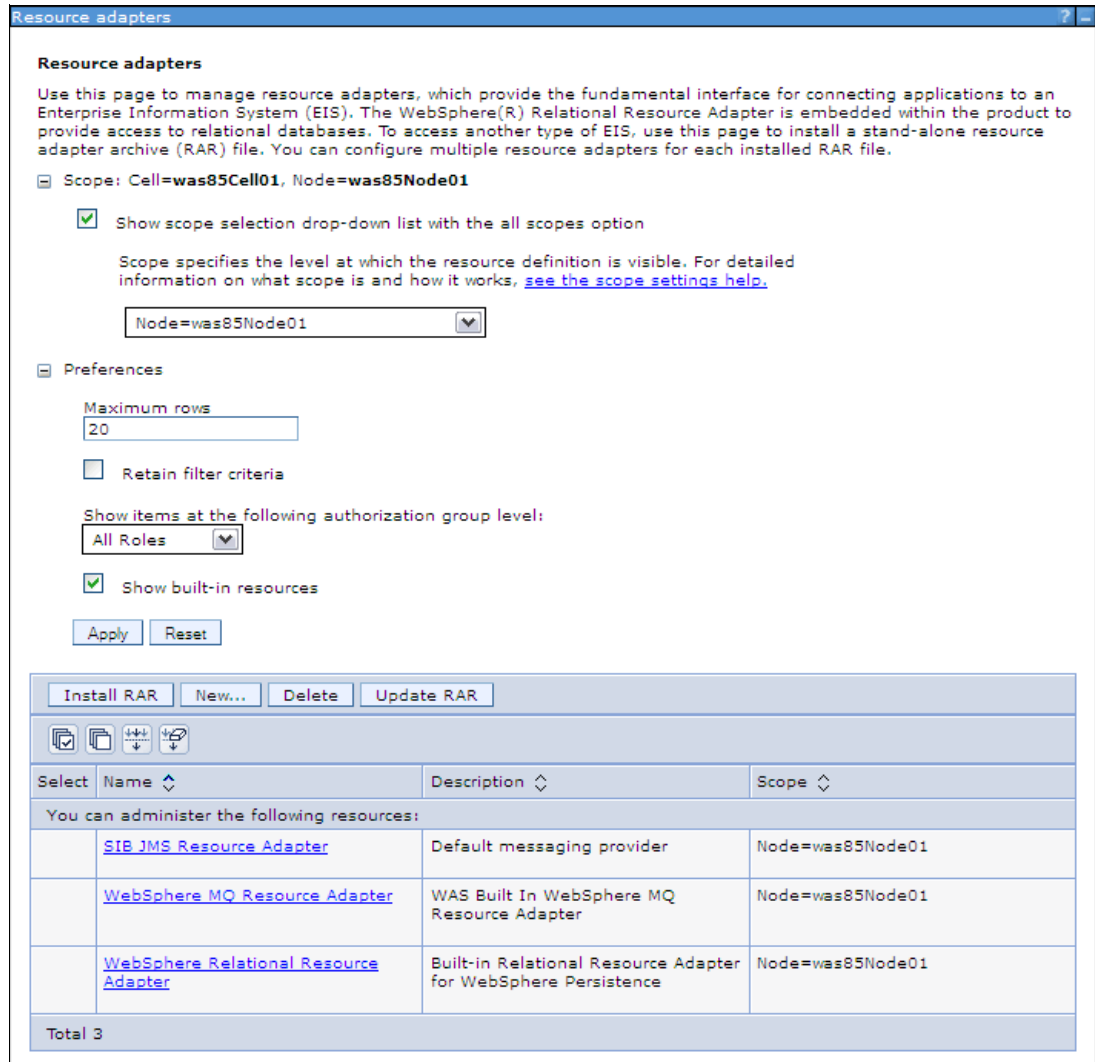


Figure 10-2 JCA resource adapters

2. Click **Install RAR** to install a new resource adapter.
3. Enter the path to the .rar file supplied by your EIS vendor. In this example, we use a JCA adapter provided by IBM. It can reside locally, on the same machine as the browser, or on any of the nodes in your cell. See Figure 10-3 on page 389.

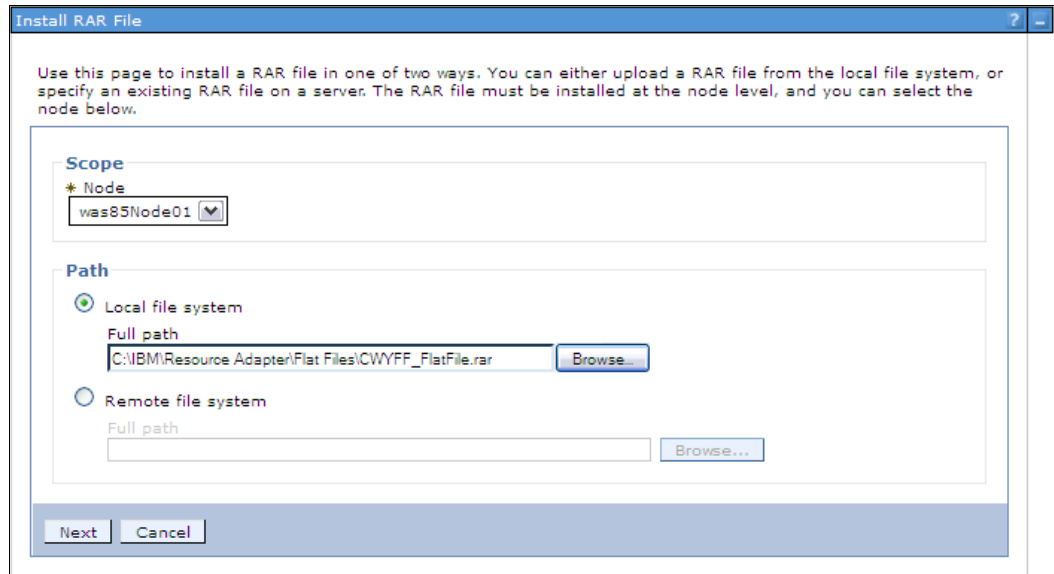


Figure 10-3 RAR file location

Select the node where you want to install the .rar file. You have to install the file on each node separately.

4. Click **Next**.

5. The Configuration window for the resource adapter selected opens containing information gathered from the deployment descriptor, as shown in Figure 10-4.

The screenshot shows a configuration window titled "General Properties". It contains the following fields and controls:

- Scope:** A text field containing "cells:was85Cell01:nodes:was85Node01".
- Name:** A text field containing "IBM WebSphere Adapter for Flat Files".
- Description:** A large empty text area.
- Archive path:** A text field containing "\${CONNECTOR_INSTALL_ROOT}".
- Class path:** A large empty text area.
- Native library path:** A large empty text area.
- Isolate this resource provider:** An unchecked checkbox.
- Buttons:** "OK", "Reset", and "Cancel" buttons at the bottom.

Figure 10-4 JCA resource adapter properties

In this example, you do not have to configure any properties. The defaults, combined with the information supplied in the RAR file, provide all of the information needed. However, you have the option of configuring the following properties:

- Name: Create an administrative name for the resource adapter.
- Description: Create an optional description of the resource adapter for your administrative records.
- Archive path: This field is the path where the RAR file is installed. If this property is not specified, the archive is extracted to the absolute path represented by the `${CONNECTOR_INSTALL_ROOT}` variable. The default is `profile_root/installedConnectors/adapter_name.rar` .
- Class path: A list of paths or JAR file names that together form the location for the resource adapter classes. The resource adapter code base itself, the RAR file, is automatically added to the class path.
- Native path: This is a list of paths that together form the location for the resource adapter native libraries (.dll and .so files).

6. Click **OK**.
7. Save the configuration and synchronize the nodes.

10.4 Configuring J2C connection factories

Terms: The terms J2C and JCA both refer to JEE Connector Architecture, and they are used here interchangeably.

A J2C connection factory represents a set of connection configuration values. Application components, such as EJBs, have <resource-ref> descriptors that refer to the connection factory, not the resource adapter. The connection factory holds the list of connection configuration properties. In addition to the arbitrary set of configuration properties defined by the vendor of the resource adapter, there are several standard configuration properties that apply to the connection factory. These standard properties are used by the connection pool manager in the application server run time and are not used by the vendor-supplied resource adapter code.

To create a J2C connection factory:

1. Click **Resources** → **Resource Adapters** → **J2C connection factories**. You can see a list of J2C connection factories at the selected scope.
2. Click **New** to create a new connection factory or select an existing one to modify the connection factory properties. The J2C Connection Factory Configuration window opens, as shown in Figure 10-5 on page 392.

General Properties

* Scope

Provider

IBM WebSphere Adapter for Flat Files

* Name

JNDI name

Description

* Connection factory interface

Category

Security settings

Select the authentication values for this resource.

Component-managed authentication alias

Mapping-configuration alias

Container-managed authentication alias

Authentication preference

The additional properties will not be available until the general properties for this item are applied or saved.

Additional Properties

- Connection pool properties
- Advanced connection factory properties
- Custom properties

Related Items

- JAAS - J2C authentication data

Figure 10-5 J2C Connection factory properties

Enter the following information:

- Name: Enter an administrative name for the J2C connection factory.
- JNDI name: This field is the connection factory name to be registered in the application server's name space, including any naming sub context.

When installing an application that contains modules with J2C resource references, the resources defined by the deployment descriptor of the module must be bound to the JNDI name of the resource.

As a convention, use the value of the Name property prefixed with `eis/`, for example, `eis/<ConnectionFactoryName>`.

- Description: This is an optional description of the J2C connection factory for your administrative records.
- Connection factory interface: This field is the name of the connection factory interfaces supported by the resource adapter.
- Category: Specify a category that you can use to classify or group the connection factory.
- Security settings: You have multiple options when securing access to the J2C resource. Although component-managed might be faster in some instances, it is not the best solution for security. Container-managed authentication is the preferred method.

For more information, see 10.5, “Resource authentication” on page 393.

3. Click **Apply**. The links under the Additional Properties section for connection pool, advanced connection factory, and custom properties become active.

The connection pool properties can affect performance of your application. Monitor and adjust these settings to maximize performance.

The advanced connection factory properties are shown in Figure 10-6.

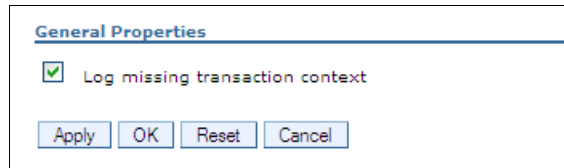


Figure 10-6 Advanced connection factory properties

The JEE programming model indicates that connections must always have a transaction context. However, some applications do not have a context associated with them. The Log missing transaction context option tells the container to log the fact that there is a missing transaction context in the activity log when the connection is obtained.

10.5 Resource authentication

Resources often require you to perform authentication and authorization before an application can access them. You can configure the settings to determine how this is done in a number of ways. This section provides information about the configuration settings and how to use them.

The party responsible for the authentication and authorization is determined by the `res-auth` setting found in the web and EJB deployment descriptors. There are two possible settings:

- ▶ `res-auth=Container`: WebSphere is responsible.
The authentication data is supplied by the application server.
- ▶ `res-auth=Application`: The application or component is responsible.

The authentication data is taken from the following elements, in the following order:

- The user ID and password that are passed to the getConnection method. (This is not a best practice. This implies that the user ID and password are coded in the application).
- The component-managed authentication alias in the connection factory or the data source.
- The custom properties for the user name and password in a data source resource. The names of the custom properties are specific to the JDBC provider associated with the data source. Consult the documentation for the JDBC driver for the correct property names. Configure these properties by creating two properly named custom properties on the data source and setting the value appropriately.

These settings can be configured during application assembly using Rational Application Developer in the EJB or web deployment descriptor. They can also be set or overridden during application installation. See Table 10-1.

Table 10-1 Authentication settings

| Authentication type | Setting at assembly Authorization type | Setting during installation Resource authorization |
|--|---|---|
| WebSphere managed: res-auth=Container | Container | Container |
| Application (component) managed: res-auth=Application | Per_Connection_Factory | Per application |

10.5.1 Container-managed authentication

Container-managed authentication removes the requirement that the component programmatically supply the credentials for accessing the resource. Instead of calling the getConnection() method with a ConnectionSpec object, getConnection() is called with no arguments. The authentication credentials are then supplied by the web container, application container, or the EJB container, depending from where the resource is accessed. WebSphere Application Server supports the Java Authentication and Authorization Service (JAAS) specification. This support means the credentials can be mapped from any of the configured JAAS authentication login modules, including any custom JAAS authentication login module.

The default selection for the JAAS application login module is located in the mapping-configuration-alias field of the J2C connection factory and is called DefaultPrincipleMapping. DefaultPrincipleMapping, maps the user ID and password using a pre-configured J2C authentication alias.

Container-managed authentication is the preferred method.

10.5.2 Component-managed authentication

In the case of component-managed authentication, the application component accessing the resource or adapter is responsible for programmatically supplying the credentials. WebSphere can also supply a default component-managed authentication alias if available. After obtaining the connection factory for the resource from JNDI, the application component creates a connection to the resource using the create method on the connection factory supplying the credentials. If no credentials are supplied when creating a connection and a component-managed authentication alias is specified on the J2C connection factory, the credentials from the authentication alias are used.

Assuming that the credentials are valid, future requests using the same connection will use the same credentials.

The application follows these basic steps.

1. Get the initial JNDI context.
2. Look up the connection factory for the resource adapter.
3. Create a ConnectionSpec object holding credentials.
4. Obtain a connection object from the connection factory by supplying the ConnectionSpec object.



Configuring messaging providers

Asynchronous messaging support provides applications with the ability to create, send, receive, and read asynchronous requests as messages. WebSphere Application Server V8.5 supports asynchronous messaging based on the Java Message Service (JMS) and the Java EE Connector Architecture (JCA) specifications.

WebSphere Application Server includes a default messaging provider and support for WebSphere MQ and third-party messaging providers. In this chapter, we introduce how to configure these messaging providers step-by-step.

This chapter contains the following topics:

- ▶ Messaging providers introduction
- ▶ Configuring resources for the default messaging provider
- ▶ Configuring resources for the WebSphere MQ messaging provider
- ▶ Configuring resources for third-party messaging providers

11.1 Messaging providers introduction

WebSphere Application Server V8.5 supports a variety of JMS providers, including:

- ▶ The WebSphere Application Server default messaging provider, which is a JCA resource adapter implementation that is fully integrated into WebSphere. The default messaging provider uses a service integration bus as the messaging system.
- ▶ The WebSphere MQ messaging provider, which uses a WebSphere MQ installation as the provider.
- ▶ Third-party messaging providers that implement either a JCA Version 1.5 or 1.6 resource adapter or JMS Version 1.1 unified connection factories.

To work with message-driven beans, third-party non-JCA messaging providers must include Application Server Facility (ASF), which is an optional feature that is part of the JMS Version 1.1 specification.

For more information about basic messaging concepts, refer to *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022.

11.2 Configuring resources for the default messaging provider

To enable a messaging application to use the WebSphere Application Server default messaging provider, configure the following resources:

- ▶ A connection factory
- ▶ A JMS destination
- ▶ A JMS activation specification

The sections that follow explain how to configure these resources step-by-step.

11.2.1 Configuring JMS connection factories

To configure a JMS connection factory for the default messaging provider:

1. If you did not create a service integration bus, create it now. (Refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770 for details.) In this example, we use a bus called *SampleBus*.

2. Click **Resources** → **JMS** → **Connection factories**. In the Connection factories window, shown in Figure 11-1, choose the scope (the **Cell scope** in this sample) and then click **New**.

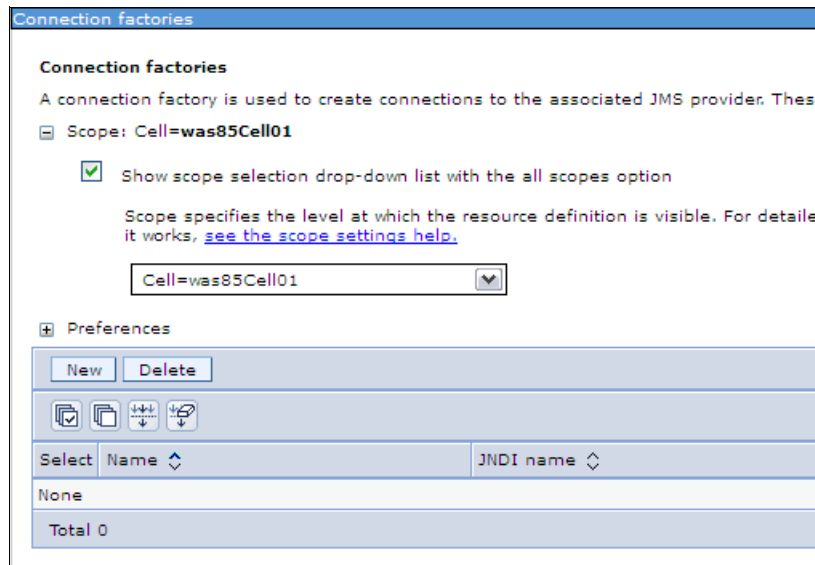


Figure 11-1 Creating a new connection factory

3. Select the **Default messaging provider** option and then click **OK**.
4. Enter the following basic properties:
 - Name
 - JNDI name
 - Bus names

In this example (refer to Figure 11-2 on page 400), SampleJMSCF connects to SampleBus. The JMS application accesses the factory using the JNDI name `jms/SampleJMSCF`.

General Properties

Administration

Scope
Cell=was85Cell01

Provider
Default messaging provider

* Name
SampleJMSCF

* JNDI name
jms/SampleJMSCF

Description

Category

Connection

* Bus name
SampleBus

Figure 11-2 Default messaging provider JMS connection factory properties

5. Click **OK** to create the new connection factory, which is then listed among the resources to be administered (see Figure 11-3).

| Select | Name | JNDI name | Provider |
|--------------------------|-------------|-----------------|----------------------------|
| <input type="checkbox"/> | SampleJMSCF | jms/SampleJMSCF | Default messaging provider |

Total 1

Figure 11-3 Default messaging provider JMS connection factories list

6. Save the changes to the configuration.

11.2.2 Configuring JMS destinations

You can configure both queue and topic destinations for the default messaging provider. In this section, we describe how to create and configure a JMS queue. For information about how to create and configure JMS topics, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

To configure JMS destinations:

1. Create a new destination in the service integration bus. Click **Service integration** → **Buses**. Click **SampleBus**, navigate to **Destination resources** → **Destination**, and click **New** (see Figure 11-4 on page 401).



Figure 11-4 Creating a new destination

2. Select **Queue** as the destination type and then click **Next**.
3. Enter the identifier for the queue destination and then click **Next**, as shown in Figure 11-5.

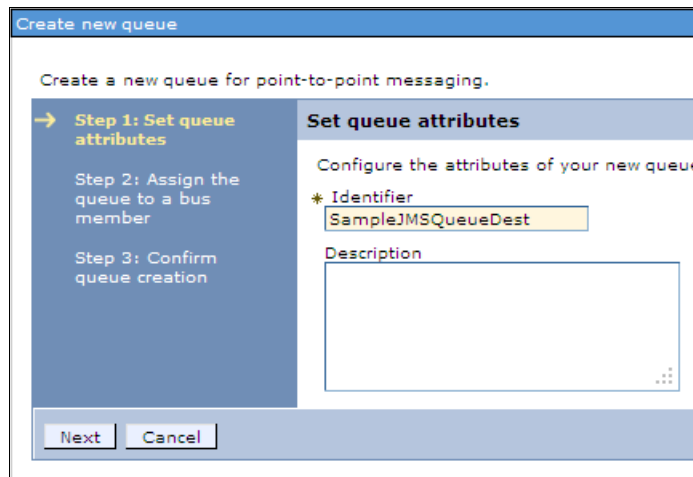


Figure 11-5 Configuring the destination

4. Accept the default values for the other options by clicking **Next** and then **Finish**. A new queue named *SampleJMSQueueDest* is created.
5. Save the changes to the configuration.

11.2.3 Configuring JMS queues

A JMS queue is an administrative object that contains the name of a queue destination on a service integration bus. Applications find the JMS queue by looking up the JMS queue name in the JNDI namespace.

To configure a JMS Queue:

1. Click **Resources** → **JMS** → **Queues**. In the Queues window, choose a **Scope** (this example uses the **Cell scope**) and then click **New**.

2. Select the **Default messaging provider** option and then click **OK**.
3. Enter the following basic properties:
 - Name
 - JNDI name
 - Bus name
 - Queue name

In this example (refer to Figure 11-6), the queue name is SampleJMSQueue.

The screenshot shows a 'General Properties' dialog box with two main sections: 'Administration' and 'Connection'.
 In the 'Administration' section:
 - 'Scope' is a text field containing 'Cell=was85Cell01'.
 - 'Provider' is a text field containing 'Default messaging provider'.
 - '* Name' is a text field containing 'SampleJMSQueue'.
 - '* JNDI name' is a text field containing 'jms/SampleJMSQueue'.
 - 'Description' is a large empty text area.
 In the 'Connection' section:
 - 'Bus name' is a dropdown menu showing 'SampleBus'.
 - '* Queue name' is a dropdown menu showing 'SampleJMSQueueDest'.

Figure 11-6 Default messaging provider queue properties

4. Click **OK**. The new queue is created.
5. Save the changes to the configuration.

11.2.4 Configuring JMS activation specifications

Activation specifications are used to configure inbound message delivery to message-driven beans (MDBs) running inside WebSphere Application Server. A JMS activation specification is associated with a MDB during application deployment.

To configure a JMS activation specification for the default messaging provider, complete the following steps:

1. Click **Resources** → **JMS** → **Activation specifications**. In the Activation specifications window, choose **Scope** (use **Cell scope** in this sample) and then click **New**.
2. Select the **Default messaging provider** option and then click **OK**.
3. Enter the following basic properties:
 - Name
 - JNDI name
 - Destination type

- Destination JNDI name
- Bus name

In this example, the activation specification name is `SampleJMSAS`, as shown in Figure 11-7.

The screenshot shows a configuration window titled "General Properties" with two main sections: "Administration" and "Destination".

- Administration:**
 - Scope: `Cell=was85Cell01`
 - Provider: `Default messaging provider`
 - * Name: `SampleJMSAS`
 - * JNDI name: `jms/SampleJMSAS`
 - Description: (Empty text area)
- Destination:**
 - * Destination type: `Queue` (dropdown menu)
 - * Destination JNDI name: `jms/SampleJMSQueue`
 - Message selector: (Empty text field)
 - * Bus name: `SampleBus` (dropdown menu)

Figure 11-7 Default messaging provider activation specification properties

You can configure other properties for this activation specification if needed. For detailed information about other properties, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

4. Click **OK**. The new activation specification is created.
5. Save the changes to the configuration.

11.3 Configuring resources for the WebSphere MQ messaging provider

In this section, we explain how to configure the resources for the WebSphere MQ messaging provider to communicate with WebSphere MQ using bindings or client connections.

11.3.1 Configuring WebSphere MQ messaging provider connection factories

To configure a JMS connection factory for the WebSphere MQ messaging provider, complete the following steps:

1. Click **Resources** → **JMS** → **Connection factories**. In the Connection factories window, shown in Figure 11-1 on page 399, select the **Scope** (this sample uses **Cell scope**) and then click **New**.
2. Select the **WebSphere MQ messaging provider** option and then click **OK**.
3. Enter the name for the connection factory and the JNDI name that binds it to the namespace and then click **Next** (Figure 11-8).

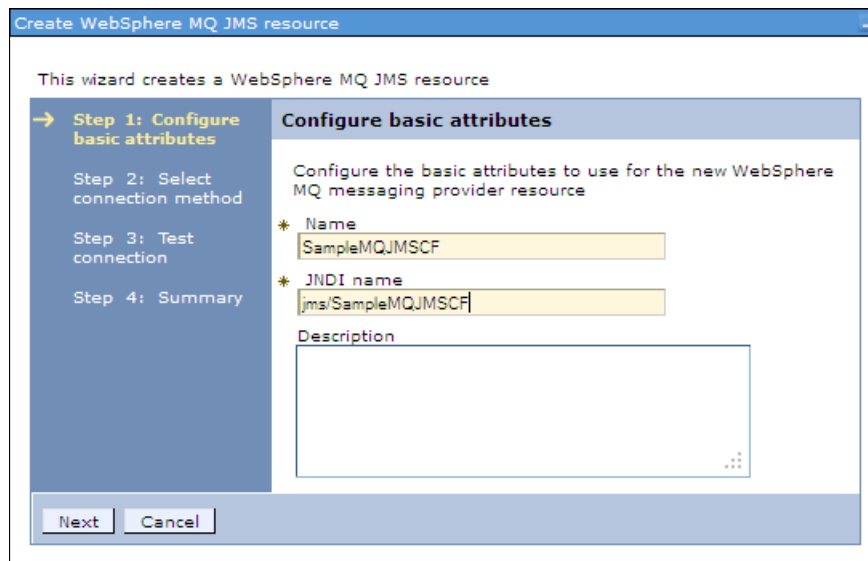


Figure 11-8 WebSphere MQ messaging provider connection factory, basic attributes

4. Select the **Enter all the required information into this wizard** option and then click **Next**.

For information about how to use a client channel definition table, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

5. Specify the queue manager or queue sharing group name and then click **Next** (Figure 11-9 on page 405).

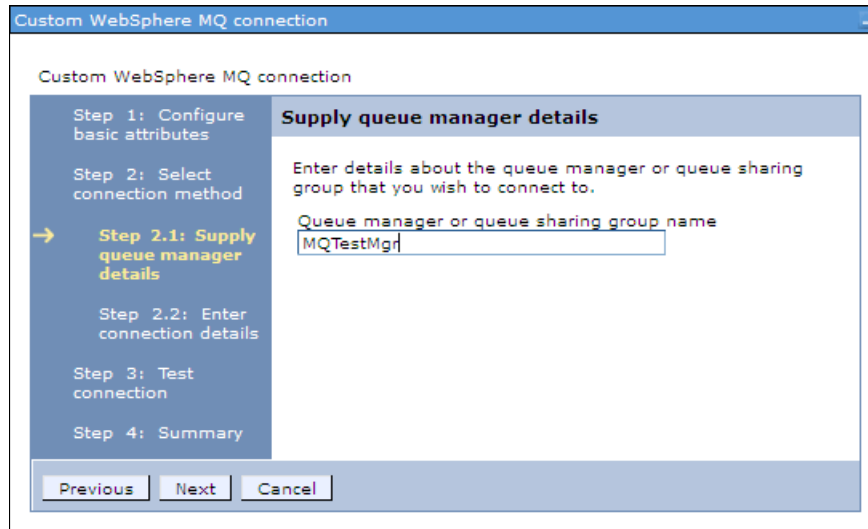


Figure 11-9 WebSphere MQ messaging provider connection factory, queue manager name

6. Choose the transport type, specify the host name and port properties of the WebSphere MQ queue manager, and click **Next**, as shown in Figure 11-10. The port must match the listener port that is defined for the queue manager, for example, 1424.

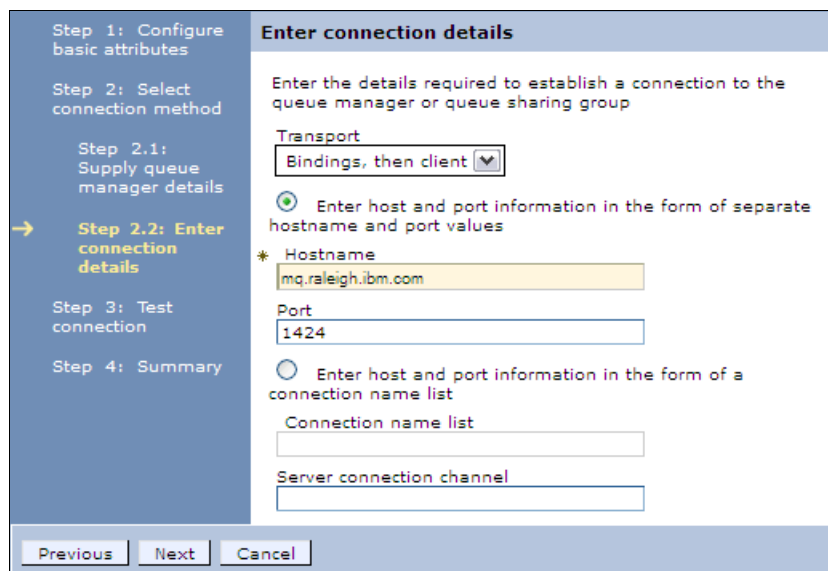


Figure 11-10 WebSphere MQ messaging provider connection factory, connection details

Important: WebSphere Application Server V8.5 supports connections to multi-instance WebSphere MQ queue managers. To do this, you provide host and port information in the form of a *connection name list*, which a connection factory or activation specification uses to connect to a multi-instance queue manager. If you are using multi-instance WebSphere MQ queue managers, select **Enter host and port information in the form of a connection name list**. For details about configuring connections to multi-instance WebSphere MQ queue managers, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.express.doc/ae/umj_pasm.html

7. Click **Test connection** to verify that you can connect to the WebSphere MQ queue manager and then click **Next**.
8. In the Summary window, review the summary and then click **Finish**. The new connection factory is created.
9. Save the changes to the configuration.

WebSphere Application Server V8.5 exposes the client reconnection properties for connection factories. You can use this property to specify whether a client mode connection reconnects automatically, in the event of a communications or queue manager failure, and also to specify a timeout value for reconnection attempts.

You can find this property on the **Advanced properties** window of a defined WebSphere MQ connection factory, as shown in Figure 11-11.



Figure 11-11 WebSphere MQ messaging provider connection factory, advanced properties

For information about other advanced properties, refer to the WebSphere Application Server information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.base.doc/ae/umj_pjcfm_advprops.html

11.3.2 Configuring WebSphere MQ messaging provider destinations

You can configure both the queue and topic destinations for the WebSphere MQ messaging provider. In this section, we introduce how to create and configure the WebSphere MQ queue destination. For information about creating and configuring the WebSphere MQ topic destination, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

To configure a queue destination for the WebSphere MQ messaging provider:

1. Click **Resources** → **JMS** → **Queues**. In the Queues window, select the **Scope** (this example uses **Cell scope**), and click **New**.
2. Select the **WebSphere MQ messaging provider** option and then click **OK**.
3. Enter the following basic properties:
 - Name
 - JNDI name
 - Queue name

In this example, the queue name is SampleMQQueue (see Figure 11-12). This name must match the queue name that is defined in the WebSphere MQ queue manager to which you are connecting.

The screenshot shows a 'General Properties' dialog box with two main sections: 'Administration' and 'WebSphere MQ Queue'.
In the 'Administration' section:
- 'Scope' is 'Cell=was85Cell01'.
- 'Provider' is 'WebSphere MQ messaging provider'.
- '* Name' is 'SampleMQQueue'.
- '* JNDI name' is 'jms/SampleMQQueue'.
- 'Description' is an empty text area.
In the 'WebSphere MQ Queue' section:
- '* Queue name' is 'SampleMQQueue'.
- 'Queue manager or Queue sharing group name' is an empty text field.
At the bottom are buttons for 'Apply', 'OK', 'Reset', and 'Cancel'.

Figure 11-12 WebSphere MQ messaging provider queue configuration

4. Click **OK**. The new queue is now created.
5. Save the changes to the configuration.

WebSphere Application Server V8.5 provides several advanced properties for WebSphere MQ queue or topic destinations, including:

- ▶ Append RFH version 2 headers to messages sent to this destination

This applies to reply messages sent to the reply-to queue obtained from a message. Select this option to append an RFH version 2 header to the reply message regardless of whether the original message had an RFH version 2 header.

Set this property in the **Advanced properties** window of a defined WebSphere MQ queue, as shown in Figure 11-13 on page 408.

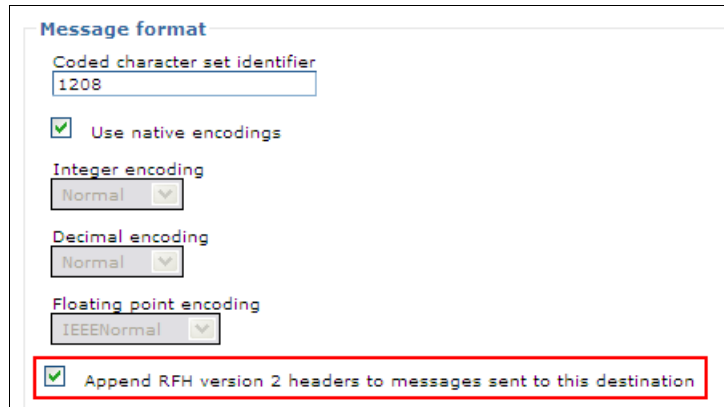


Figure 11-13 WebSphere MQ messaging provider queue, RFH property

► Reply to style

This property specifies how the JMSReplyTo header field in a WebSphere MQ messaging provider message is generated.

Set this property in the Advanced properties window of a defined WebSphere MQ queue, as shown in Figure 11-14.

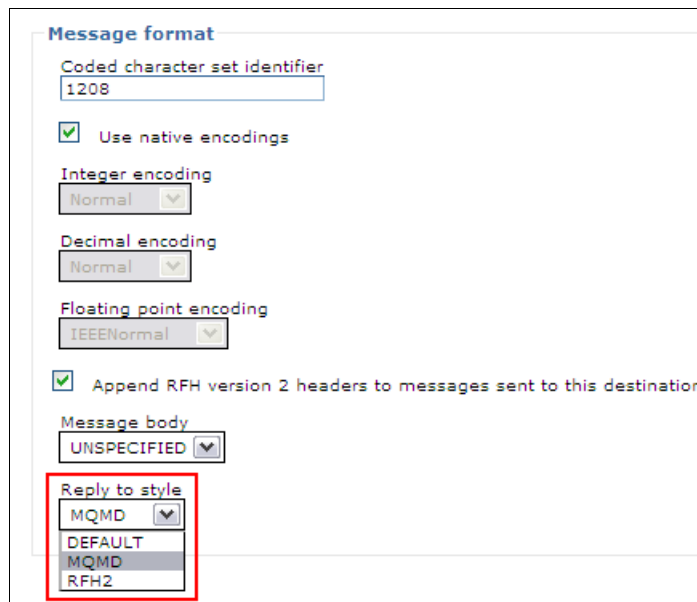


Figure 11-14 WebSphere MQ messaging provider queue, reply to style property

► Message descriptor

This set of properties specifies the following information:

- Whether an application can read or write the values of MQMD fields from JMS messages that were sent or received using the WebSphere MQ messaging provider.
- Which message context options are used when sending messages to a destination.

Set these properties in the Advanced properties window of a defined WebSphere MQ queue, as shown in Figure 11-15 on page 409.



Figure 11-15 WebSphere MQ messaging provider queue, message descriptor properties

11.3.3 Configuring WebSphere MQ messaging provider activation specifications

To configure an activation specification for the WebSphere MQ messaging provider:

1. Click **Resources** → **JMS** → **Activation specifications**. In the Activation specifications window, select the **Scope** (use **Cell scope** in this sample) and then click **New**.
2. Select the **WebSphere MQ messaging provider** option and then click **OK**.
3. Enter the following basic properties, as shown in Figure 11-16:
 - Name
 - JNDI name

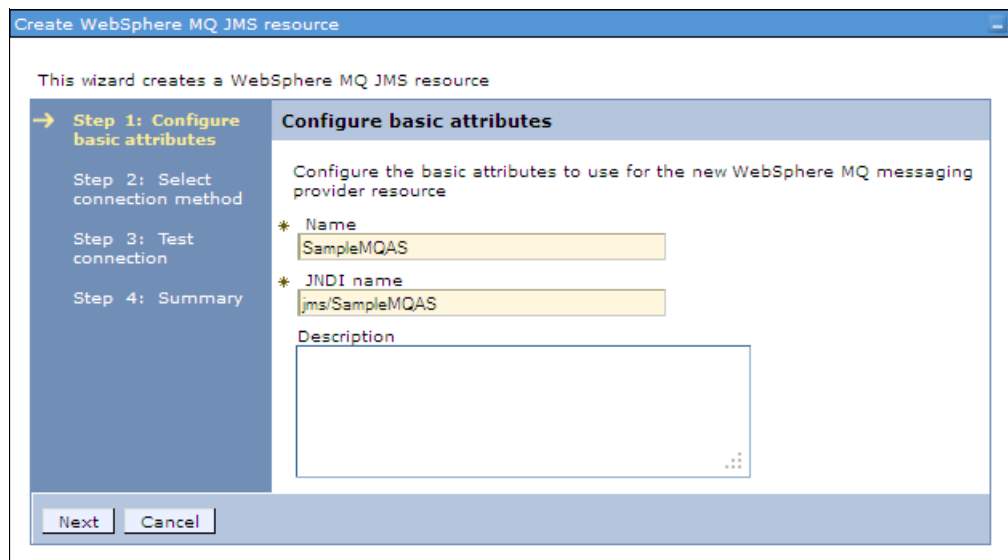


Figure 11-16 WebSphere MQ messaging provider activation specification, basic attributes

4. Click **Next** and then enter the information of the destination.

In this example, the destination JNDI name is `jms/SampleMQQueue` (see Figure 11-17 on page 410). This name corresponds to the queue destination JNDI name that we created in section 11.3.2, “Configuring WebSphere MQ messaging provider destinations” on page 406.

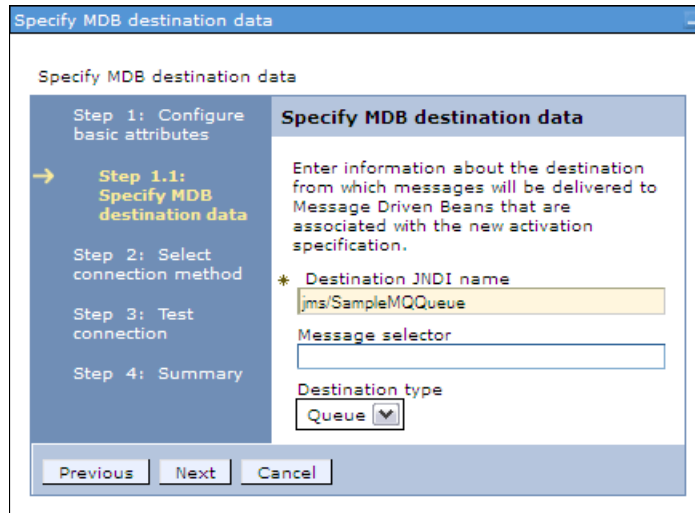


Figure 11-17 WebSphere MQ messaging provider activation specification, destination data

5. Click **Next**. On the next page, select **Enter all the required information into this wizard** and then click **Next** (refer to Figure 11-18).

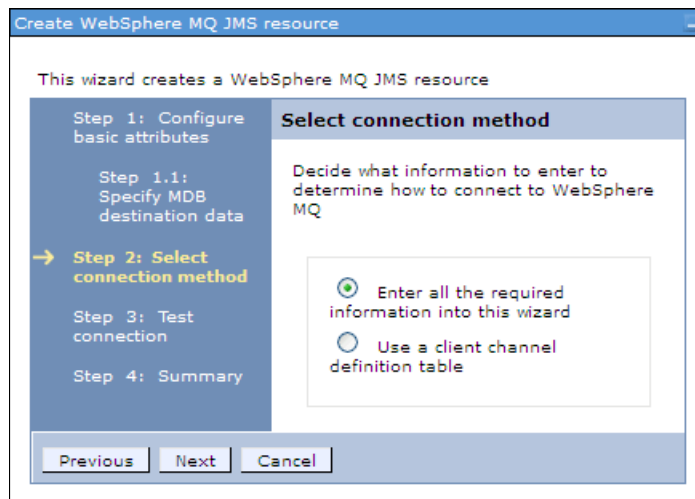


Figure 11-18 WebSphere MQ messaging provider activation specification, connection method

6. Enter the name of the queue manager or queue sharing group and then click **Next**, as shown in Figure 11-19 on page 411.

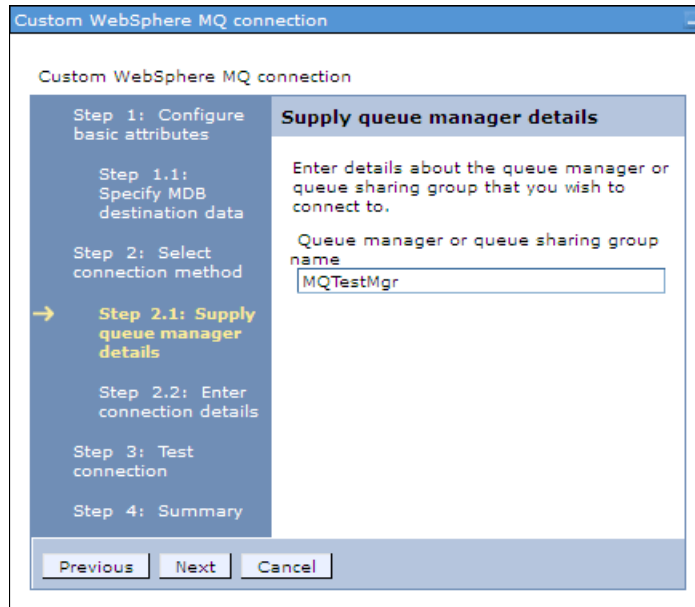


Figure 11-19 WebSphere MQ messaging provider activation specification, queue manager

7. Select the transport method. This is the manner in which a connection to WebSphere MQ is established. For details about how to choose the transport method, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.express.doc/ae/umj_pasm.html

8. Enter the host and port information of WebSphere MQ and then click **Next** (refer to Figure 11-20).

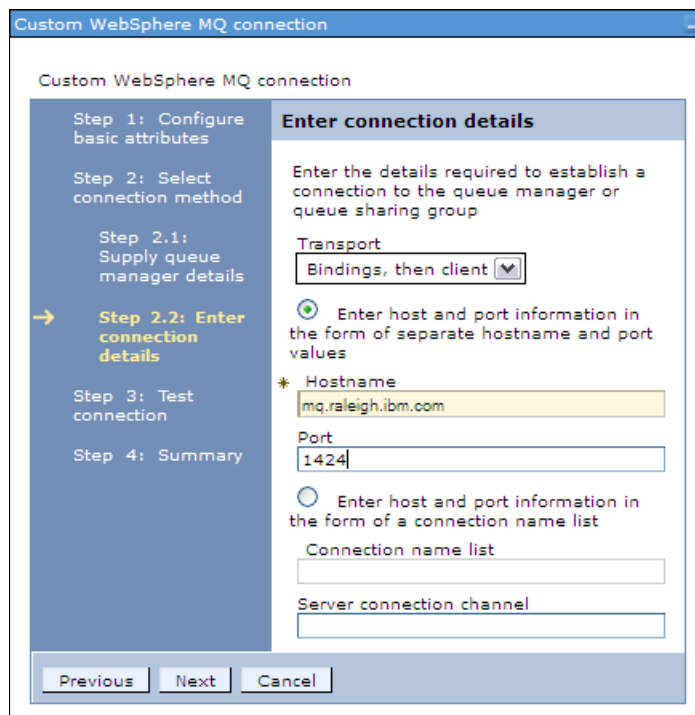


Figure 11-20 WebSphere MQ messaging provider activation specification, connection method

9. Click **Test connection** to verify that you can connect to the WebSphere MQ queue manager and then click **Next**.
10. In the Summary window, review the summary and then click **Finish**. The new activation specification is created.
11. Save the changes to the configuration.

WebSphere Application Server V8.5 exposes several WebSphere MQ connection properties to configure the WebSphere MQ resource adapter that is used by the WebSphere MQ messaging provider. Among these properties, which affect the connection pool that is used by activation specifications, are:

- ▶ `maxConnections`
- ▶ `connectionConcurrency` (Setting this property only affects WebSphere Application Server V7 nodes. The property has no effect for nodes in WebSphere Application Server V8 or later.)
- ▶ `reconnectionRetryCount`
- ▶ `reconnectionRetryInterval`

To configure these properties, go to the navigation tree, and click **JMS providers** → **WebSphere MQ messaging provider**. Under Additional properties, click **Resource adapter properties**, as shown in Figure 11-21.

The screenshot shows a 'General Properties' dialog box with a 'Connection pool properties' section. The properties are as follows:

| Property Name | Value | Unit |
|-----------------------------|--------|--------------|
| Max connections | 50 | connections |
| Connection concurrency | 1 | |
| Reconnection retry count | 5 | retries |
| Reconnection retry interval | 300000 | milliseconds |

Figure 11-21 WebSphere MQ resource adapter properties

11.4 Configuring resources for third-party messaging providers

For messaging between application servers, most requirements can be satisfied using either the WebSphere Application Server default messaging provider or the WebSphere MQ messaging provider. However, if desired, you can instead use a third-party messaging provider (that is, another company's product).

To administer a third-party messaging provider, use either the resource adapter (for a Java EE Connector Architecture (JCA) 1.5-compliant or 1.6-compliant messaging provider) or the client (for a non-JCA messaging provider) that is supplied by the third party.

In this section, we describe how to configure a third-party, non-JCA messaging provider using the administrative console. You can configure any third-party non-JCA messaging provider that supports the JMS Version 1.1 unified connection factory. For details about how to configure third-party JCA messaging provider refer to the WebSphere Application Server information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/tmp_ep_ra.html

11.4.1 Configuring JMS messaging providers

To define a new third-party messaging provider:

1. Click **Resources** → **JMS** → **JMS providers**. In the JMS providers window, select the **Scope** (use **Cell scope** in this sample) and then click **New**.
2. Enter the following properties:
 - Name
 - Class path
 - Native library path
 - External initial context factory. This property is the Java class name of the third-party JMS provider's initial context factory. For example, for the ActiveMQ JMS provider, this is `org.apache.activemq.jndi.ActiveMQInitialContextFactory`.
 - External provider URL. This is the JMS provider URL for external JNDI lookups. The external provider URL specifies how the initial context factory connects to the external naming service. The format of the external provider URL is:
`<protocol>://<host name>:<port number>`
3. Click **OK**. A new JMS provider is created.
4. Save the changes to the configuration.

11.4.2 Configuring JMS connection factories

To configure a JMS connection factory for a third-party JMS provider:

1. Click **Resources** → **JMS** → **Connection factories**. In the Connection factories window, click **Scope** (use **Cell scope** in this sample) and then click **New**.
2. Choose the third-party JMS provider that you created in the previous sub-section and then click **OK**.
3. Enter the following basic properties:
 - Name
 - JNDI name
 - External JNDI name

Figure 11-22 shows the configuration for this example.

The screenshot shows a 'General Properties' dialog box with the following fields and values:

- Scope:** Cell=was85Cell01
- Provider:** GenericJMSProvider
- * Name:** SampleGenericJMSCF
- * Type:** UNIFIED
- * JNDI name:** jms/SampleGenericJMSCF
- Description:** (Empty text area)
- Category:** (Empty text field)
- * External JNDI name:** SampleGenericJMSCF

Figure 11-22 Third-party JMS connection factory properties

You can configure other properties for this connection factory if needed. For detailed information about other properties, refer to the product information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/a/e/tmj_adm29.html

4. Click **OK**. The new connection factory is created.
5. Save the changes to the configuration.

11.4.3 Configuring JMS destinations

You can configure both queue and topic destinations for third-party JMS messaging providers. In this section, we describe how to create and configure a JMS queue. For information about creating and configuring a JMS topic, refer to *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770.

To configure a JMS destination:

1. Click **Resources** → **JMS** → **Queues**. In the Queues window, click **Scope** (this sample uses **Cell scope**) and then click **New**.
2. Select the **Generic JMS provider** option and then click **OK**.
3. Enter the following basic properties:
 - Name
 - JNDI name
 - External JNDI name

Figure 11-23 shows the configuration for this example.

The screenshot shows a configuration window titled "General Properties" with a blue header bar. Below the header, there are several labeled fields:

- * Scope**: A text box containing "Cell=was85Cell01".
- Provider**: A text box containing "GenericJMSProvider".
- * Name**: A text box containing "SampleGenericJMSQueue".
- * Type**: A text box containing "QUEUE".
- * JNDI name**: A text box containing "jms/SampleGenericJMSQueue".
- Description**: A large, empty text area with a small icon in the bottom right corner.
- Category**: A text box that is currently empty.
- * External JNDI Name**: A text box containing "SampleGenericJMSQueue".

Figure 11-23 Third-party JMS queue properties

4. Click **OK**. The new queue is created.
5. Save the changes to the configuration.



Configuring and managing web servers

Web servers are an important component of the entire architecture or solution. They provide scalability, security, and additional control over system performance. The IBM WebSphere Application Server is compatible with many web servers, which can be managed easily by federating them to the WebSphere Application Server. Architecture of this foundation provides more flexibility, control, and ease of operations.

In this chapter, we present the basic concepts of web servers, their features, and importance. We also focus on installations, configurations, testing the configurations, and troubleshooting. WebServer also plays important roles in troubleshooting the issues.

The following topics are covered:

- ▶ Web server overview and basic concepts
- ▶ Installations
- ▶ Web server configuration using the WebSphere Customization Toolbox
- ▶ Working with web servers and plug-ins
- ▶ Working with the plug-in configuration file
- ▶ IBM HTTP Server and Web Server Plug-ins for IBM WebSphere Application Server for z/OS
- ▶ Troubleshooting some common errors

12.1 Web server overview and basic concepts

Web servers are one of the most critical components in the entire architecture. They work as a front end for WebSphere Application Server and provide the load balancing for WebSphere Application Server. They enable WebSphere Application Server to achieve scalability and also provide better control over performance and security.

Web servers provide the following features in a WebSphere Application Server environment:

- ▶ **Load Balancing:** Through plug-in files, the server has information about back-end Java virtual machines (JVMs) and applications. Based on settings, the server forwards requests to its corresponding back-end server. These plug-ins contain the entire information about the application server, such as clusters, applications, session affinity, and the weight of JVM.

Plug-ins have to be on the local server where the web server is installed. The path of the plug-ins must also be in the configuration file of the web server.

- ▶ **Security:** Through web servers, we get an additional layer of security. We can enable forwarding, based on security settings. Secure Sockets Layer (SSL) certificates can be deployed on web servers that provide SSL enabling from outside the network to the web server. It is complex to use an SSL certificate with WebSphere Application Server directly without web servers. Apart from SSL, the system is also protected from the direct traffic of untrusted networks by filtering and forwarding mechanisms applied at the web server level.
- ▶ **Performance Control:** It helps in providing additional control over connection pools. Various performance tuning parameters, such as threads, timeout, and child threads, provide additional layers of performance in the system. Based on the web container connection pool in WebSphere Application Server, we can control the incoming traffic through web servers.
- ▶ **(New in V8.5.5)** With WebSphere Application Server Network Deployment, Apache and IBM HTTP Server web servers can also be enabled for Intelligent Management features. When Intelligent Management is enabled in the WebSphere plug-in, routing information is not defined in the `plugin-cfg.xml` file. Instead, the plug-in connects to a REST service to dynamically gather routing information for one or more WebSphere cells. This option is discussed further in Chapter 13, “Intelligent management” on page 469.

There are more available parameters that help to manage the application requests. Even a short list lets us understand how important web servers are for our system environments.

Plug-ins installed on the system with the web server are the bridges between the web servers and WebSphere Application Server. The plug-in directory path must be defined in the web server configuration file. For better management, administration, and to provide standardization, WebSphere Application Server comes with a plug-in installer. Using this installer is a best practice.

A plug-in configuration file, generated on the application server and placed on the web server, is used for routing information. To manage the generation and propagation of these plug-in configuration files, web servers are defined to the WebSphere Application Server configuration repository. In some cases, web server configuration and management features are also available from the WebSphere administrative tools.

Web servers are defined to WebSphere Application Server. A web server resides on a managed or unmanaged node. If located on a managed node, in a distributed server environment only, a node agent is installed on the web server system and belongs to a WebSphere Application Server administrative cell. The administrative tools communicate with

the web server through the node agent. If located on an unmanaged node, the web server is still defined to the cell. However, it does not have a node agent running to manage the process. In either case, the web server definition allows you to generate the plug-in configuration file for the web server.

Concept: Whether a web server resides on a managed or an unmanaged node, the point is that it can be managed through WebSphere administrative console. If the web server is installed on the WebSphere cell node, that same node agent can be used; otherwise, it is an unmanaged node. Essentially, this design provides ease in generating and propagating plug-ins, stopping and starting the web server, and changing the configurations. Either option operates similarly and both send plug-in requests to the WebSphere Application Server.

Web applications can be mapped to web servers. This mapping is used to generate routing information during plug-in configuration generation.

IBM HTTP Server V8.5 is bundled with WebSphere Application Server V8.5. The administrative functionality is integrated into WebSphere Application Server to provide remote administration through the administrative console. This enhanced administrative function is only available to the IBM HTTP Server.

The following list notes the supported web servers for WebSphere Application Server V8.5:

- ▶ Apache HTTP Server
- ▶ IBM Lotus Domino Web Server
- ▶ IBM HTTP Server
- ▶ Microsoft Internet Information Services
- ▶ Sun Java System Web Server (formerly Sun ONE and iPlanet)
- ▶ HTTP Server for zOS

12.1.1 Request routing using the plug-in

The web server plug-in uses an XML configuration file to determine whether a request is for the web server or the application server. When a request reaches the web server, the URL is compared to the URLs managed by the plug-in. If a match is found, the plug-in configuration file contains the information needed to forward that request to the web container using the web container inbound transport chain. See Figure 12-1 on page 420.

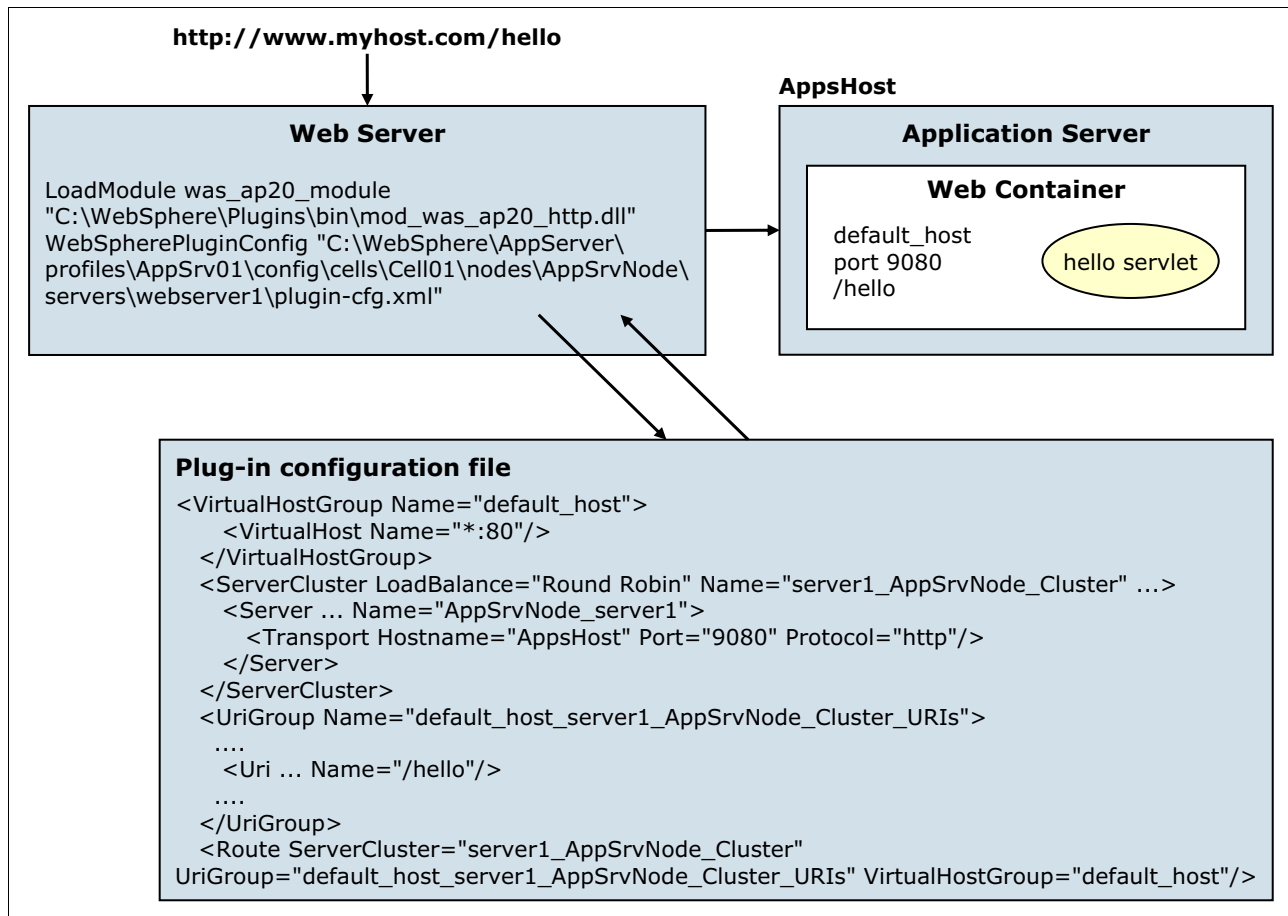


Figure 12-1 Web server plug-in routing

The plug-in configuration file is generated using the WebSphere administrative tools. Each time you make a change to the WebSphere Application Server configuration that affects how requests are routed from a web server to the application server, you need to regenerate and propagate the plug-in configuration file to the web server. You can propagate the file manually or configure the propagation to be done automatically.

12.1.2 Web server and plug-in management

The setup of your web server and web server plug-in environment is defined in a web server definition. The web server definition includes information about the location of the web server, its configuration files, and plug-in configuration. Each web server is association with a node, either managed or unmanaged. The web server definition is configured as part of the plug-in installation process. The web server definition is also used during application deployment. Web modules can be mapped to a web server, ensuring the proper routing information is generated for the plug-in configuration file.

Web server definitions are located under **Servers** → **Server Types** → **Web servers** in the administrative console. See Figure 12-2.

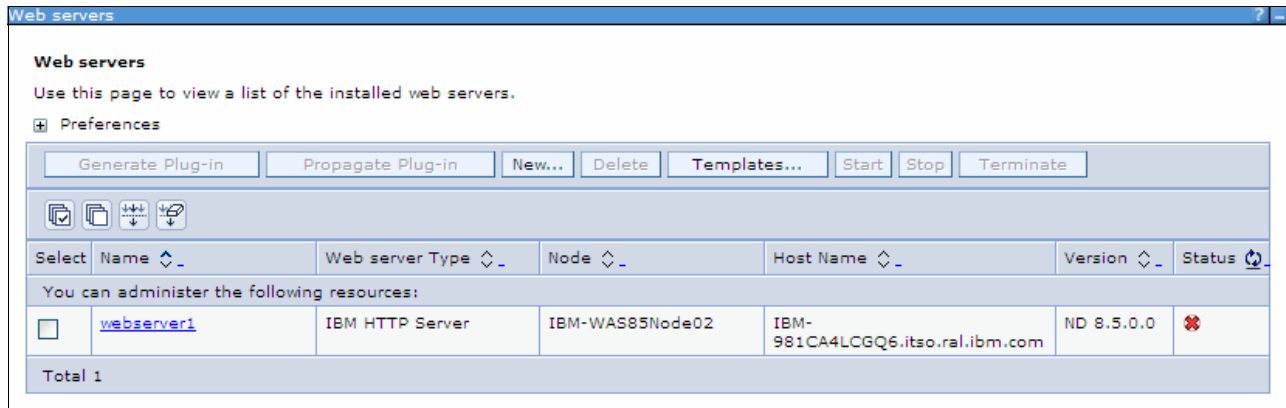


Figure 12-2 Web server definition

Contrasting managed and unmanaged

In a distributed server environment, you can define multiple web servers. These web servers can be defined on managed or unmanaged nodes depending on the environment on which you are running the web server. This section covers the differences between managed and unmanaged nodes.

WebSphere Application Server supports basic administrative functions for all supported web servers. For example, the generation of a plug-in configuration can be performed for all web servers. If the web server is defined on a managed node, automatic propagation of the plug-in configuration can be performed using node synchronization. If the web server is defined on an unmanaged node, automatic propagation of a plug-in configuration is only supported for IBM HTTP Servers.

WebSphere Application Server supports some additional administrative console tasks for IBM HTTP Servers on managed and unmanaged nodes. For example, you can start IBM HTTP Servers, stop them, terminate them, display their log files, and edit their configuration files.

Unmanaged nodes

An *unmanaged node* does not have a node agent to manage its servers. In a stand-alone server environment, you can define one web server and it, by necessity, resides on an unmanaged node. In a distributed server environment, web servers defined to an unmanaged node are typically remote web servers. The web server is usually found in the demilitarized zone (DMZ) outside the firewall. The DMZ is a safe zone between firewalls that is typically located between a client and a back-end server.

Figure 12-3 displays a web server configured on an unmanaged node. In this configuration, the plug-in configuration file is generated on the deployment manager. The plug-in configuration file must be manually propagated to the web server on the unmanaged node. To start or stop this web server, use the specific web server administrative tools.

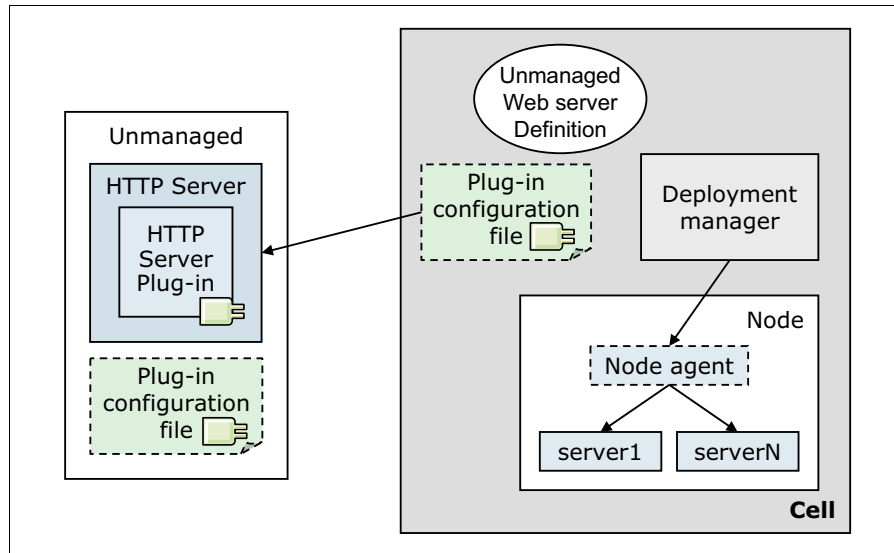


Figure 12-3 Unmanaged web server on an unmanaged node

The IBM HTTP Server is a special case. If your web server is the IBM HTTP Server, you can configure the server on an unmanaged node and still be able to administer the web server using the WebSphere administrative tools. This unmanaged node does not need a node agent on the web server machine. The IBM HTTP Server administration process provides administrative functions for the IBM HTTP Server within WebSphere.

Figure 12-4 shows an IBM HTTP Server web server configured on an unmanaged node. In this configuration, the plug-in configuration file is generated on the deployment manager. Because the web server is the IBM HTTP Server, you can automatically propagate the plug-in configuration file to the remote server, make configuration changes to the web server configuration file, and use the administrative console to start and stop the web server.

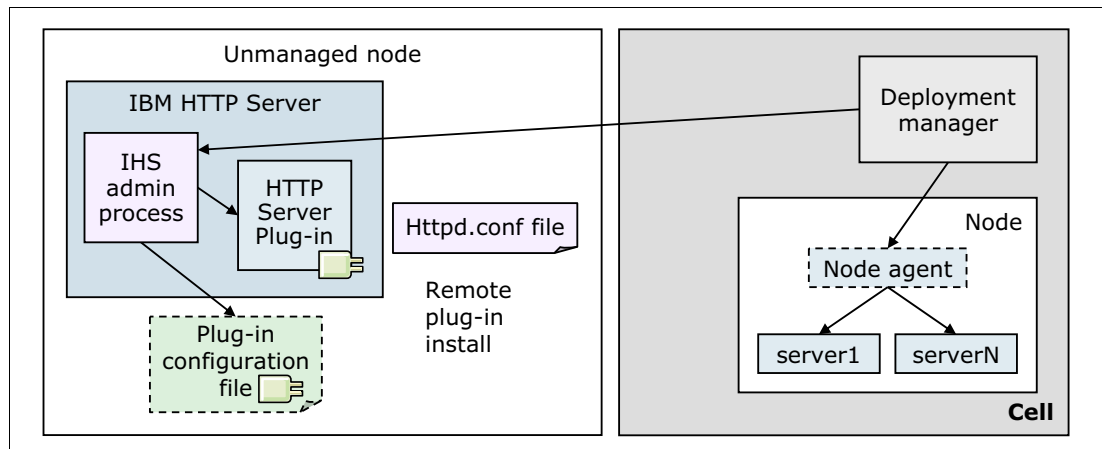


Figure 12-4 IBM HTTP Server on an unmanaged node

If the web server is defined to an unmanaged node, complete these steps:

1. Check the status of the web server.
2. Generate a plug-in configuration file for that web server.

If the web server is an IBM HTTP Server and the IBM HTTP Server Administration server is installed and properly configured, you can also:

- a. Display the IBM HTTP Server Error log (`error.log`) and Access log (`access.log`) files.
- b. Start and stop the server.
- c. Display and edit the IBM HTTP Server configuration file (`httpd.conf`).
- d. Propagate the plug-in configuration file after it is generated.

You cannot propagate an updated plug-in configuration file to a non-IBM HTTP Server web server that is defined to an unmanaged node. You must install an updated plug-in configuration file manually to a web server that is defined to an unmanaged node.

Managed nodes

A *managed node* has a node agent for managing web servers. You can use the WebSphere administrative tools to communicate with web servers on a managed node. Figure 12-5 displays a web server configured on a managed node. In this configuration, the plug-in configuration file is generated on the deployment manager. The plug-in configuration file is automatically propagated to the web server on the managed node. To start or stop this web server, you can use the administrative console.

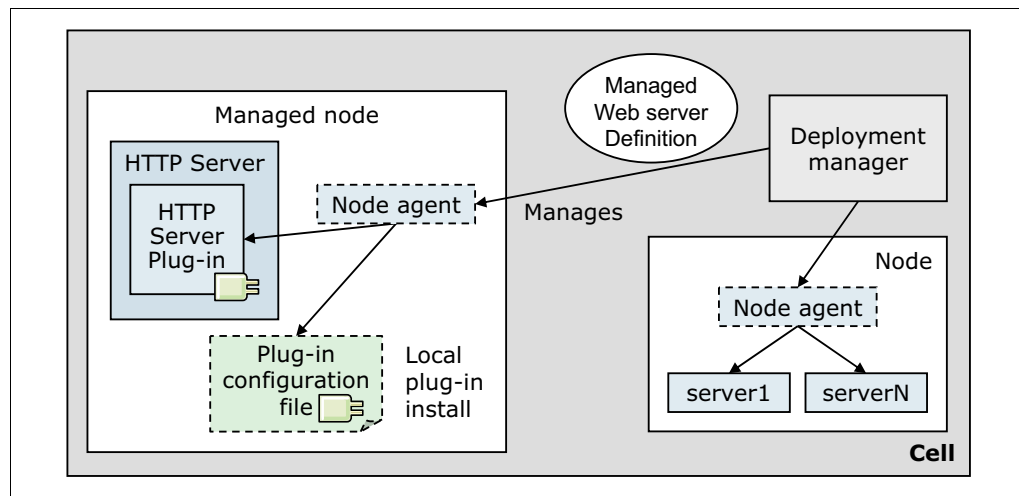


Figure 12-5 Managed web server on a managed node

If the web server is defined to a managed node, complete the following steps:

1. Check the status of the web server.
2. Generate a plug-in configuration file for that web server.
3. Propagate the plug-in configuration file after it is generated.

If the web server is an IBM HTTP Server and the IBM HTTP Server Administration server is installed and properly configured, you can also:

- a. Display the IBM HTTP Server Error log (`error.log`) and Access log (`access.log`) files.
- b. Start and stop the server.
- c. Display and edit the IBM HTTP Server configuration file (`httpd.conf`).

How nodes and servers are defined

During the installation of the plug-in, the Plug-ins installation wizard creates a web server configuration script named `configureWeb_server_name`. This configuration script is used to create the web server definition and, if necessary, the node definition in the configuration of the application server.

If a web server definition already exists for a stand-alone application server, running the script does not add a new web server definition. Each stand-alone application server can have only one web server definition. A managed node, conversely, can have multiple web server definitions. The script creates a new web server definition unless the web server name is the same.

The Plug-ins installation wizard stores the script in the `<plug-in_home>/bin` directory on the web server machine. If the plug-in is installed locally (on the same machine as the application server), the configuration script is run automatically.

For remote installations, you must copy the script from the web server machine to the `<was_home>/bin` directory on the application server machine for execution. The script runs against the default profile. If one machine is running under Linux or UNIX and the other machine is running under Windows, use the script created in the `<plug-in_home>/Plugins/bin/crossPlatformScripts` directory.

Note: Always open a new command window in which to execute the `configureWeb_server_name` script. There is a potential conflict between a shell environment variable, the `WAS_USER_SCRIPT` variable, and the real default profile. The script always works against the default profile. However, if the `WAS_USER_SCRIPT` environment variable is set, a conflict arises as the script attempts to work on the profile identified by the variable.

If you need to create a web server definition for a distributed server environment, you must federate your stand-alone application servers to the deployment manager first. Any web server definitions created for a stand-alone application server are lost when they are federated into a cell.

Using administrative tools: In a distributed server environment, the administrative console can also be used to define the nodes and web servers. For more details, see 12.4.1, “Manually defining nodes and web servers” on page 439.

12.2 Installations

You can install a web server and web server plug-in using the Installation Manager. You can perform the installation using the GUI, command line, console mode, or silently. To install the IBM HTTP Server and web server plug-in, complete the following steps:

1. Install IBM Installation Manager.
2. Launch the Installation Manager GUI.
3. Configure the Installation Manager repository to point to the Supplements package.
4. Click the **Install wizard** to begin the installation.
5. Select the following packages for installation:
 - **IBM HTTP Server for WebSphere Application Server**

– **Web Server Plug-ins for IBM WebSphere Application Server**

Click **Next**.

6. Read and accept the license agreement. Click **Next**.
7. Select the installation directory for each package. You can keep the default paths or update them to suit your environment. Click **Next**.
8. On the Features window, verify the selected packages, and click **Next**.
9. Configure a port number for the IBM HTTP Server to communicate. Keep the default port 80 or modify to a port number that is not in use. Click **Next**.
10. Review the settings on the Summary window, and click **Install**.
11. When the installation is complete, review the summary, and click **Finish**. It is also a best practice to view the log file to verify that the installation was successful.

After installation of the web server and web server plug-in, you need to configure the web server plug-in. To configure the plug-in, the stand-alone WebSphere Customization Toolbox offering must be installed. You can find more information about the stand-alone WebSphere Customization Toolbox offering in 2.6.1, “WebSphere Customization Toolbox” on page 52.

12.3 Web server configuration using the WebSphere Customization Toolbox

After installing the web server plug-in, you must configure it. A new tool in WebSphere Application Server V8.5 is the Web Server Plug-in Configuration Tool in the WebSphere Customization Toolbox (WCT), which is used for configuring web server plug-ins. The Web Server Plug-in Configuration Tool creates one or more configurations for the web server plug-ins that can direct requests from a web client through the web server and then interact with applications running on an application server. The Web Server Plug-in Configuration Tool edits the configuration file or files for a web server by creating directives that point to the location of the binary plug-in module and the plug-in configuration file.

Before configuring the plug-in, determine the topology set up. The options for defining and managing web servers depend on your chosen web server topology and your WebSphere Application Server package. Decisions to make include whether to collocate the web server with other WebSphere Application Server processes and whether to make the web server managed or unmanaged.

The following examples outline the process required to create each sample topology. Note that each example assumes that only the WebSphere processes shown in the diagrams are installed on each system and that the profile for the process is the default profile.

This section is not a substitute for using the product documentation but is intended to help you understand the process. For detailed information about how to complete a local or remote installation scenario, see the *Web Server Plug-in Installation Roadmaps for WebSphere Application Server Network Deployment Version 8.0* guide that comes with the plug-in. You can also find this information at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multipatform.doc%2Fae%2Ftins_road_plugins.html

12.3.1 Configuration files

When configuring a web server and plug-in, a number of configuration files are created and used:

- ▶ Web server configuration file

This file is installed as part of the web server, for example, if you install the IBM HTTP Server, the web server configuration file is `httpd.conf`. During configuration of the plug-in, the Web Server Plug-in Configuration Tool adds directives to the file specifying the location of the binary web server plug-in file and the plug-in configuration file (`plugin-cfg.xml`).

- ▶ Binary web server plug-in file

This file resides on the web server machine. An example of a binary web server plug-in file on Linux for the IBM HTTP Server is `mod_was_ap22_http.do`. The configuration file for the binary web server plug-in is the `plugin-cfg.xml` file. The binary module reads this XML file to discover where to route requests.

- ▶ Plug-in configuration file (`plugin-cfg.xml`)

The plug-in configuration file is generated on an application server or deployment manager machine and must be copied to the web server machine. As you make changes to your environment, such as deploying applications, creating clusters, updating virtual hosts, and so on, this information needs to be placed in the plug-in configuration file to ensure proper routing of content. The plug-in configuration file needs to be generated and then propagated to the web server. See 12.5.1, “Regenerating the plug-in configuration file” on page 452 and 12.5.2, “Propagating the plug-in configuration file” on page 457 for more information.

- ▶ Default (temporary) plug-in configuration file

This is a temporary plug-in configuration file that is generated by the Web Server Plug-in Configuration Tool for every remote installation scenario. This file is replaced by the Plug-in configuration file (`plugin-cfg.xml`) that is relevant for your environment.

- ▶ The `configureweb_server_name` script

This script is created by the Web Server Plug-in Configuration Tool on the web server machine. Copy this script to the `profile_root/bin` directory of the application server or deployment manager. You need to run the script to create a web server definition in the application server or deployment manager configuration.

12.3.2 Stand-alone server environment

In a stand-alone server environment, a web server can be remote to the application server machine or local, but there can only be one defined to WebSphere Application Server. The web server always resides on an unmanaged node.

Remote web server

In this scenario, the application server and the web server are on separate machines. The web server machine can reside in the internal network, or more likely, will reside in the DMZ. Use this configuration for a production environment. See Figure 12-6 on page 427.

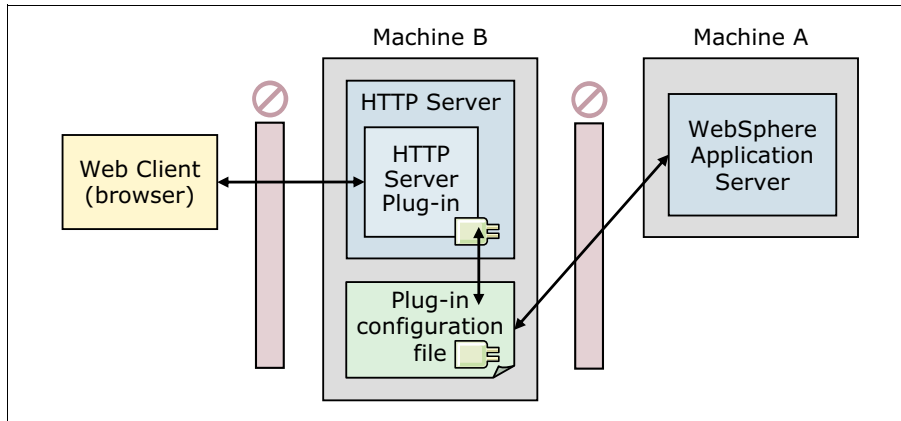


Figure 12-6 Remote web server in a stand-alone server environment

Assume that the application server is already installed and configured on Machine A. Complete the following steps:

1. Install Installation Manager on Machine B.
2. Install the web server, web server plug-in, and WebSphere Customization Toolbox on Machine B.
3. Configure the web server plug-in on Machine B by completing the following steps:
 - a. Launch the Web Server Plug-ins Configuration Tool.
 - b. Configure a *remote* web server plug-in.
 - c. Configure a web server definition. The default is `webserver1`.

During configuration, the following tasks are completed:

- a. A default temporary plug-in configuration file is created and placed into the location specified.
 - b. The web server configuration file is updated with the plug-in configuration, including the location of the plug-in configuration file.
 - c. A script is generated to define the web server to WebSphere Application Server. The script is located in:


```
<plug-in_home>/bin/configure<web_server_name>
```
4. At the end of the plug-in configuration, copy the `configure<web_server_name>` script to the `<profile_root>/bin` directory of the application server machine, Machine A. Start the application server, and then execute the script.
 5. When the web server is defined to WebSphere Application Server, the plug-in configuration file is generated automatically. For the IBM HTTP Server, the new plug-in file is propagated to the web server automatically. For other web server types, you need to propagate the new plug-in configuration file to the web server.
 6. Start the web server, and verify the configuration by accessing an application from a web client.

Local web server

In this scenario, a stand-alone application server exists on machine A. The web server and web server plug-in are also installed on machine A. This topology is suited to a development environment or for internal applications. See Figure 12-7 on page 428.

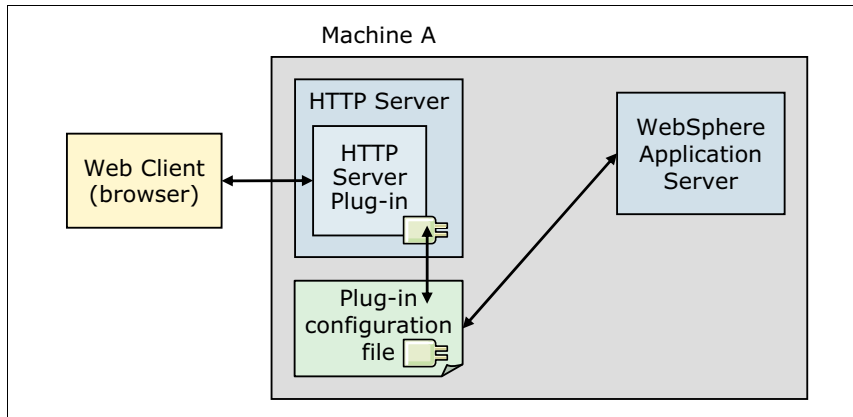


Figure 12-7 Local web server in a stand-alone server environment

Assume the application server is already installed and configured. Complete the following steps:

1. Install the web server, web server plug-in, and WebSphere Customization Toolbox on Machine A.
2. Configure the web server plug-in on Machine A by completing the following steps:
 - a. Launch the Web Server Plug-ins Configuration Tool.
 - b. Configure a *local* web server plug-in.
 - c. Configure a web server definition. The default is `webserver1`.

During configuration, the following tasks are performed:

- a. A default temporary plug-in configuration file is created and placed into the location specified.
 - b. The web server configuration file is updated with the plug-in configuration, including the location of the plug-in configuration file.
 - c. A script to define the web server to WebSphere Application Server is generated. The script is located in:

```
<plug-in_home>/bin/configure<web_server_name>
```
3. At the end of the plug-in configuration, copy the `configure<web_server_name>` script to the `<profile_root>/bin` directory of the application server machine, Machine A. Start the application server and then execute the script.
 4. When the web server is defined to WebSphere Application Server, the plug-in configuration file is generated automatically. Because this is a local installation, you do not have to propagate the new plug-in configuration to the web server. For the IBM HTTP Server, the new plug-in file is propagated to the web server automatically. For other web server types, you need to propagate the new plug-in configuration file to the web server.
 5. Start the application server and the web server, and verify the configuration by accessing an application from a web client.

12.3.3 Distributed server environment

Web servers in a distributed server environment can be local to the application server or remote. The web server can also reside on the deployment manager system. You have the possibility of defining multiple web servers and the web servers can reside on managed or unmanaged nodes.

Note: If you are planning to add the application server node into a deployment manager cell but have not done so yet, start the deployment manager and federate the node before configuring the plug-in. You cannot add an application server with a web server definition into the deployment manager cell.

Remote web server

The deployment manager and the web server are on separate machines. The web server machine can reside in the internal network, or more likely, it resides in the DMZ. Use this configuration for a production environment.

Note that this scenario and the process are almost identical to the process outlined for a remote web server in a stand-alone server environment. The primary difference is that the script that defines the web server is run against the deployment manager, and you will see an unmanaged node created for the web server node. In Figure 12-8, the node is unmanaged because there is no node agent on the web server system.

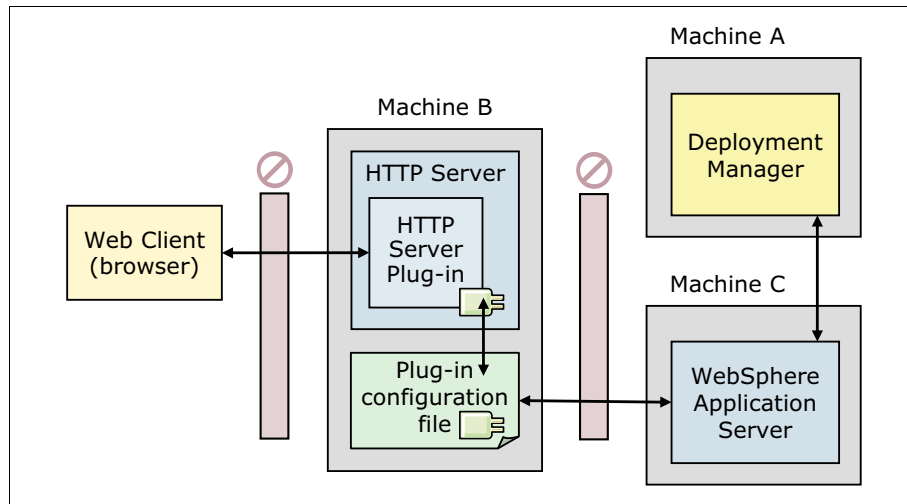


Figure 12-8 Remote web server in a stand-alone server environment

Assume that the deployment manager is already installed and configured on Machine A and the application server is already installed and configured on Machine C. Complete the following steps:

1. Install Installation Manager on Machine B.
2. Install the web server, web server plug-in, and WebSphere Customization Toolbox on Machine B.
3. Configure the web server plug-in on Machine B by completing the following steps:
 - a. Launch the Web Server Plug-ins Configuration Tool.
 - b. Configure a *remote* web server plug-in.
 - c. Configure a web server definition. The default is `webserver1`.

During configuration, the following tasks are performed:

- a. A default temporary plug-in configuration file is created and placed into the location specified.
- b. The web server configuration file is updated with the plug-in configuration, including the location of the plug-in configuration file.
- c. A script is generated to define the web server to WebSphere Application Server. The script is located in:

```
<plug-in_home>/bin/configure<web_server_name>
```

4. At the end of the plug-in configuration, copy the `configure<web_server_name>` script to the `<profile_root>/bin` directory of the deployment manager machine, Machine A. Start the deployment manager and then execute the script.
5. When the web server is defined to WebSphere Application Server, the plug-in configuration file is generated automatically. For the IBM HTTP Server, the new plug-in file is propagated to the web server automatically. For other web server types, you need to propagate the new plug-in configuration file to the web server.
6. Start the web server, and verify the configuration by accessing an application from a web client.

Local to a federated application server

In this scenario, the web server is installed on a machine that also has a managed node. Note that this scenario functions the same if the deployment manager were installed on Machine B. See Figure 12-9.

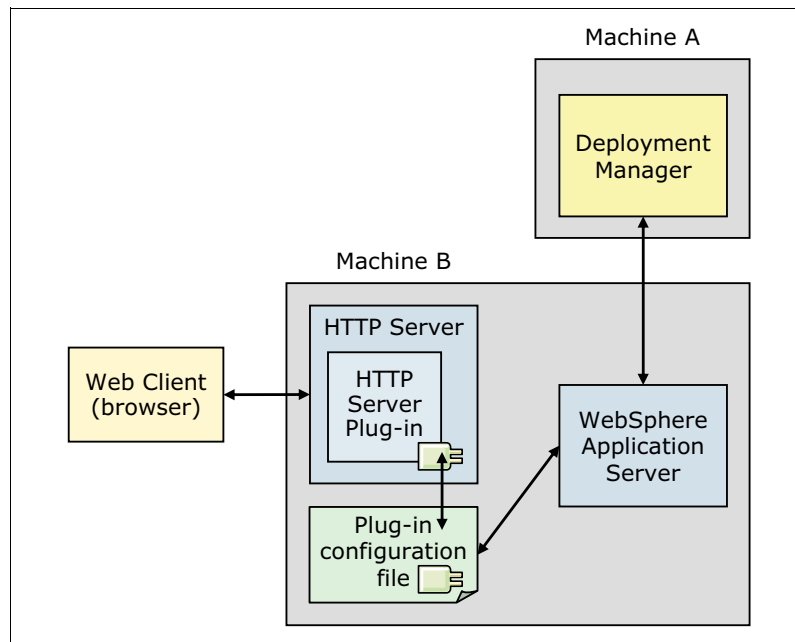


Figure 12-9 Web server installed locally on an application server system

Assume that the deployment manager is already installed and configured on Machine A and the application server is already installed and configured on Machine B. Complete the following steps:

1. Install the web server, web server plug-in, and WebSphere Customization Toolbox on Machine B.

2. Configure the web server plug-in on Machine B by doing the following:
 - a. Launch the Web Server Plug-ins Configuration Tool.
 - b. Configure a *local* web server plug-in.
 - c. Configure a web server definition. The default is `webserver1`.

During configuration, the following tasks are performed:

- a. A default temporary plug-in configuration file is created and placed into the location specified.
 - b. The web server configuration file is updated with the plug-in configuration, including the location of the plug-in configuration file.
 - c. A script is generated to define the web server to WebSphere Application Server. The script is located in:

```
<plug-in_home>/bin/configure<web_server_name>
```
3. At the end of the plug-in configuration, copy the `configure<web_server_name>` script to the `<profile_root>/bin` directory of the deployment manager machine, Machine A. Start the deployment manager and then execute the script.
 4. When the web server is defined to WebSphere Application Server, the plug-in configuration file is generated automatically. For the IBM HTTP Server, the new plug-in file is propagated to the web server automatically. For other web server types, you need to propagate the new plug-in configuration file to the web server.
 5. Start the web server, and verify the configuration by accessing an application from a web client.

The plug-in configuration file is generated automatically and is propagated at the next node synchronization.

12.3.4 Configuring a remote web server in a distributed environment

Assume that the Installation Manager, deployment manager, and application server are all installed and configured. The IBM HTTP Server and web server plug-in are also installed. Complete the following steps to configure a remote web server in a distributed environment using the Web Server Plug-ins Configuration Tool:

1. Launch the stand-alone WebSphere Customization Toolbox.
2. Select the **Web Server Plug-ins Configuration Tool** and then click **Launch Selected Tool**.
3. Select a plug-in runtime location. Click **Add** in the Web Server Plug-in Runtime Locations tab. See Figure 12-10 on page 432:
 - a. Enter a name for the plug-in location.
 - b. Browse to the location of where the plug-in is installed.

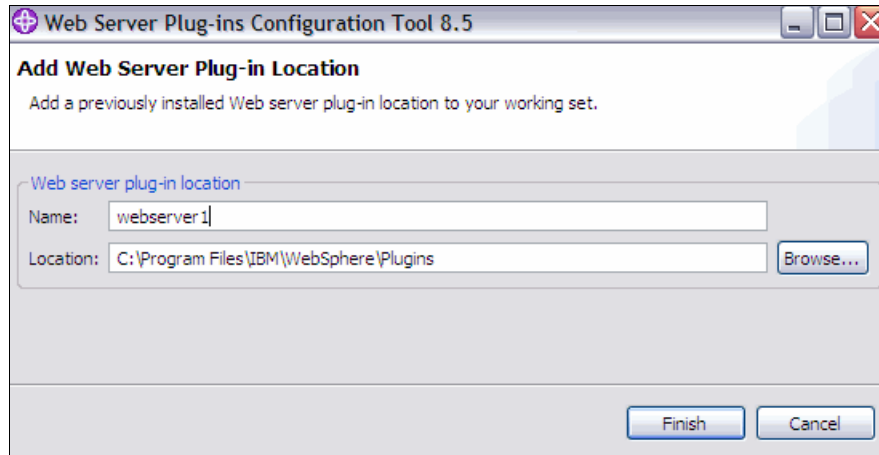


Figure 12-10 Web server plug-in location

Click **Finish**.

4. Add the web server configuration information. In the Web Server Plug-in Configurations area, click **Create**.
5. Select the web server you want to configure, and click **Next**. See Figure 12-11.

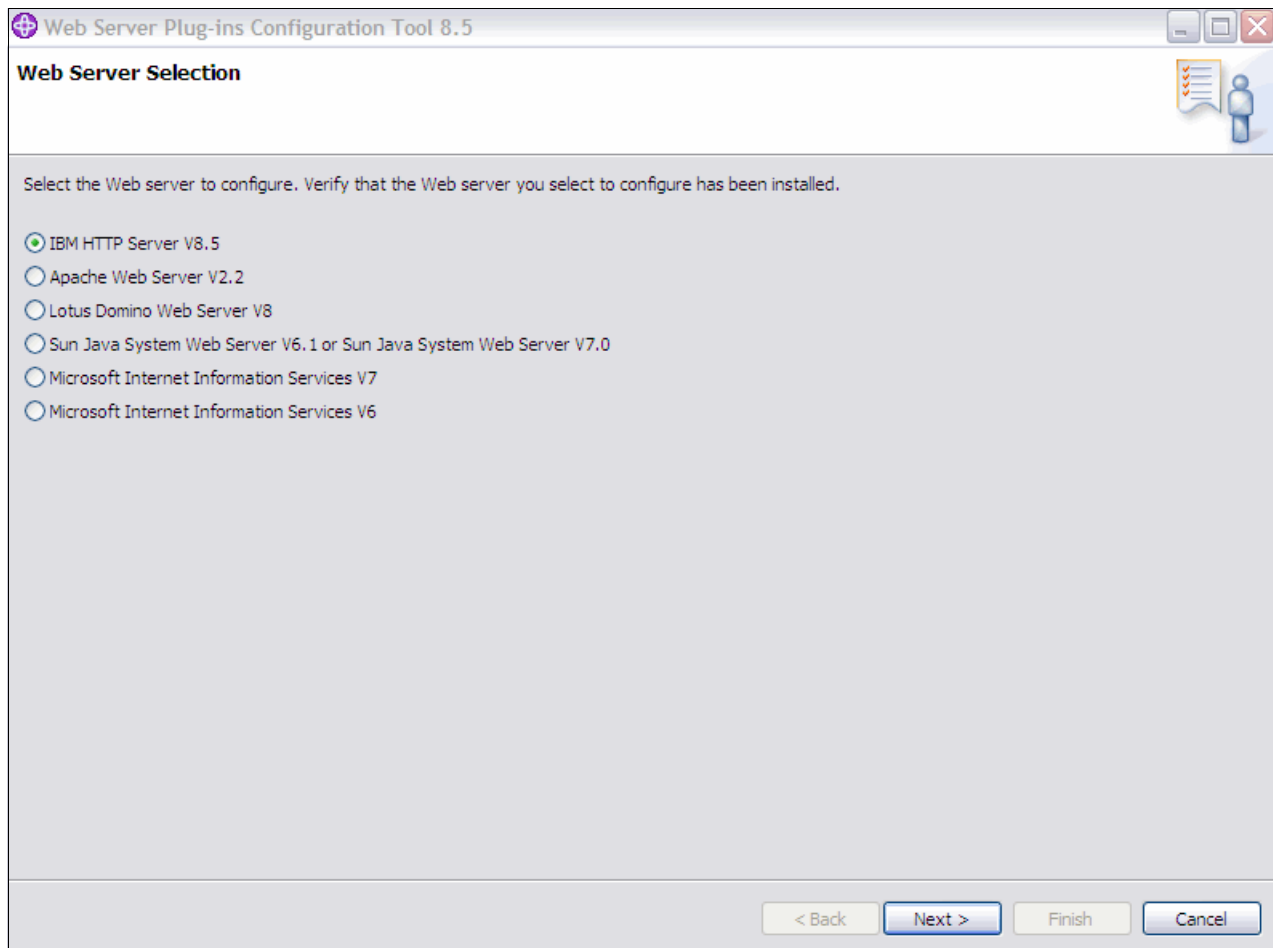


Figure 12-11 Web server selection

6. Select the existing **HTTP Server configuration file** and provide the web server port. See Figure 12-12.

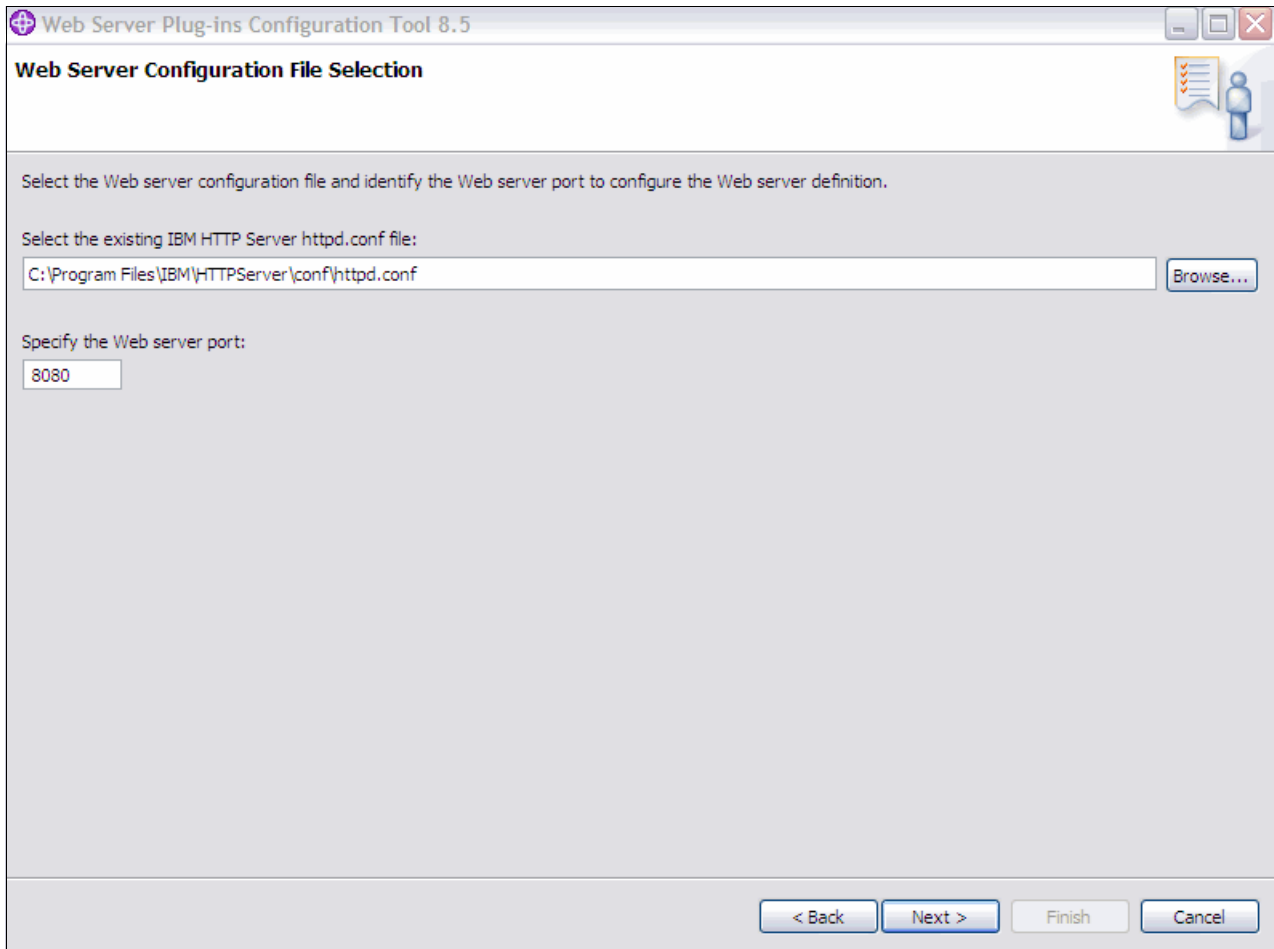


Figure 12-12 Web server configuration file selection

Click **Next**.

7. If you are configuring an IBM HTTP web server plug-in, complete the following steps:
 - a. Enter a port number for the administration server. Keep the default of 8008 or enter a unique port.
 - b. Create a user ID for the administration server authentication. In Figure 12-13 on page 434, the user ID is `ihsadmin` and the Password is `ihsadmin`.

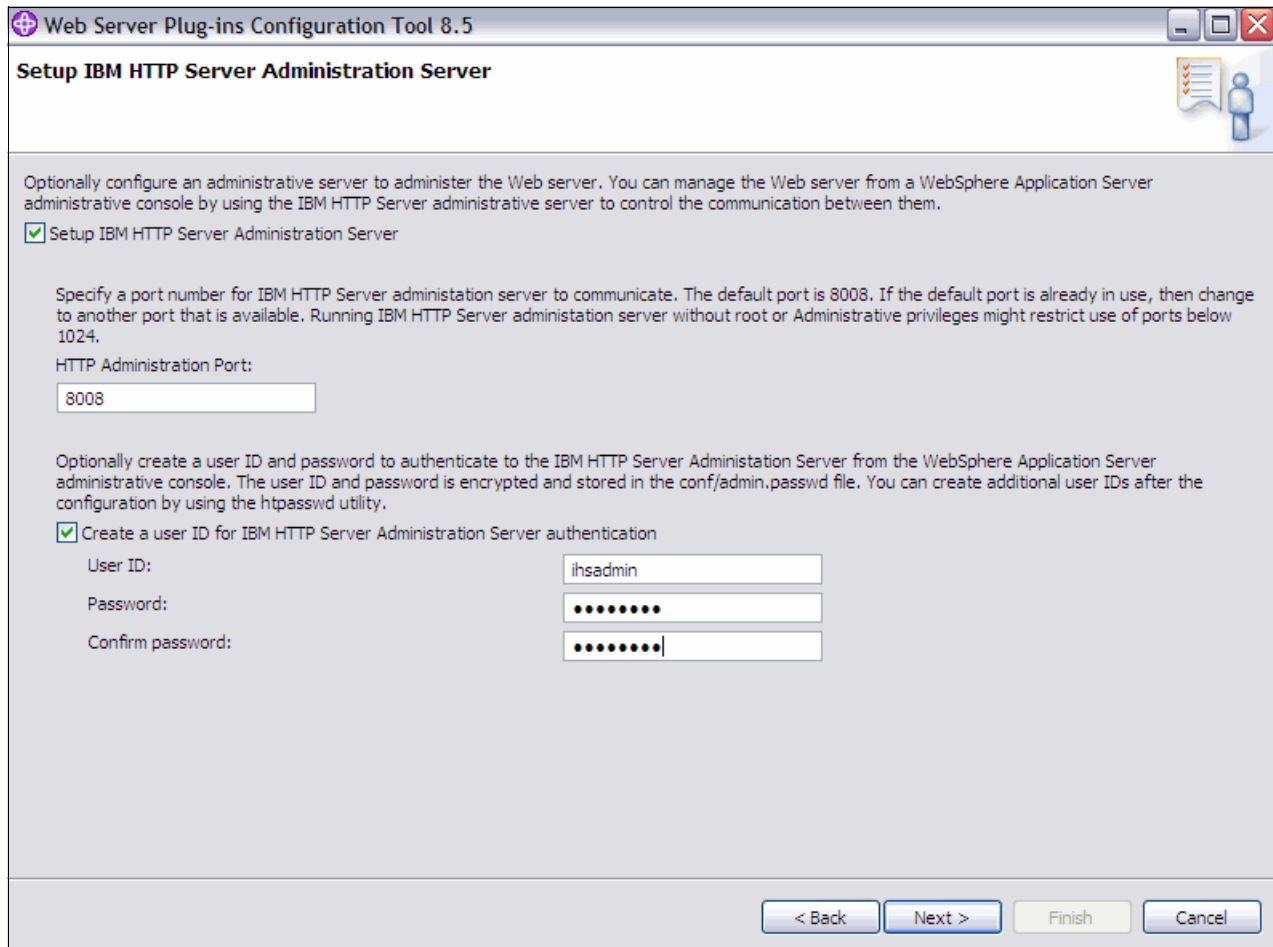


Figure 12-13 IBM HTTP Server administration server settings

Click **Next**.

- c. Provide a unique user ID and group for web server administration. In Figure 12-13 on page 434, the user ID and Group is ihs.

Optionally create a user ID and password to authenticate to the IBM HTTP Server Administration Server from the WebSphere Application Server administrative console. The user ID and password is encrypted and stored in the conf/admin.passwd file. You can create additional user IDs after the configuration by using the htpasswd utility.

Create a user ID for IBM HTTP Server Administration Server authentication

User ID:

Password:

Confirm password:

Figure 12-14 IBM HTTP Server administration server setup

Click **Next**.

8. Specify a unique name for the web server definition, as shown in Figure 12-15. Click **Next**.

Web Server Plug-ins Configuration Tool 8.5

Web Server Definition Name

Use a Web server definition to manage a Web server through the WebSphere Application Server administrative console or the wsadmin tool. The definition name must be unique because this name is used to identify this Web server in the administrative console.

Specify a unique Web server definition name:

The Web server definition name must not be empty and it must not contain the following special characters or space:

/ \ * , ; = + ? | < > & % ' " [] > # \$ ^ { }

Note: a period(.) is not valid if it is the first character.

< Back Next > Finish Cancel

Figure 12-15 Web server definition name

9. Select the **remote scenario configuration**. Provide a host name or IP address of the application server or deployment manager, as shown in Figure 12-16 on page 436.

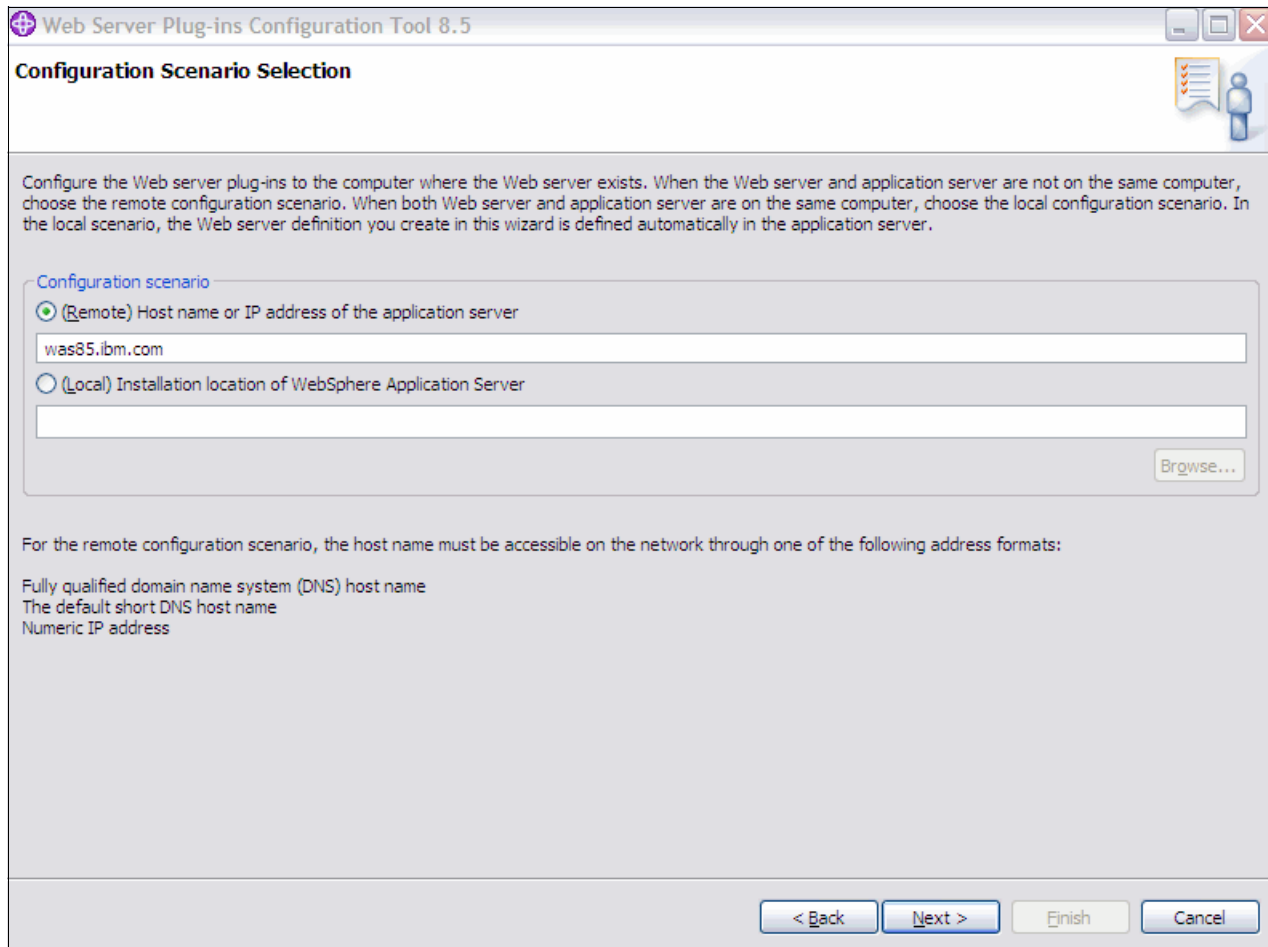


Figure 12-16 Configuration scenario selection

Click **Next**.

10. On the Summary window, review the settings, and click **Configure**.

11. When the configuration is complete, clear the **Launch the plug-in configuration roadmap** check box, and click **Finish**. Also note the manual configuration step that must be performed, as shown in Figure 12-17.

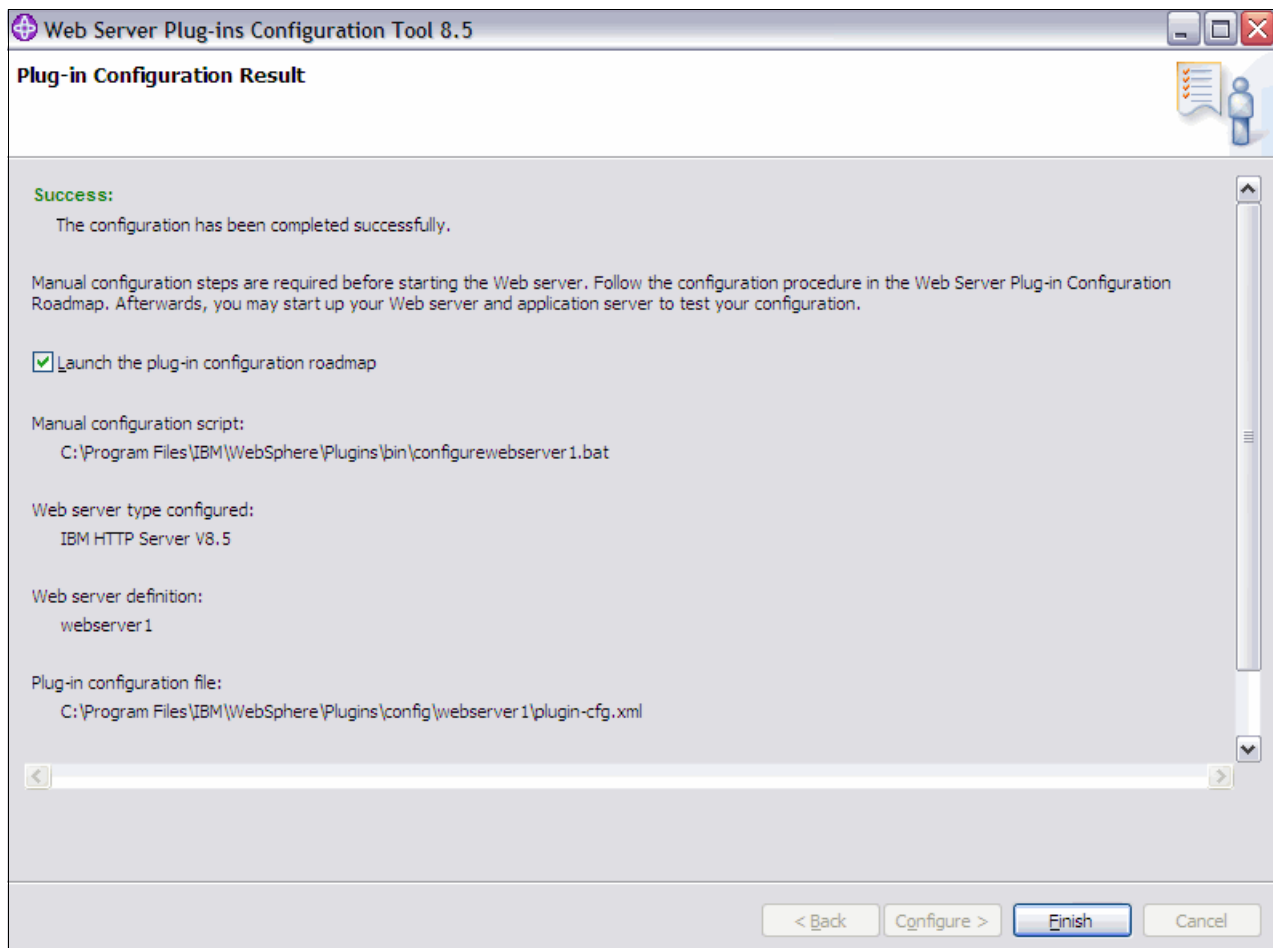


Figure 12-17 Configuration results

12. You can see the configuration file in the WebSphere Customization Toolbox. Exit the WebSphere Customization Toolbox. When the WebSphere Customization Toolbox GUI is used for the plug-in configuration, the selections made are saved and are available in a response file. See Figure 12-18.

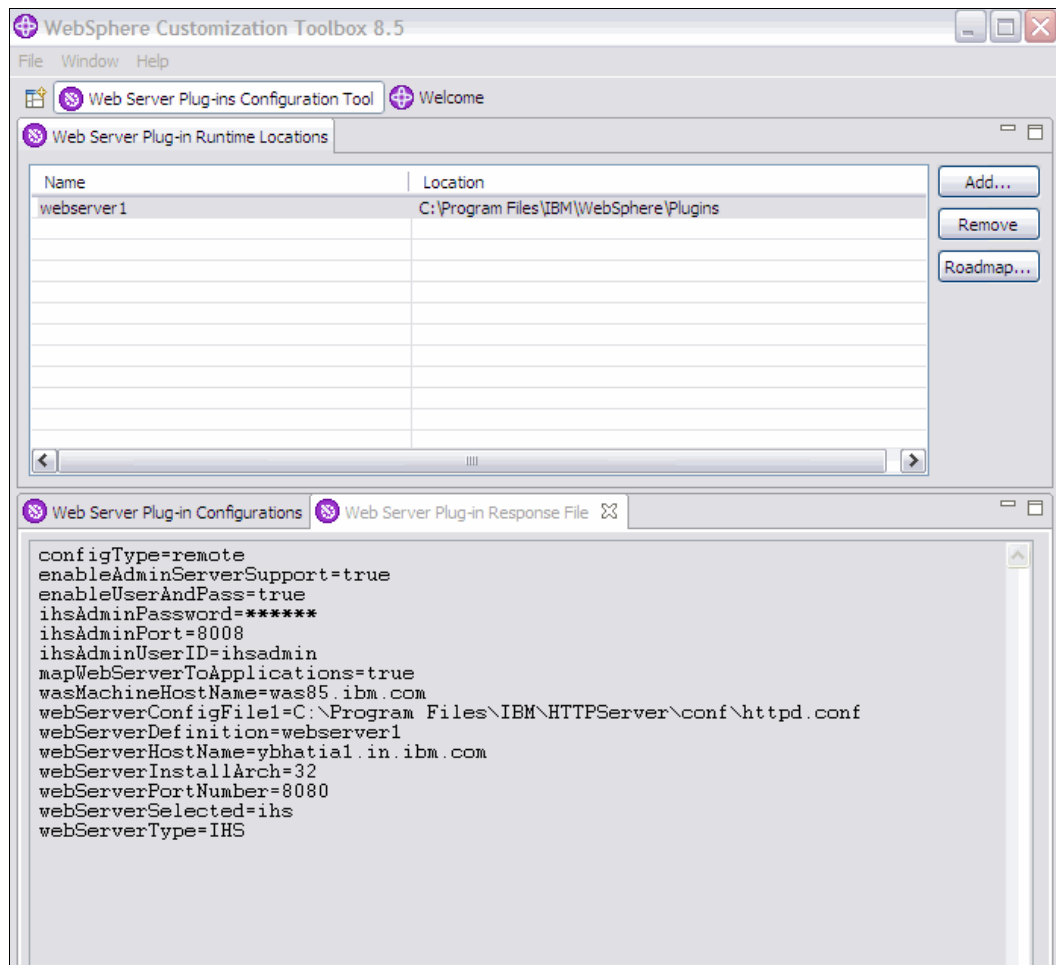


Figure 12-18 Web server plug-in response file

Alternative: An alternative to using the WebSphere Customization Toolbox GUI is to use the WebSphere Customization Toolbox command-line utility, which can be used with a response file to configure a web server.

For more information about the WebSphere Customization Toolbox command-line utility, refer to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/tins_pctcl_using.html

13. The Web Server Plug-ins Configuration Tool creates the `configureweb_server_name` script in the `plugins_root/bin/` directory on the machine with the web server. Examine the script and make any needed changes. You might need to compensate for file encoding differences to prevent script failure.

The Web Server Plug-ins Configuration Tool also creates the `plugin-cfg.xml` file in the `plugins_root/config/web_server_name` directory.

14. Copy the `configureweb_server_name` script in the `plugins_root/bin/` directory to `profile_root/Dmgr/bin`.

Note: If one platform is a system, such as AIX or Linux, and the other is a Windows platform, copy the script from the `plugins_root/bin/crossPlatformScripts` directory.

15. Run the `configureweb_server_name` script. If administrative security is enabled, provide the `-username` and `-password` arguments or wait until prompted for the credentials.

16. Log in to the administrative console, and verify the configuration. Click **System administration** → **Nodes**. See Figure 12-19.

Nodes

Use this page to manage nodes in the application server environment. A node corresponds to a physical computer system with a distinct IP host address. The following table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell and to this list by clicking Add Node.

☏ Preferences

Buttons: Add Node, Remove Node, Force Delete, Synchronize, Full Resynchronize, Stop

| Select | Name | Host Name | Version | Discovery Protocol | Status |
|---|--|----------------------------------|----------------|--------------------|--------|
| You can administer the following resources: | | | | | |
| <input type="checkbox"/> | IBM-WAS85CellManager01 | IBM-981CA4LCGQ6.itso.ral.ibm.com | ND 8.5.0.0 | TCP | ↔ |
| <input type="checkbox"/> | IBM-WAS85Node02 | IBM-981CA4LCGQ6.itso.ral.ibm.com | ND 8.5.0.0 | TCP | ? |
| <input type="checkbox"/> | ihsnode | was85node01 | Not applicable | TCP | |
| Total 3 | | | | | |

Figure 12-19 Nodes listing

12.4 Working with web servers and plug-ins

The introduction of web server definitions to the WebSphere Application Server administrative tools provides the following administrative features:

- ▶ Defining nodes (distributed server environment)
- ▶ Defining and modifying web servers
- ▶ Checking the status of a web server
- ▶ Starting and stopping IBM HTTP Servers
- ▶ Administering IBM HTTP Servers
- ▶ Viewing or modifying the web server configuration file
- ▶ Mapping modules to servers

12.4.1 Manually defining nodes and web servers

A managed node is added to the cell as part of the process when you federate an application server profile or custom profile to the cell. An unmanaged node, however, is not created using a profile. As you have seen, the web server definition script created by the Web Server Plug-ins Configuration Tool defines an unmanaged node for a web server.

However, there might be times when you need to define or update the definitions using the administrative console.

Adding an unmanaged node to the cell

To add an unmanaged node using the administrative console:

1. In the console navigation tree, click **System administration** → **Nodes**.
2. Click **Add Node**.
3. Select **Unmanaged node**. See Figure 12-20.

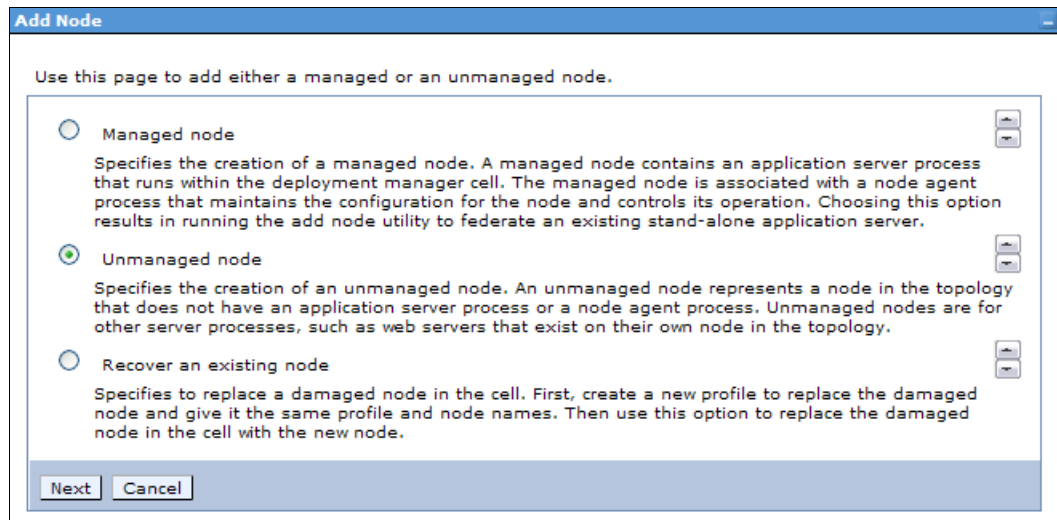


Figure 12-20 Add node selection window

4. Click **Next**.
5. Enter the following values in the General Properties window. See Figure 12-21 on page 441.
 - a. **Name:** Type a logical name for the node. The name must be unique within the cell. A node name usually is identical to the host name for the computer. However, you can make the node name different than the host name.
 - b. **Host Name:** Enter the host name of the unmanaged node that is added to the configuration.
 - c. **Platform Type:** Select the operating system on which the unmanaged node runs. Valid options are:
 - Windows
 - AIX
 - HP-UX
 - Solaris
 - Linux
 - IBM OS/400®
 - z/OS

Nodes > New...

Use this page to view or change the configuration for an unmanaged node. An unmanaged node is a node defined in the cell topology that does not have a node agent running to manage the process. Unmanaged nodes are typically used to manage web servers.

Configuration

General Properties

* Name
ihsNode02

* Host Name
was8.RHEL56.ibm.com

* Platform Type
Linux

The additional properties will not be available until the general properties for this item are applied or saved.

Additional Properties

- Custom Properties

Apply OK Reset Cancel

Figure 12-21 General properties for an unmanaged node

6. Click **OK**. The node is added, and the name is displayed in the collection on the Nodes page. See Figure 12-22.

Nodes

Use this page to manage nodes in the application server environment. A node corresponds to a physical computer system with a distinct IP host address. The following table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell and to this list by clicking Add Node.

Preferences

Add Node Remove Node Force Delete Synchronize Full Resynchronize Stop

| Select | Name | Host Name | Version | Discovery Protocol | Status |
|--------------------------|-----------------------------------|---------------------|----------------|--------------------|--------|
| <input type="checkbox"/> | ihsNode01 | was8.RHEL56.ibm.com | Not applicable | TCP | |
| <input type="checkbox"/> | ihsNode02 | was8.RHEL56.ibm.com | Not applicable | TCP | |
| | was8CellManager01 | was8.RHEL56.ibm.com | ND 8.0.0.0 | TCP | ↔ |
| <input type="checkbox"/> | was8Node01 | was8.RHEL56.ibm.com | ND 8.0.0.0 | TCP | ↔ |
| <input type="checkbox"/> | was8Node02 | was8.RHEL56.ibm.com | ND 8.0.0.0 | TCP | ↔ |
| Total 5 | | | | | |

Figure 12-22 Nodes in a cell

Adding a web server

After the node for the web server is defined, you can add the web server definition. To add a web server definition, complete the following steps:

1. Click **Servers** → **Server Types** → **Web servers**.
2. Click **New...**
3. Select the node, and enter the web server name. Using the drop-down menu, select the **web server type**. See Figure 12-23 on page 442. Click **Next**.

Use this page to create a new web server.

→ **Step 1: Select a node for the Web server and select the Web server type**

Step 2: Select a Web server template

Step 3: Enter the properties for the new Web server

Step 4: Confirm new Web server

Select a node for the Web server and select the Web server type

Select a node that corresponds to the Web server you want to add.

Select node

* Server name

* Type

Figure 12-23 Defining a web server

- Select a template. Initially, this template is the one supplied with WebSphere that is specific to the web server type. After you define a web server, you can make it a template for future use. See Figure 12-24.

Step 1: Select a node for the Web server and select the Web server type

→ **Step 2: Select a Web server template**

Step 3: Enter the properties for the new Web server

Step 4: Confirm new Web server

Select a Web server template

Select the template that corresponds to the server that you want to create.

| Select | Template Name | Type | Description |
|----------------------------------|---------------|--------|-----------------------------|
| <input checked="" type="radio"/> | IHS | System | The IHS Web Server Template |

Figure 12-24 Defining a web server: Template

Click **Next**.

- Enter the properties for the web server. You also need to enter the parameters required for remote administration. See Figure 12-25 on page 443.

Step 1: Select a node for the Web server and select the Web server type

Step 2: Select a Web server template

→ Step 3: Enter the properties for the new Web server

Step 4: Confirm new Web server

Enter the properties for the new Web server

Enter the Web server properties.

- * Port: 80
- * Web server installation location: /opt/IBM/HTTPServer
- * Plug-in installation location: /opt/IBM/HTTPServer/Plugins

Application mapping to the Web server: All

Enter the IBM Administration Server properties.

- * Administration Server Port: 8008
- * Username: ihsadmin
- * Password: [masked]
- * Confirm password: [masked]

Use SSL

Previous Next Cancel

Figure 12-25 Defining a web server: Properties

- Review the options, and click **Finish**, as shown in Figure 12-26.

Confirm new Web server

The following is a summary of your selections. Click the Finish button to complete the Web server creation. If there are settings you wish to change, click on Previous button to review the server settings.

Summary of actions:


- New Web server entry "webserv2" will be created on node "ihsNode02"
- Platform Type "Linux"
- Web server installation root "/opt/IBM/HTTPServer"
- Plug-in installation root "/opt/IBM/HTTPServer/Plugins".

Previous Finish Cancel

Figure 12-26 Defining a web server: Confirmation

12.4.2 Viewing the status of a web server

The web server status is reflected in the administrative console. To view web servers and their statuses:

- Click **Servers** → **Web servers**. If a web server is started or stopped using a native command, you might need to refresh the view by clicking the  icon to see the new status, as shown in Figure 12-27 on page 444.

Web servers
Use this page to view a list of the installed web servers.

⊕ Preferences

Generate Plug-in Propagate Plug-in New... Delete Templates... Start Stop Terminate

| Select | Name | Web server Type | Node | Host Name | Version | Status |
|---|-----------------------------|-----------------|-----------|---------------------|----------------|--------|
| You can administer the following resources: | | | | | | |
| <input type="checkbox"/> | webservice1 | IBM HTTP Server | ihsNode01 | was8.RHEL56.ibm.com | Not applicable | ➔ |
| <input type="checkbox"/> | webservice2 | IBM HTTP Server | ihsNode02 | was8.RHEL56.ibm.com | Not applicable | ➔ |
| Total 2 | | | | | | |

Figure 12-27 Web server status

WebSphere Application Server reports server status using the web server host name and port that you defined. This is normally port 80. Do not use the remote administration port. If Use secure protocol is defined, SSL is used. See Figure 12-28 on page 446.

12.4.3 Starting and stopping a web server

You can start or stop a web server using either the administrative console or a command window.

From the administrative console

You can start or stop the following web servers from the WebSphere administrative console:

- ▶ All web servers on a managed node
The node agent is used to start or stop the web server.
- ▶ IBM HTTP Server on an unmanaged node
The IBM HTTP Server administration must be running on the web server node.

To start or stop a web server from the administrative console:

1. Click **Servers** → **Web servers**.
2. Select the check box to the left of each web server that you want to start or stop.
3. Click **Start** or **Stop**.

If you have problems starting or stopping an IBM HTTP Server, check the WebSphere console logs (trace) and, if using the IBM HTTP administration server, check the admin_error.log file.

If you have problems starting and stopping IBM HTTP Server on a managed node using the node agent, you can try to start and stop the server by setting up the managed profile. After setting up the profile, issue the `startserver <IBM HTTP Server> -nowait -trace` command and check the startServer.log file for the IBM HTTP Server specified.

From a command window

You can also use the native startup or shutdown procedures for the supported web server. From a command window, change to the directory of your IBM HTTP Server installed image or to the installed image of a supported web server.

You can use one of the following two ways to start or stop the web server:

- ▶ To start or stop the IBM HTTP Server for UNIX platforms, enter one of the following commands at a command prompt:
 - # <ihs_install>/bin/apachectl start
 - # <ihs_install>/bin/apachectl stop
- ▶ To start or stop the IBM HTTP Server on a Windows platform, select the **IBM HTTP Server 8.0** service from the Services window, and invoke the appropriate action.

Note: When the web server is started or stopped with the native methods, the web server status on the web servers page of the administrative console is updated accordingly.

12.4.4 IBM HTTP Server remote administration

You can administer and configure IBM HTTP Server V8.0 using the WebSphere administrative console. On a managed node, administration is performed using the node agent. This is true of all web server types. However, unlike other web servers, administration is possible for an IBM HTTP Server installed on an unmanaged node. In this case, administration is done through the IBM HTTP administration server. This server must be configured and running. Administration is limited to generation and propagation of the plug-in configuration file.

Remote administration set up

For the administrative console to access the IBM HTTP administration server, you must define a valid user ID and password to access the IBM HTTP Server administration server. The user ID and password are stored in the web server's IBM HTTP Server administration server properties.

You can update your IBM HTTP Server administration server properties in the web server definition through the Remote Web server management properties window of the administrative console. Complete the following steps to set or change these properties:

1. Click **Servers** → **Web servers**.
2. Select the name of the web server.
3. In the Additional Properties section, click **Remote Web server management**.
4. Enter the remote web server management information, as shown in Figure 12-28 on page 446.

[Web servers](#) > [webserver1](#) > **Remote Web server management**

Use this page to configure the IBM(R) HTTP Server administration server for a web server. These properties are required for a web server that is not installed on the same machine as the WebSphere(R) application server.

Configuration

Remote Web server management

* Port
8008

* Username
ihsadmin

* Password

Use SSL

Apply OK Reset Cancel

Figure 12-28 IBM HTTP Server remote management properties

- a. Enter the port number for the IBM HTTP Server administration server. The default is 8008.
 - b. Enter a user ID and password that are defined to the IBM HTTP administration server. The IBM HTTP administration server user ID and password are not verified until you attempt to connect.
 - c. Select the **Use SSL** option, if the port is secure. The default is not set.
5. Click **OK**, and save the configuration.

Setting the user ID and password in the IBM HTTP administration server: The IBM HTTP administration server is set, by default, to refer to the following file to get the user ID and passwords to use for authentication:

```
<ihs_install>/conf/admin.passwd
```

To initialize this file with a user ID, use the **htpasswd** command. Example 12-1 initializes the file with the user ID webadmin.

Example 12-1 Authentication with user ID

```
/opt/IBM HTTP Server/bin>./htpasswd.sh /opt/IBM HTTP Server/conf/admin.passwd webadmin
```

```
New password: *****
```

```
Re-type new password: *****
```

```
Adding password for user webadmin
```

When you are managing an IBM HTTP Server using the WebSphere administrative console, you must ensure that the following conditions are met:

- ▶ Verify that the IBM HTTP Server administration server is running.
- ▶ Verify that the web server host name and port defined in the administrative console match the IBM HTTP Server administration host name and port.

- ▶ Verify that the firewall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere administrative console.
- ▶ Verify that the user ID and password specified in the WebSphere administrative console under the Remote Web server management is an authorized combination for IBM HTTP Server administration.
- ▶ If you are trying to connect securely, verify that you exported the IBM HTTP Server administration server keydb personal certificate into the WebSphere key database as a signer certificate. This key database is specified by the `com.ibm.ssl.trustStore` in the `sas.client.props` file in the profile your console is running. This setup is mainly for self-signed certificates.
- ▶ Verify that the IBM HTTP Server `admin_error.log` file and the WebSphere Application Server logs (`trace.log`) do not contain any errors.

Hints and tips

The following list describes hints and tips about starting, stopping, and obtaining the status for the IBM HTTP Server using the WebSphere administrative console:

Viewing or modifying the web server configuration file

The Web Server Plug-ins Configuration Tool automatically configures the web server configuration file with the information necessary to use the plug-in. For example, among the updates made are the lines in Example 12-2 at the bottom of the `httpd.conf` file.

Example 12-2 Plug-in configuration location defined in httpd.conf

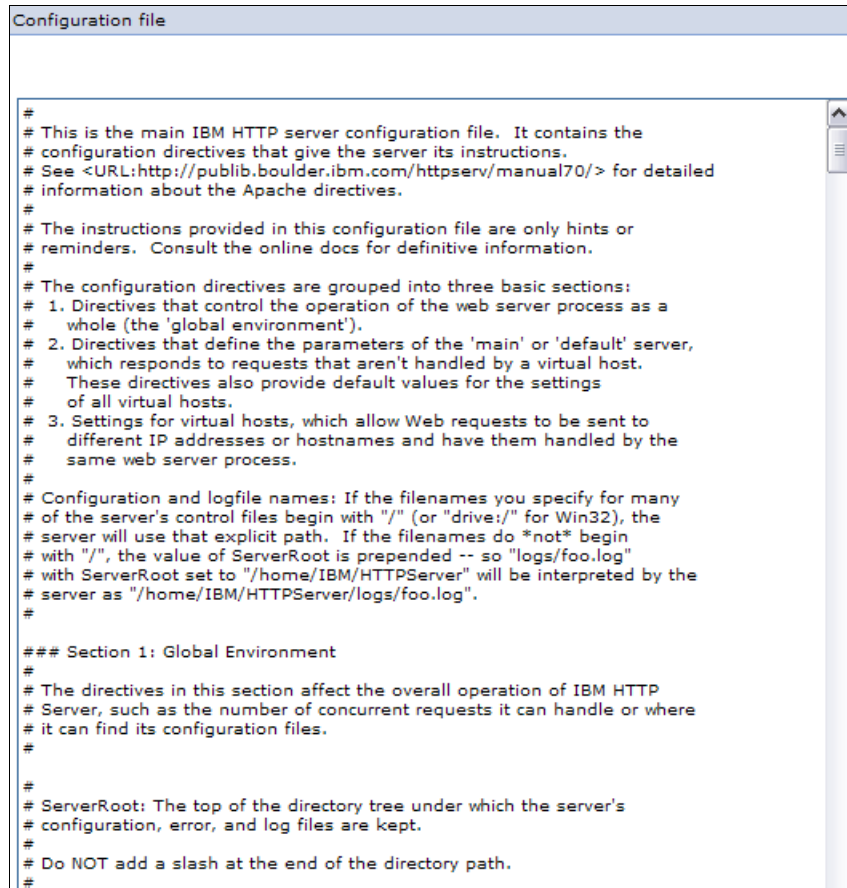
```
LoadModule was_ap22_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap22_http.so
WebSpherePluginConfig /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

Note that the location that the web server expects to find the plug-in configuration file is specified in these lines. When you generate the web server plug-in configuration from the managed web server, you need to propagate or copy the generated file to this location.

The web server configuration file is a text file and can be modified or viewed manually with a text editor. You can also view or modify this file using the administrative console.

To view or modify the contents of the web server configuration file in your web browser, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Select the name of the web server.
3. In the Additional Properties section, click **Configuration File**. See Figure 12-29 on page 448.



```
Configuration file

#
# This is the main IBM HTTP server configuration file. It contains the
# configuration directives that give the server its instructions.
# See <URL:http://publib.boulder.ibm.com/htpserv/manual70/> for detailed
# information about the Apache directives.
#
# The instructions provided in this configuration file are only hints or
# reminders. Consult the online docs for definitive information.
#
# The configuration directives are grouped into three basic sections:
# 1. Directives that control the operation of the web server process as a
# whole (the 'global environment').
# 2. Directives that define the parameters of the 'main' or 'default' server,
# which responds to requests that aren't handled by a virtual host.
# These directives also provide default values for the settings
# of all virtual hosts.
# 3. Settings for virtual hosts, which allow Web requests to be sent to
# different IP addresses or hostnames and have them handled by the
# same web server process.
#
# Configuration and logfile names: If the filenames you specify for many
# of the server's control files begin with "/" (or "drive:" for Win32), the
# server will use that explicit path. If the filenames do *not* begin
# with "/", the value of ServerRoot is prepended -- so "logs/foo.log"
# with ServerRoot set to "/home/IBM/HTTPServer" will be interpreted by the
# server as "/home/IBM/HTTPServer/logs/foo.log".
#

### Section 1: Global Environment
#
# The directives in this section affect the overall operation of IBM HTTP
# Server, such as the number of concurrent requests it can handle or where
# it can find its configuration files.
#
#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# Do NOT add a slash at the end of the directory path.
#
```

Figure 12-29 IBM HTTP Server configuration file httpd.conf

4. Type your changes directly into the window, and click **OK**. Save the changes.

Note: If you made changes to the configuration file, you must restart your web server for the changes to take effect.

Viewing web server logs

With remote administration, you can also view the IBM HTTP Server access log and error log. To view the logs:

1. Click **Servers** → **Web servers**.
2. Select the name of the web server.
3. In the Additional Properties section, click **Log file**.
4. Click the **Runtime** tab. See Figure 12-30 on page 449.

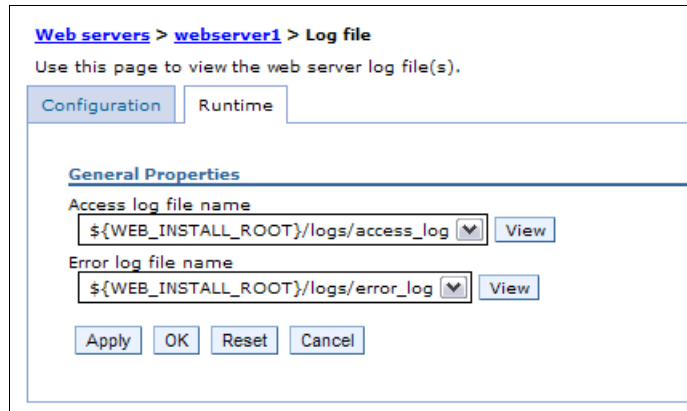


Figure 12-30 Web server Runtime tab for logs

5. Beside the log you want to view, click **View**. See Figure 12-31.



Figure 12-31 Viewing the error log

12.4.5 Mapping modules to servers

Each module of an application is mapped to one or more target servers. The target server can be an application server, cluster of application servers, or a web server. Modules can be installed on the same application server or dispersed among several application servers. Web servers, specified as targets, have routing information for the application generated in the plug-in configuration file for the web server.

This mapping takes place during application deployment. After an application is deployed, you can view or change these mappings. To check or change the mappings, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click the application for which you want to review the mapping.
3. Click **Manage modules** in the Modules section.
4. The Select server window opens, as shown in Figure 12-32. Examine the list of mappings. Ensure that each Module entry is mapped to all targets identified under Server.

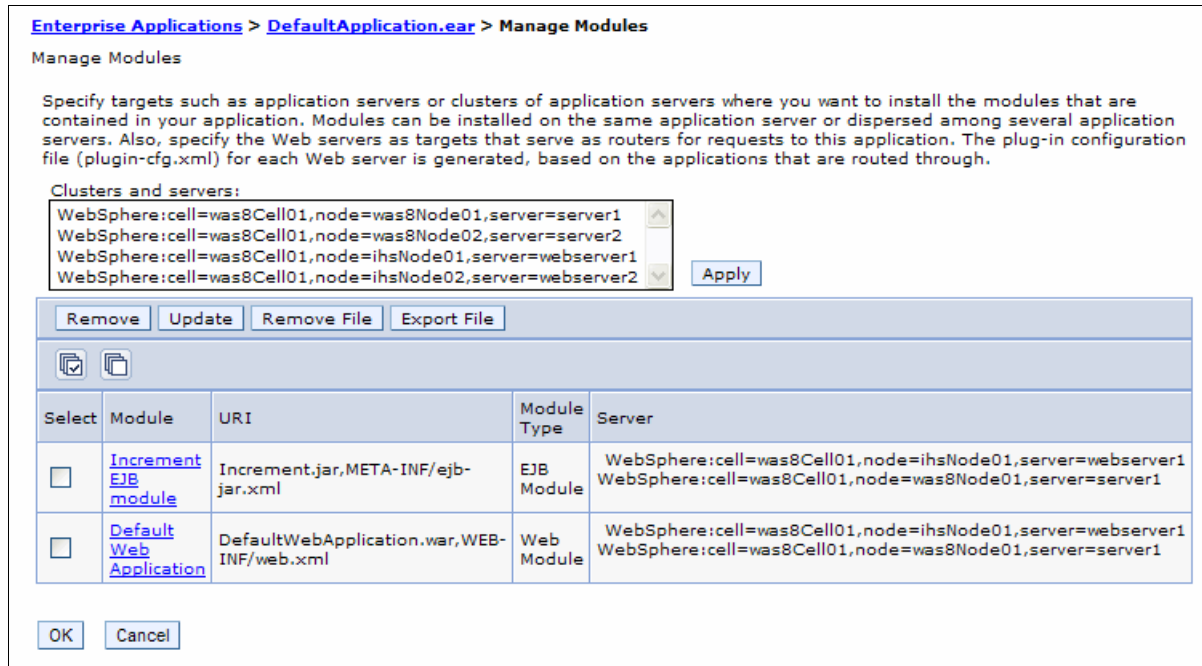


Figure 12-32 Application module mappings

5. To change a mapping:
 - a. Select each module that you want mapped to the same targets by selecting the box to the left of the desired module.
 - b. From the Clusters and Servers list, select one or more targets. Use the Ctrl key to select multiple targets. For example, to have a web server serve your application, use the Ctrl key to select an application server and the web server together.
6. Click **Apply**.
7. Repeat step 5 until each module maps to the desired targets.
8. Click **OK**, and save your changes.
9. Regenerate and propagate the plug-in configuration, if it is not done automatically.

After you define at least one web server, you must specify a web server as a deployment target whenever you deploy a web application. If the web server plug-in configuration service is enabled, a web server plug-in's configuration file is automatically regenerated whenever a new application is associated with that web server.

12.5 Working with the plug-in configuration file

The plug-in configuration file (plugin-cfg.xml) contains routing information for all applications mapped to the web server. This file is read by a binary plug-in module loaded in the web server. An example of a binary plug-in module is the mod_ibm_app_server_http.dll file for IBM HTTP Server on the Windows platform.

The binary plug-in module does not change. However, the plug-in configuration file for the binary module needs to be regenerated and propagated to the web server whenever a change is made to the configuration of applications mapped to the web server. The binary

module reads the XML file to adjust settings and to locate deployed applications for the web server. Example 12-3 shows an excerpt from a generated plug-in configuration file.

Example 12-3 An excerpt from the plugin-cfg.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?><!--HTTP server plugin config file for
the webserver ITSOCell.wan.webserver1 generated on 2004.10.29 at 03:32:12 PM
BST-->
<Config ASDisableNagle="false" AcceptAllContent="false"
AppServerPortPreference="HostHeader" ChunkedResponse="false" FIPSEnable="false"
FailoverToNext="false" HTTPMaxHeaders="300" IISDisableNagle="false"
IISPluginPriority="High" IgnoreDNSFailures="false"
OS400ConvertQueryStringToJobsCCSID="false" RefreshInterval="60"
ResponseChunkSize="64" SSLConsolidate="true" TrustedProxyEnable="false"
VHostMatchingCompat="false">
  <Log LogLevel="Error"
Name="c:\opt\WebSphere\Plugins\logs\webserver1\http_plugin.log"/>
  <Property Name="ESIEnable" Value="true"/>
  <Property Name="ESIMaxCacheSize" Value="1024"/>
  <Property Name="ESIInvalidationMonitor" Value="false"/>
  <Property Name="ESIEnableToPassCookies" Value="false"/>
  <Property Name="PluginInstallRoot" Value="c:\opt\WebSphere\Plugins\*" />
<VirtualHostGroup Name="default_host">
  <VirtualHost Name="*:9080"/>
  <VirtualHost Name="*:80"/>
  <VirtualHost Name="*:9443"/>
</VirtualHostGroup>

  <ServerCluster CloneSeparatorChange="false"GetDWLMTable="false"
IgnoreAffinityRequests="true" LoadBalance="Round Robin"
Name="server1_NodeA_Cluster" PostBufferSize="64" PostSizeLimit="-1"
RemoveSpecialHeaders="true" RetryInterval="60">
  <Server ConnectTimeout="0" ExtendedHandshake="false" MaxConnections="-1"
Name="NodeA_server1" WaitForContinue="false">
    <Transport Hostname="wan" Port="9080" Protocol="http"/>
    <Transport Hostname="wan" Port="9443" Protocol="https">
      <Property Name="keyring"
Value="c:\opt\WebSphere\Plugins\etc\plugin-key.kdb"/>
      <Property Name="stashfile"
Value="c:\opt\WebSphere\Plugins\etc\plugin-key.sth"/>
    </Transport>
  </Server>
</ServerCluster>

  <UriGroup Name="default_host_server1_NodeA_Cluster_URIs">
    <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/snoop/*"/>
    <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
Name="/hello"/>
  </UriGroup>
<Route ServerCluster="server1_NodeA_Cluster"
UriGroup="default_host_server1_NodeA_Cluster_URIs"
VirtualHostGroup="default_host"/>
</Config>
```

The specific values for the UriGroup Name and AffinityCookie attributes depend on how you assembled your application. Keep the following points in mind when you assemble your application:

- ▶ If you specify **File Serving Enabled**, only a wildcard URI is generated, regardless of any explicit servlet mappings.
- ▶ If you specify **Serve servlets by class name**, a URI of the form URI name = <webappuri>/servlet/ is generated.

Both of these options apply for both the Name and AffinityCookie attributes.

When the plug-in configuration file is generated, it does not include admin_host in the list of virtual hosts. See “Allowing Web servers to access the administrative console” in the information center for information about how to add it to the list at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Fae%2Ftwsv_plugin.html

12.5.1 Regenerating the plug-in configuration file

The plug-in configuration file needs to be regenerated and propagated to the web servers when there are changes to your WebSphere configuration that affect how requests are routed from the web server to the application server. The following list shows the included changes:

- ▶ Installing an application
- ▶ Creating or changing a virtual host
- ▶ Creating a new server
- ▶ Modifying HTTP transport settings
- ▶ Creating or altering a cluster

The plug-in file can be regenerated manually using the administration tools. You can also set up the plug-in properties of the web server to enable automatic generation of the file whenever a relevant configuration change is made. Refer to “Enabling automated plug-in regeneration” on page 456 for more information.

To regenerate the plug-in configuration manually, you can either use the administrative console, or you can issue the **GetPluginCfg** command.

Generating the plug-in with the administrative console

To generate or regenerate the plug-in configuration file:

1. Click **Servers** → **Web servers**.
2. Select the **check box** to the left of your web server.
3. Click **Generate Plug-in**.
4. Verify that the generation was successful by looking at the messages. A success message is accompanied with the location of the generated plug-in configuration file:

```
<profile_home>/config/cells/<cell_name>/nodes/<web_server_node>/servers/<web_server>/plugin-cfg.xml
```

See Figure 12-33 on page 453.

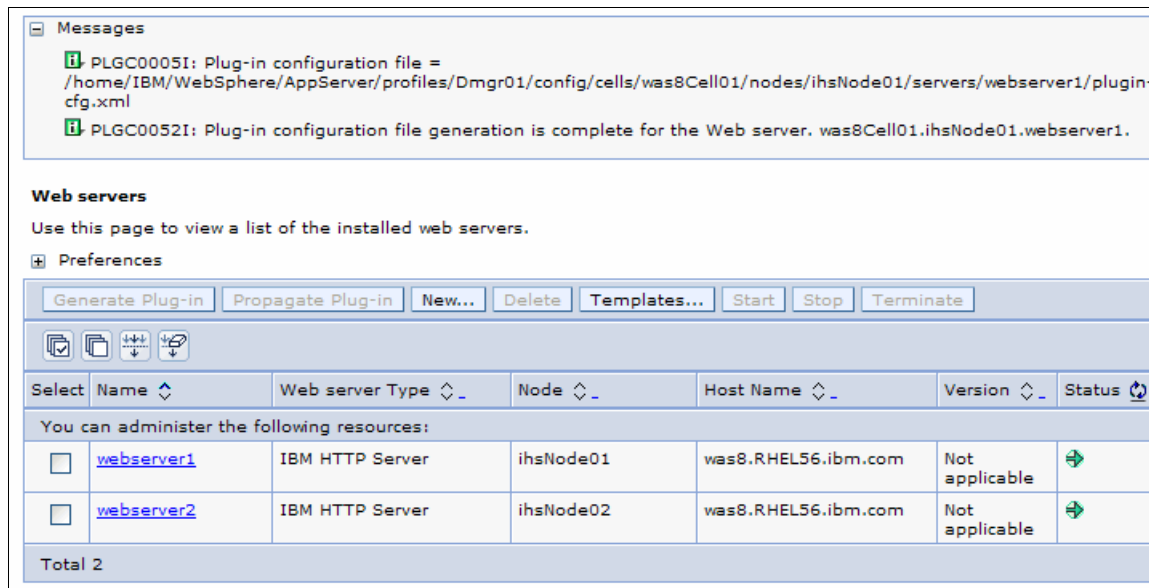


Figure 12-33 Web server definitions

- Click the name of the web server. You can view the plug-in configuration file by clicking the **View** button next to the Plug-in configuration file name on the Plug-in properties window of your web server definition, as shown in Figure 12-34. You can also open it with a text editor.

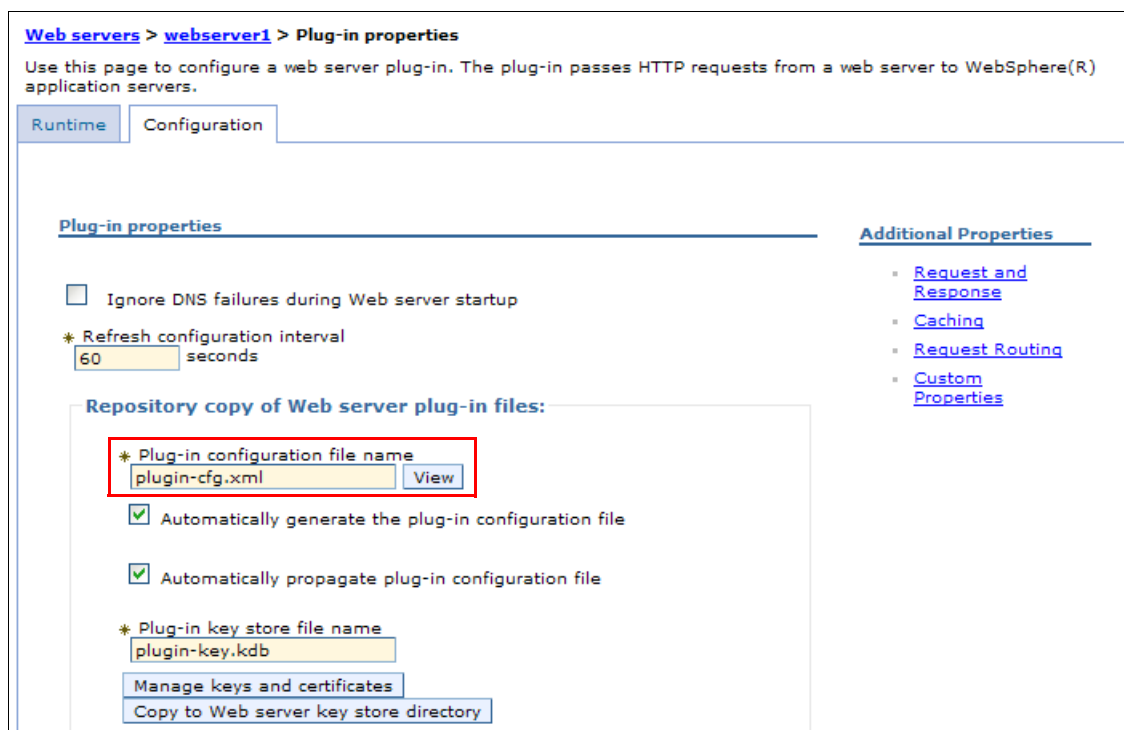


Figure 12-34 Plug-in properties

To use the new plugin-cfg.xml file, you must propagate it to the web server system. See 12.5.2, “Propagating the plug-in configuration file” on page 457 for more information.

Regenerating the plug-in with the GenPluginCfg command

The **GenPluginCfg** command is used to regenerate the plug-in configuration file. Depending on the operating platform, the command is:

- ▶ On UNIX: **GenPluginCfg.sh**
- ▶ On Windows: **GenPluginCfg.bat**

You can use the `-profileName` option to define the profile of the Application Server process in a multi-profile installation. The `-profileName` option is not required for running in a single profile environment. The default for this option is the default profile. For a distributed server environment, the default profile is the deployment manager profile.

Syntax

The **GenPluginCfg** command reads the contents of the configuration repository on the local node to generate the Web server plug-in configuration file.

The syntax of the **GenPluginCfg** command is:

```
GenPluginCfg.bat(sh) [options]
```

The options are listed in Table 12-1. All options are optional.

Table 12-1 Options for GenPluginCfg

| Option | Description |
|--|--|
| -config.root <config root> | Specifies the directory path of the particular configuration repository to be scanned. The default is the value of CONFIG_ROOT defined in the SetupCmdLine.bat(sh) script. |
| -profileName <profile> | Runs the command against this profile. If the command is run from <was_home>/bin and -profileName is not specified, the default profile is used. If it is run from <profile_home>/bin, that profile is used. |
| -cell.name <cell name> | Restricts generation to only the named cell in the configuration repository. The default is the value of WAS_CELL defined in the SetupCmdLine.bat (sh) script. |
| -node.name <node name> | Restricts generation to only the named node in the particular cell of the configuration repository. The default is the value of WAS_NODE defined in the SetupCmdLine.bat(sh) script. |
| -webserver.name <webserver1> | Required for creating plug-in configuration file for a given web server. |
| -propagate yes/no | This option applies only when the option webserver.name is specified. The default is no. |
| -propagateKeyring yes/no | This option applies only when the option webserver.name is specified. The default is no. |
| -cluster.name <cluster_name,cluster_name> ALL | Generates an optional list of clusters. It is ignored when the option webserver.name is specified. |
| -server.name <server_name, server_name> | Generates an optional list of servers. It is required for single server plug-in generation. It is ignored when the option webserver.name is specified. |
| -output.file.name <filename> | Defines the path to the generated plug-in configuration file. The default is <configroot_dir>/plugin-cfg.xml file. It is ignored when the option webserver.name is specified. |
| -destination.root <root> | Specifies the installation root of the machine on which the configuration is used. It is ignored when the option webserver.name is specified. |
| -destination.operating.system windows/unix | Specifies the operating system of the machine on which the configuration is used. It is ignored when the option webserver.name is specified. |

| Option | Description |
|-------------------|--|
| -force yes | An optional argument to overwrite the existing configuration file. The default is no. |
| -debug <yes no> | Enables or disables the output of debugging messages. The default is no (debugging is disabled). |
| -help or -? | Prints the command syntax. |

Examples

To generate a plug-in configuration for all of the clusters in a cell, run:

```
GenPluginCfg -cell.name NetworkDeploymentCell
```

To generate a plug-in configuration for a single server, run:

```
GenPluginCfg -cell.name BaseApplicationServerCell -node.name appServerNode
-server.name appServerName
```

To generate a plug-in configuration file for a web server, run:

```
GenPluginCfg -cell.name BaseApplicationServerCell -node.name webserverNode
-webserver.name webserverName
```

When this command is issued without the option `-webserver.name webserverName`, the plug-in configuration file is generated based on the topology.

Enabling automated plug-in regeneration

The web server plug-in configuration service, by default, regenerates the `plugin-cfg.xml` file automatically. You can view or change the configuration settings for the web server plug-in configuration service. See Figure 12-33 on page 453. To view or change the plug-in generation property, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Click the name of your web server to open the configuration page.
3. In the Additional Properties section, select **Plug-in properties**.
4. View or change the **Automatically generate the plug-in configuration file** option, as shown in Figure 12-35 on page 457.

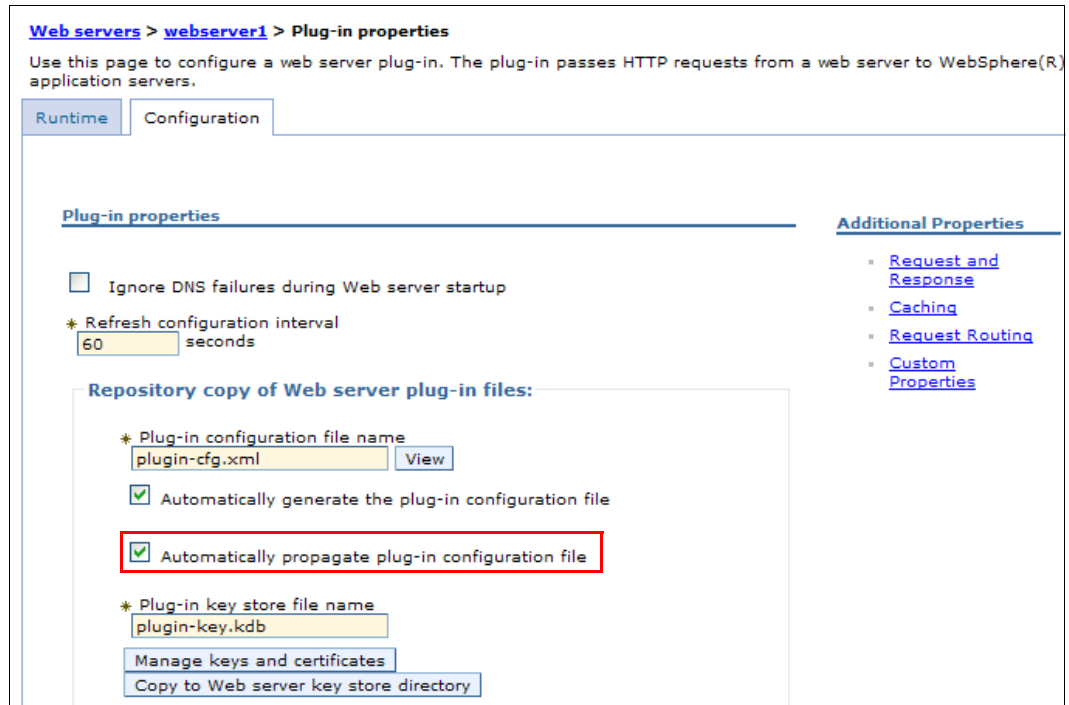


Figure 12-35 Plug-in properties

When selected, the web server plug-in configuration service automatically generates the plug-in configuration file whenever the web server environment changes. For example, the plug-in configuration file is regenerated whenever one of the following activities occurs:

- A new application is deployed on an associated application server.
- The web server definition is saved.
- An application is removed from an associated application server.
- A new virtual host is defined.

Whenever a virtual host definition is updated, the plug-in configuration file is automatically regenerated for all of the web servers.

By default, this option is automatically selected. If you clear the check from the box, you must manually generate a plug-in configuration file for this web server.

12.5.2 Propagating the plug-in configuration file

After a plug-in configuration file is regenerated, it needs to be propagated to the web server.

The configuration service can automatically propagate the plugin-cfg.xml file to a web server machine if it is configured on a managed node, and to an IBM HTTP Server if it is configured on an unmanaged node. For other scenarios, you must manually copy the file to the web server machines.

You can manually propagate the file by copying it from the application server machine to the web server machine, or you can do it from the administrative console.

From a command window

To copy the file from one machine to another machine:

1. Copy the file from its original location, for example:

```
<profile_home>/config/cells/<cell_name>/nodes/<web_server_node>/servers/<web_server>/plugin-cfg.xml
```

2. Place the copy in this directory on the remote web server machine:

```
<plug-ins_home>/config/<web_server>
```

From the administrative console

To propagate the plug-in configuration manually from the administrative console:

1. Click **Servers** → **Web servers**.
2. Select the **check box** to the left of your web server.
3. Click **Propagate plug-in**.
4. Verify that the propagation was successful by looking at the messages.

If you are in doubt, check whether the plug-in configuration file was propagated to the web server plug-in location by viewing it.

Activating the new plug-in configuration

The web server binary plug-in module checks for a new configuration file every 60 seconds. You can wait for the plug-in to find the changes, or you can restart the web server to invoke the changes immediately.

Tip: If you encounter problems restarting your web server, check the http_plugin.log file in `<plug-ins_home>/config/<web_server>` for information about what portion of the plugin-cfg.xml file contains an error. The log file states the line number on which the error occurred along with other details that might help you diagnose why the web server did not start.

Enable automated plug-in propagation

The web server plug-in configuration service, by default, propagates the plugin-cfg.xml file automatically. To view or change the plug-in propagation property, complete the following steps:

1. Click **Servers** → **Web servers**.
2. Click the name of your web server.
3. In the Additional Properties sub section, click **Plug-in properties**. See Figure 12-35 on page 457 for a visual of plug-in properties and information.
4. View or change the **Automatically propagate plug-in configuration file** option.

By default, this option is automatically selected. If you clear the check box, you must manually propagate the plug-in configuration file for this web server.

To verify that the automatic propagation was successful, look in the SystemOut.log file of the deployment manager for details.

12.5.3 Modifying the plug-in request routing options

You can specify the load balancing option that the plug-in uses when sending requests to the various application servers associated with that web server.

To view or modify the Request routing:

1. Click **Servers** → **Web Servers**.
2. Click the name of your web server.
3. In the Additional Properties section, click **Plug-in properties**.
4. In the Additional Properties section, click **Request Routing**, as shown in Figure 12-36.

Web servers > webserver1 > Plug-in properties > Request routing

Use this page to configure request routing properties for a web server plug-in. These properties apply to all requests the web server routes to application servers.

Configuration

Request routing

Load balancing option
Round Robin

* Retry interval
60 seconds

Maximum size of request content

No Limit
 Set Limit

* Maximum buffer size used when reading the HTTP request content
64 KBytes

Remove special headers
 Clone separator change

Apply OK Reset Cancel

Figure 12-36 Request routing properties

a. Load balancing option

This field corresponds to the LoadBalanceWeight element in the plugin-cfg.xml file. The load balancing options are covered in detail in *WebSphere Application Server V6 Scalability and Performance Handbook*, SG24-6392. The following items are short overviews:

- Round robin (default)

When using this algorithm, the plug-in selects a cluster member at random from which to start. The first successful browser request is routed to this cluster member and then its weight is decremented by one. New browser requests are then sent round robin to the other application servers and, subsequently, the weight for each application server is decremented by one. The spreading of the load is equal between application servers until one application server reaches a weight of zero. From then on, only application servers without a weight higher than zero receive routed requests. The only exception to this pattern is when a cluster member is added or restarted.

- **Random**
Requests are passed to cluster members randomly. Weights are not taken into account as in the round robin algorithm. The only time the application servers are not chosen randomly is when there are requests with associated sessions. When the random setting is used, cluster member selection does not take into account where the last request was handled, which means that a new request can be handled by the same cluster member as the last request.
- **Retry interval**
The length of time, in seconds, that elapses from the time an application server is marked down to the time that the plug-in retries a connection.

This field corresponds to the `ServerWaitforContinue` element in the `plugin-cfg.xml` file. The default is 60 seconds.
- **Maximum size of request content**
Limits the size of the request content. If limited, this field also specifies the maximum number of bytes of request content allowed for the plug-in to attempt to send the request to an application server.

This field corresponds to the `PostSizeLimit` element in the `plugin-cfg.xml` file. When a limit is set, the plug-in fails any request that is received that is greater than the specified limit.

You can set a limit in kilobytes or no limit. The default is set to no limit for the post size.
- **Maximum buffer size used when reading HTTP request content**
The maximum buffer size in kilobytes that is used when the content of an HTTP request is read. If the application server that initially receives a request cannot process that request, the data contained in this buffer is sent to another application server in an attempt to have that application server process the request.

This field corresponds to the `PostBufferSize` element in the `plugin-cfg.xml` file. The default is 64 KB.
- **Remove special headers**
When enabled, the plug-in removes any headers from incoming requests before adding the headers that the plug-in is supposed to add before forwarding the request to an application server.

This field corresponds to the `RemoveSpecialHeaders` element in the `plugin-cfg.xml` file. The plug-in adds special headers to the request before it is forwarded to the application server. These headers store information about the request that need to be used by the application. Not removing the headers from incoming requests introduces a potential security exposure.

The default is to remove special headers.
- **Clone separator change**
When enabled, the plug-in expects the plus character (+) as the clone separator.

This field corresponds to the `ServerCloneID` element in the `plugin-cfg.xml` file. Some pervasive devices cannot handle the colon character (:) used to separate clone IDs in conjunction with session affinity. If this field is checked, you must also change the configurations of the associated application servers so that the application servers separate clone IDs with the plus character too.

Example 12-4 shows a sample execution where the following parameters were used:

| | |
|-------------------------------|---|
| -pluginInstallLocation | The directory where the plug-in code is installed |
| -pluginRuntimeLocation | The directory where the plug-in configuration is created. |
| -wasInstallLocation | The directory where the WebSphere Application Server code is installed. |

Example 12-4 Creating the plug-in configuration directory

```
/opt/zWebSphere_Plugins/V8R0/bin: >./install_plugin.sh -pluginInstallLocation
/opt/zWebSphere_Plugins/V8R0 -pluginRuntimeLocation /etc/websrv1/Plugins
-wasInstallLocation /opt/zWebSphere/V8R0
```

Using the plug-in install location /opt/zWebSphere_Plugins/V8R0.

Using the plug-in runtime location /etc/websrv1/Plugins.

Using the WAS install location /opt/zWebSphere/V8R0.

```
#####
```

```
    Show the install_plugin.sh cmdline args
```

```
    Plugin Install Location=/opt/zWebSphere_Plugins/V8R0
```

```
    Plugin Runtime Location=/etc/websrv1/Plugins
```

```
    WAS Install Location  =/opt/zWebSphere/V8R0
```

```
#####
```

```
#####
```

```
    The install_plugin.sh has finished "/etc/websrv1/Plugins" is the Plugin
runtime location
```

```
#####
```

4. Go to the bin subdirectory from the directory you created in step 3 on page 461 or from the plug-in installation directory.
5. Configure the web server instance to use the plug-in by running the ConfigureIHSPugin.sh script. Example 12-5 shows a sample execution where the following parameters were used:

| | |
|----------------------------|--|
| -plugin.home | The directory where the plug-in code is installed. |
| -plugin.config.xml | The location where the plug-in configuration file will be created. |
| -ihs.conf.file | The location of the IBM HTTP Server config file. |
| -operating.system | The operating system where there configuration is being performed. |
| -WAS.webserver name | The web server name that is defined in the WebSphere Application Server configuration. |
| -WAS.host.name | The WebSphere Application Server host name or IP address. |

Example 12-5 Configuring a web server instance to use the plug-in

```
/SYSTEM/etc/websrv1/Plugins/bin: >./ConfigureIHSPugin.sh -plugin.home
/etc/websrv1/Plugins -plugin.config.xml
```



```

chkInstall_Arch:

logBoth:
    Ÿecho" 64 bit directory was located.

updateLogFile:

terminateOnFailure:

checkPlgCfg:

logBoth:
    Ÿecho" 64 bit directory was located.

updateLogFile:

terminateOnFailure:

updatePlgCfg:

logBoth:
    Ÿecho" Installing default plugin-cfg.xml file in directory
/etc/websrv1/Plugins/config/webserver1

updateLogFile:

createCfgDir:

logBoth:
    Ÿecho" Creating directory /etc/websrv1/Plugins/config/webserver1

updateLogFile:
    Ÿmkdir" Created dir: /etc/websrv1/Plugins/config/webserver1

changeCfgDirPerms:
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1
updatePlgKeyStore:

logBoth:
    Ÿecho" Installing default keystore files in directory
/etc/websrv1/Plugins/config/webserver1

updateLogFile:
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1
    Ÿcopy" Copying 1 file to /etc/websrv1/Plugins/config/webserver1

updatePluginCfgGroupOwnership:

checkConfigFiles:

logBoth:
    Ÿecho" Located config file /etc/websrv1/conf/httpd.conf

```



```
updateLogFile:
terminateOnFailure:
chkLogDir:
logBoth:
    Ÿecho" Log directory /etc/websrv1/Plugins/logs/webserver1 does not exist
updateLogFile:
createLogDir:
logBoth:
    Ÿecho" Creating directory /etc/websrv1/Plugins/logs/webserver1
updateLogFile:
    Ÿmkdir" Created dir: /etc/websrv1/Plugins/logs/webserver1
changePluginLogFileOwner:
    Ÿtouch" Creating /etc/websrv1/Plugins/logs/webserver1/http_plugin.log
nobodyIfNotZOS:
appendIHSConfigurationFileLoadModuleEntry:
e2a_for_zos:
logBoth:
    Ÿecho" Commenting out previous LoadModule entries
updateLogFile:
logBoth:
    Ÿecho" Commenting out previous bootstrap entries
updateLogFile:
appendToGivenFileGivenString:
appendToGivenFileGivenString:
logBoth:
    Ÿecho" Appending: LoadModule was_ap22_module
/etc/websrv1/Plugins/bin/mod_was_ap22_http.soto/etc/websrv1/conf/httpd.conf
updateLogFile:
appendToGivenFileGivenString:
logBoth:
    Ÿecho" Appending: WebSpherePluginConfig
/etc/websrv1/Plugins/config/webserver1/plugin-cfg.xml to
/etc/websrv1/conf/httpd.conf
```

```
updateLogFile:
a2e_for_zos:
ConfigureIHSPlugin:
logBoth:
    "Echo" Install complete
updateLogFile:
BUILD SUCCESSFUL
Total time: 7 seconds
```

6. If you are not using the standard ports for a web server (80 and 443), change them under a virtual host. At the WebSphere Application Server console, click **Environment** → **Virtual hosts**, and define your IBM HTTP Server ports to the proper virtual host.
7. Click **Servers** → **New server**, and create the web server definition generated during step 5 on page 462.
8. Save the configuration, and restart your application server.
9. Click **Servers** → **Server Types** → **Web servers**, select your newly created web server, and click **Propagate Plug-in**.
10. Restart the web server.

Further details about Web Server Plug-Ins for IBM WebSphere Application Server for z/OS configuration are at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_plugin_ihsz.html

12.7 Troubleshooting some common errors

After completing all of the configuration steps, check that all configurations are correct and tested by accessing the application url or sample an application.

There can be many types of errors, and the following list shows the most common errors you encounter:

- ▶ Error 404 "Page cannot be found"
- ▶ Error 500 "Internal Server Error"

12.7.1 Troubleshooting Error 404

When you attempt to access a URL and receive error 404 (page cannot be found), complete the following steps:

1. Check the WebSphere administration console to verify if the virtual hosts entry has the appropriate http port (for example 8080). This can be verified by selecting **Environment** → **Virtual Hosts** → **default_host** (see Figure 12-37 on page 467).

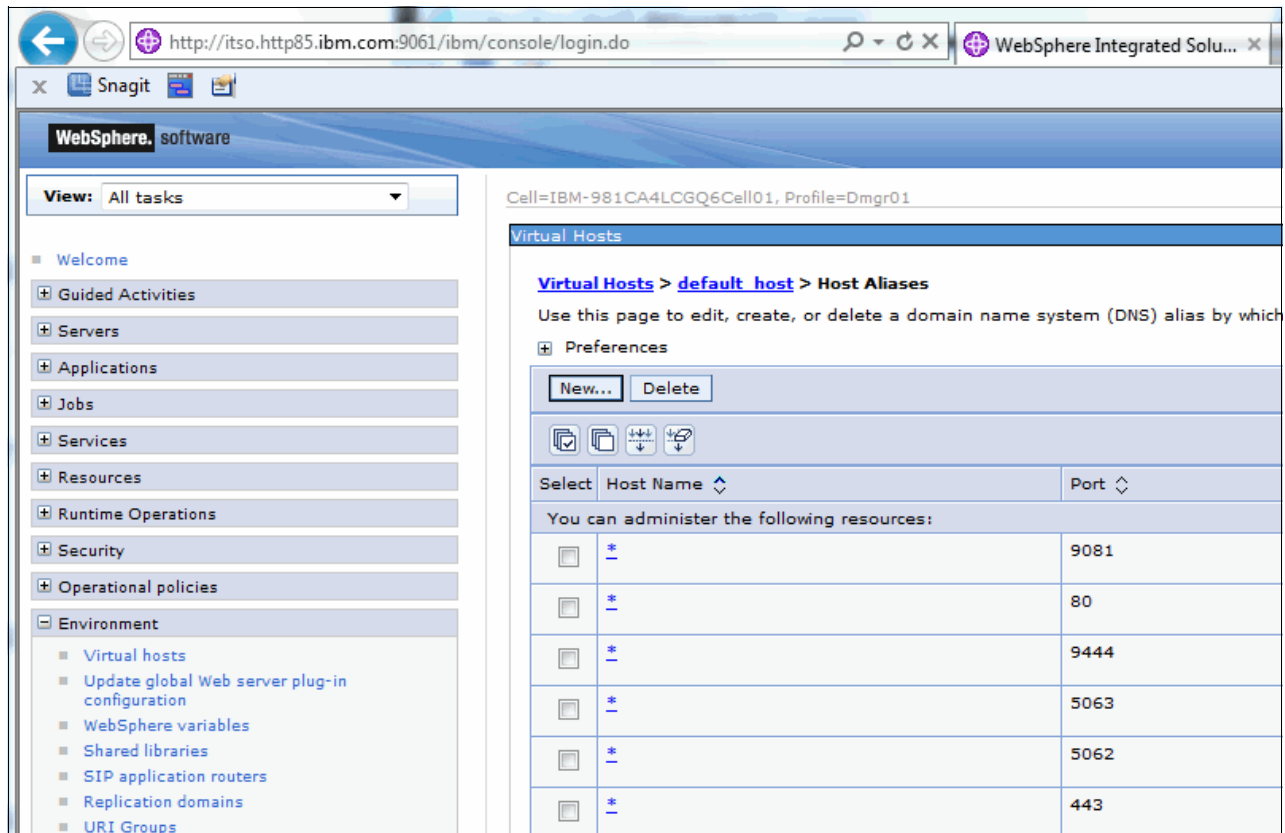


Figure 12-37 Administrative console showing virtual hosts

2. If port 8080 is missing, click **New**, and add the port. Process the following steps:
 - a. Restart the deployment manager.
 - b. Regenerate the plug-ins.
 - c. Propagate the plug-ins.
 - d. Refresh the web servers.
 - e. Check the URL again.

If port 8080 is present in the virtual hosts list, then do the following steps:

- a. Regenerate the plug-ins.
- b. Propagate the plug-ins.
- c. Restart the web servers.

Tip: Plug-ins do not always have information about the application or the virtual hosts, and re-generating the plug-ins can resolve the issue.

12.7.2 Troubleshooting Error 500

When you receive error 500 (internal server error), it means the web server cannot reach the application server. There are multiple reasons this can occur. We explore the most common two reasons:

- ▶ The JVM web server is not working. To troubleshoot this issue, the following steps are suggested:
 - a. Identify which JVM web server is trying to access from `$IHS_HOME/logs/error_log`

- b. In the WebSphere administration console, select **Servers** → **Server Types** → **WebSphere Application Server**. Check to see if the JVM web server is down. If it is down, select the **JVM web server**, and click **Start**.
- The port is not connected or blocked at the firewall level. To troubleshoot this issue, the following steps are suggested:
- a. Go to **Servers** → **Server Types** → **WebSphere Application Server**.
 - b. Click **server**, and select ports, as shown in Figure 12-38.
 - c. Check the `wc_defaulthost` port, exemplated as 9081.
 - d. From the web server, run `telnet WAS_SERVER_IP 9081`. The port shows if it is connected. If it does not show connected, it might be blocked at the firewall level.

| Port Name | Port | Details |
|---------------------------------------|-------|---------|
| BOOTSTRAP_ADDRESS | 2811 | |
| SOAP_CONNECTOR_ADDRESS | 8881 | |
| ORB_LISTENER_ADDRESS | 0 | |
| SAS_SSL_SERVERAUTH_LISTENER_ADDRESS | 9409 | |
| CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS | 9408 | |
| CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS | 9407 | |
| WC_adminhost | 9062 | |
| WC_defaulthost | 9081 | |
| DCS_UNICAST_ADDRESS | 9355 | |
| WC_adminhost_secure | 9045 | |
| WC_defaulthost_secure | 9444 | |
| SIP_DEFAULTHOST | 5063 | |
| SIP_DEFAULTHOST_SECURE | 5062 | |
| OVERLAY_UDP_LISTENER_ADDRESS | 11007 | |
| OVERLAY_TCP_LISTENER_ADDRESS | 11008 | |
| IPC_CONNECTOR_ADDRESS | 9634 | |
| SIB_ENDPOINT_ADDRESS | 7278 | |
| SIB_ENDPOINT_SECURE_ADDRESS | 7287 | |
| SIB_MQ_ENDPOINT_ADDRESS | 5559 | |
| SIB_MQ_ENDPOINT_SECURE_ADDRESS | 5579 | |

Figure 12-38 Checking port connections to troubleshoot error 500



Intelligent management

Intelligent management is a new group of capabilities integrated into WebSphere Application Server V8.5. These capabilities allow you to build virtualized application infrastructures to support the operation of your applications. Intelligent management provides a set of autonomic components that respond dynamically to the real-time conditions of the environment, adapting the infrastructure to respond to business needs, allowing requests to be prioritized, and intelligent routing to respond to critical applications and users.

This chapter includes the following topics:

- ▶ Introduction to Intelligent Management
- ▶ Configuring dynamic operations
- ▶ Configuring health management

13.1 Introduction to Intelligent Management

Intelligent management extends the quality of service provided by your middleware environment. With Intelligent management, configurable operational policies govern the performance and health of your applications. Total cost of ownership is decreased through server consolidation and lower administrative overhead, and you experience lower response times and increased availability. In short, you experience the benefits of an autonomic middleware environment that is self-configuring, self-protecting, self-healing, and self-optimizing.

Intelligent Management is not intended to eliminate the role of WebSphere Application Server administrator; instead, it allows the administrator to focus on managing more complex business requirements.

Intelligent Management is the integration of WebSphere Virtual Enterprise into WebSphere Application Server Network Deployment V8.5. The Intelligent management functionality includes the following key features:

- ▶ Intelligent routing improves the quality of service by ensuring that priority is given to business critical applications and users. Requests to applications are prioritized and routed based on administrator-defined rules.
- ▶ Health management provides the ability for you to specify conditions to be automatically detected and take corrective actions when these conditions are observed.
- ▶ Application edition management provides the ability to roll out new versions of applications without experiencing downtime for a maintenance window. Using this feature, you can validate a new edition of an application in your production environment without affecting users and upgrade your applications without incurring outages to your users. You can also run multiple editions of a single application concurrently, directing different users to different editions.
- ▶ Performance management provides a self-optimizing middleware infrastructure. By using dynamic clusters you can automatically scale up or down the number of running instances of a cluster to meet the defined service policies. You can take advantage of an overload protection to limit the workload of a server instance, and prevent heap exhaustion, CPU exhaustion, or both from occurring.

These capabilities are referred to as *dynamic operations*, which is the core functionality that provides application infrastructure virtualization.

The Intelligent Management functionality also provides support for a range of middleware servers. Middleware servers encompass all servers in the middleware tier that provide the infrastructure for applications or their data.

Middleware server support includes the following servers:

- ▶ Apache HTTP Server
- ▶ Apache Geronimo Server
- ▶ WebSphere Application Server Community Edition
- ▶ External Java application servers

For further information about the support of middleware servers, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.wve.doc/ae/cwve_xdmws.html

The following are the key elements and functions of Intelligent management:

► On Demand Router (ODR)

The ODR is an intelligent proxy and workload manager that acts as the entry point for traffic coming into a Websphere Application Server Network Deployment cell with an Intelligent management topology. The ODR performs request prioritization, flow control, and dynamic workload management, for HTTP requests and SOAP over HTTP requests.

In WebSphere Application Server V8.5.5, the ODR can be integrated into the web server plug-in for an Apache or IBM HTTP Server. This integration can simplify the topology and improve performance.

► Autonomic managers

Autonomic managers make decisions for the environment, including application management, traffic shaping, and health placement. The autonomic managers include the following components:

- Application placement controller (APC)
- Dynamic workload manager (DWLM)
- Autonomic request flow manager (ARFM)
- Health controller

For further information about the tasks that each of the autonomic managers provide, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.wve.doc/ae/cwve_odoecomponents.html

► Dynamic clusters

A dynamic cluster is a server cluster that enables application server virtualization. Members of a dynamic cluster are:

- Automatically created based on a membership policy
- Automatically updated using a server template
- Automatically started and stopped based on current demand, available resources, and service policies

This allows the application environment to dynamically expand and contract depending on the amount of workload that needs to be handled at any given time. For further information about dynamic clusters, see 15.1.3, “Dynamic cluster” on page 522.

The APC controls the operation of the dynamic clusters. Each node within a dynamic cluster has an instance of an application server running that cluster’s applications, which can be started dynamically as traffic for that application increases.

► Operational policies

An operational policy is a business or performance objective that supports specific goals for specific requests. Operational policies include service and health policies.

- Service policies specify performance goals for applications
- Health policies specify what constitutes server’s sickness and the appropriate actions to take when a sick server is detected

► Traffic shaping

Traffic shaping is the process of classifying incoming requests based on policies and managing the distribution of requests among application servers.

Traffic shaping is done at different entry points, depending on the type of request. For HTTP, SOAP, and Session Initiation Protocol (SIP) requests, traffic shaping occurs in the

ODR. For Internet Inter-ORB Protocol (IIOP) and Java Message Service (JMS) requests, traffic shaping occurs at the application server.

Autonomic managers play a key role in traffic shaping. They perform the classification and prioritization of requests and manage the environment to balance the workload.

- ▶ Health management

The health monitoring and management subsystem continuously monitors the operation of servers against user-defined health policies to detect functional degradation that is related to user application or server malfunctions.

- ▶ Runtime operation monitoring

The visualization components enhance the administrative console to provide live data on the performance and health characteristics of the entire cell. Refer to 16.5, “Monitoring operations” on page 584 for further information about runtime operation monitoring.

- ▶ Application edition management

Loss of service to users means loss of business to you. The application edition management feature ensures that the users of your application experience no loss of service when you install an application update in your environment. Refer to 24.4, “Application edition management and rollout” on page 889 for further information about how to use application edition management.

The topology shown in Figure 13-1 illustrates a WebSphere Application Server Network Deployment topology with the web server enabled for Intelligent Management.

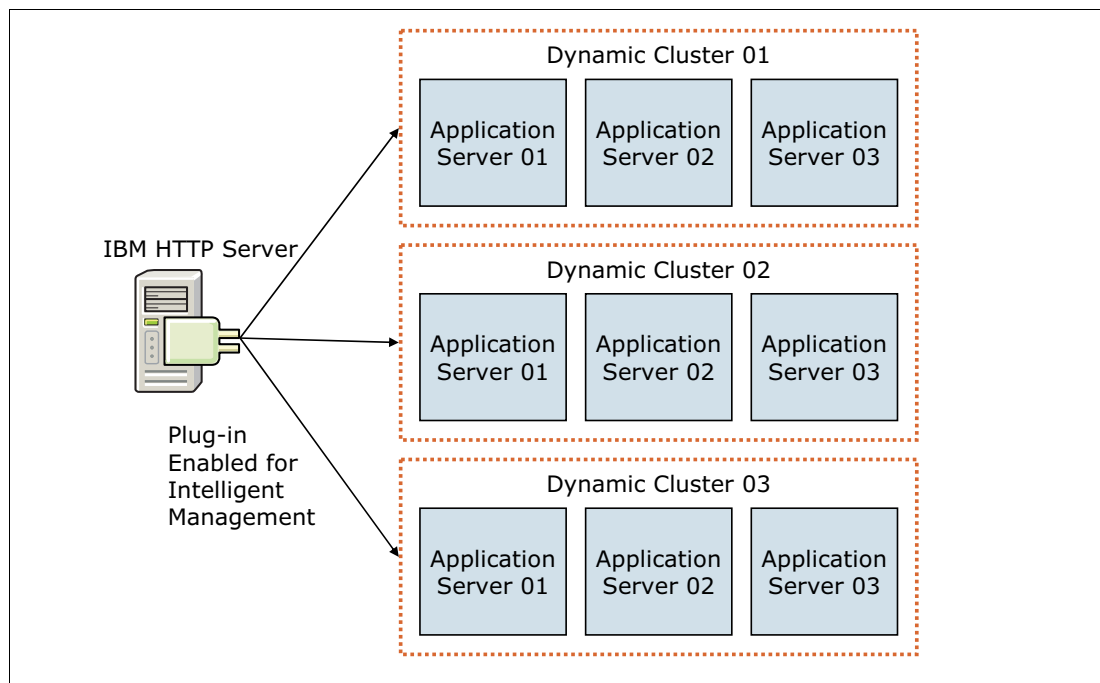


Figure 13-1 Intelligent Management topology with the web server enabled for Intelligent Management

13.2 Configuring dynamic operations

The dynamic operations environment consists of autonomic managers whose purpose is to maximize utilization using defined business goals. Dynamic operations allow an application environment to scale as required by the virtualization of WebSphere resources and the use of

a goals-directed infrastructure. Therefore, you can increase the speed at which your environment adapts to the business requirements.

Using the dynamic operations features of WebSphere Application Server you can change the way a typical WebSphere environment is configured to one that has the following features:

- ▶ Improves the utilization of available resources such as CPU and memory
- ▶ Classifies and monitors the workload
- ▶ Provides a business-centric view of the workload and how it is performing
- ▶ Responds in real time to changes in the workload mix (without human intervention if so desired) using business guidelines that the organization has specified

13.2.1 Creating the ODR

In versions prior to V8.5.5, the ODR is implemented as a server. With WebSphere Application Server V8.5.5, the ODR can be configured to run in the web server tier by enabling Intelligent Management in the web server plug-in. This option is available for Apache and IBM HTTP Servers. Using the plug-in option can simplify your topology, reduce the latency due to one less hop in the network, and be implemented easily.

Before determining whether to use the ODR server or the web server plug-in with Intelligent Management enabled, be aware that the plug-in option has the following limitations:

- ▶ If you configure the same application in multiple cells, there is no support for load balancing or failover between cells.
- ▶ CPU or memory overload protection is not supported.
- ▶ There is no support for queuing and reordering of requests from service policies. The dynamic clusters are supported.
- ▶ Highly available deployment manager topology is not supported.

Enabling Intelligent Management in the web server plug-in

The following steps describe how to enable Intelligent Management in the web server plug-in. In this example, the IBM HTTP Server is the target web server. First, configure the web server in the administrative console. After the configuration is complete, perform the following steps:

1. Click **Servers** → **Server Types** → **Web servers**.
2. Click the web server to open the configuration.
3. In the Additional Properties section, click **Intelligent Management**.
4. In the General Properties section, select the **Enable** check box. Click **Apply** to save the changes to the master repository.
5. Generate the plug-in and propagate it to the web server:
 - a. Click **Servers** → **Server Types** → **Web servers**.
 - b. Select the check box by the web server name.
 - c. Click **Generate Plug-in**.
 - d. Select the check box by the web server name again and click **Propagate Plug-in**.

It is not necessary to restart the web server process in Apache web server models; the process will reload the configuration file. After the save is completed, the `plugin-cfg.xml` file contains the information required for the web server to connect to the REST service port. You will see this service in the administrative console listed as `XDAGENT_PORT` in the port list.

This service is available in the deployment manager and node agent processes. Example 13-1 shows a snippet of the `plugin-cfg.xml` file with a reference to REST ports.

Example 13-1 Intelligent Management entries in the plugin-cfg.xml file

```
<IntelligentManagement>
  <Property name="webserverName"
value="localhostCell103_localhostNode02_webserver1"/>
  <ConnectorCluster enabled="true" maxRetries="-1" name="localhostCell103"
retryInterval="60000">
    <Connector host="localhost" port="7060" protocol="https">
      <Property name="keyring"
value="/opt/IBM/WebSphere855/Plugins/config/webserver1/plugin-key.kdb"/>
    </Connector>
    <Connector host="localhost" port="7061" protocol="https">
      <Property name="keyring"
value="/opt/IBM/WebSphere855/Plugins/config/webserver1/plugin-key.kdb"/>
    </Connector>
  </ConnectorCluster>
</IntelligentManagement>
```

Creating an ODR server

To create an ODR in a WebSphere Application Server Network Deployment cell, complete the following steps from the administrative console:

1. Click **Servers** → **Server Types** → **On Demand Routers**. In the ODR window, click **New**.
2. Select the node where the ODR will run, and enter a name for the ODR. Click **Next**. See Figure 13-2.

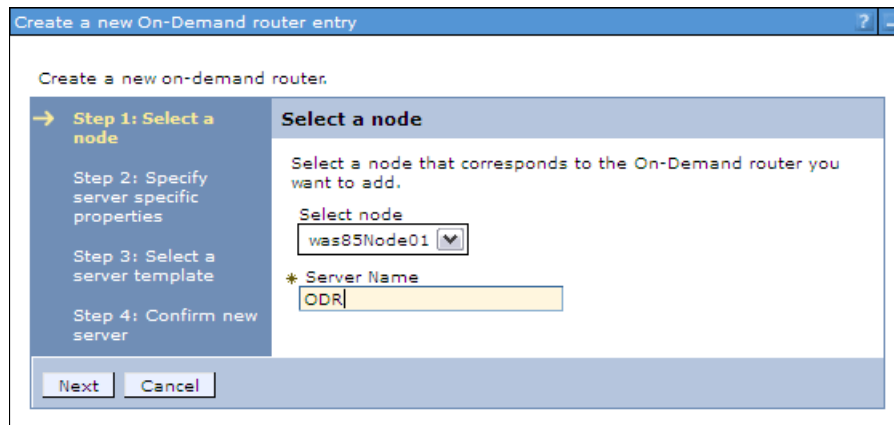


Figure 13-2 Creating a new ODR

3. Specify the protocols that the new ODR will handle. In this example, we use HTTP. Clear the check box for SIP, and click **Next**, as shown in Figure 13-3 on page 475.

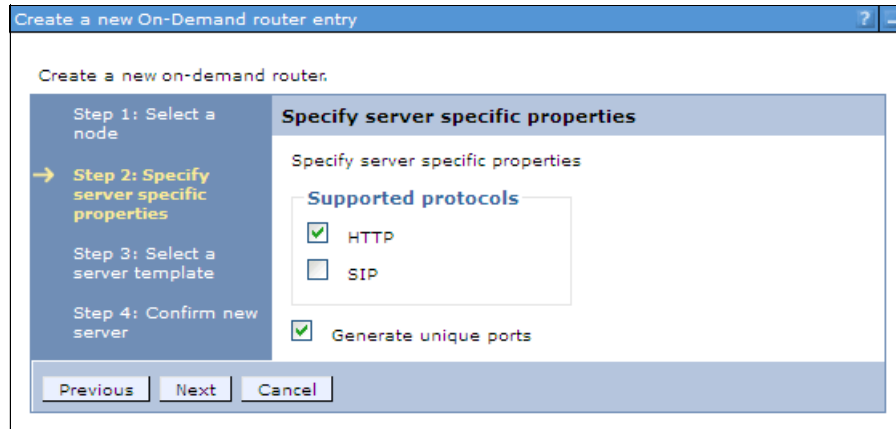


Figure 13-3 Specify the ODR properties

4. Select the ODR server template, and click **Next**.
5. Review the summary and then click **Finish**. Save the changes to the master repository. The new ODR is now created.

ODR servers can be clustered in a dynamic cluster (see “Creating an ODR server dynamic cluster” on page 549). If you use a cluster of ODR servers, there are other considerations regarding the web server plug-in configuration (see 15.2, “Workload management” on page 532 for further information).

13.2.2 Service policies

A *service policy* is a user-defined categorization that is assigned to potential work as an attribute that is read by the ARFM. You can use a service policy to classify requests based on request attributes, including the URI, the client name and address, and the user ID or group. By configuring service policies, you apply varying levels of importance to the actual work. You can use multiple service policies to deliver differentiated services to different categories of requests. Service policy goals can differ in performance targets as well as importances.

Note: Request prioritization is not supported when you use a web server enabled for Intelligent Management.

A service policy consists of a user-defined performance goal and, in some cases, an importance level. Service policies are related to work requests through transaction classes. There are three types of performance goals in WebSphere Application Server Network Deployment V8.5:

- ▶ Discretionary

This goal type indicates work that does not have significant value. Requests are processed when no higher request is waiting. As a result, work of this type can see a degradation in performance when resources are constrained. This is the default service goal.
- ▶ Average response time

This goal type allows you to specify the average response time goal in milliseconds or seconds. The system attempts to achieve this goal at a target percentage of 90% by default.

- ▶ Percentile response time

This goal type allows you to specify both the average response time goal and the target percentage, for example, 95% of all requests must be answered in less than 1000 milliseconds. This performance goal type is useful for applications that have application response times that occasionally deviate from the norm and can skew the average response time.

Administrators can specify the relative level of importance of a service policy. A request associated with a service policy of higher importance is given priority over a request that is associated with a service policy of lower importance. This guarantees that if performance goals for all service policies cannot be met due to prolonged intense overload to your environment, WebSphere Application Server can use the level of importance to decide which service policy takes priority. The following seven levels of importance can be set:

- ▶ Highest
- ▶ Higher
- ▶ High
- ▶ Medium
- ▶ Low
- ▶ Lower
- ▶ Lowest

Planning is essential to select the correct importance value that makes sense to the business requirements. One approach is to leave the majority of your applications with a discretionary goal, assign a higher goal to the important applications using service policies, and use the highest importance levels only if you need to further differentiation between the higher goal applications.

Work classes

A work class is the grouping of work to be done by an application server. WebSphere Application Server determines how to handle the work class through a set of rules that each work class contains.

For most requests, work classes are used to map incoming requests to transaction classes. As requests enter the ODR, they are mapped to a work class, they are then mapped to a transaction class depending on the classification rules and by extension to a service policy. For generic server clusters and for SIP, work classes are not used. The rules for classifying requests to transaction classes are configured on the ODRs.

There are two main types of work classes:

- ▶ Service policy work classes: Work class rules associate incoming work with a service policy, thus indicating to WebSphere Application Server when to forward the work to the application server.
- ▶ Routing work classes: Work class rules associate incoming work with a routing policy, thus indicating to WebSphere Application Server where to send the work.

HTTP requests and SIP messages are also associated with a single routing work class. Routing work classes do not exist for IIOP and JMS because these protocols do not flow through the ODR, so no routing policy is needed.

Work classes combined with classification rules allow the Autonomic Request Flow Manager (ARFM) to prioritize a request. For example, the /shop/checkout URI can get more resources than the /shop/info URI because checkout takes more time or because the business considers checkout higher importance.

There are four possible types of work classes based on the supported protocols in the application:

- ▶ HTTP work classes
- ▶ SOAP work classes
- ▶ IIOP work classes
- ▶ JMS work classes

Note: For applications that run on platforms other than WebSphere Application Server, only work classes based on the HTTP protocol are supported.

Work class requests classification rules

Work class requests can be classified by rules. The syntax and semantics of a boolean expression for a rule are similar to the WHERE clause of a structured query language (SQL) expression. You can combine the expressions with operators. WebSphere Application Server provides a subexpression builder to help you define these rules. Classification rules can be based on different information from the request, including client ip, user ID, roles, request query parameter, request header, HTTP method, and more. For a complete list of the information that can be used for building classification rules, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.wve.doc/ae/rwve_odrworkclass.html

Transaction classes

Transaction classes provide the link between applications and service policies. The service policy creates the goal, while the transaction and work classes are used to map requests to that goal. Transaction classes are defined in service policies. The relationship between service policies and transaction classes is one to many. A single service policy can have multiple transaction class definitions, but each transaction class belongs to exactly one service policy.

Transaction classes are a subcontainer of the service policy for work being classified into the service policy that can be used for finer-grained monitoring. They can also be used as a mechanism to group cross application work together for common monitoring.

Every service policy has a default transaction class, which in most scenarios is sufficient. Additional transaction classes are created when finer-grained monitoring is necessary for the environment. Each transaction class name must be unique within the cell.

Each work request belongs to exactly one transaction class, and each transaction class belongs to exactly one service policy.

Figure 13-4 on page 478 shows the relationship between service policies, work classes, and transaction classes. The uniform resource identifiers (URI) are grouped together in work classes. When a request for a specific URI arrives, the URI is checked against the classification rules. Based on the rules, different transaction classes are addressed. These transaction classes are uniquely assigned to a service policy. The request is processed based on the service policy. A request filter in the ODR handles these steps and classifies the incoming requests into the associated service policies.

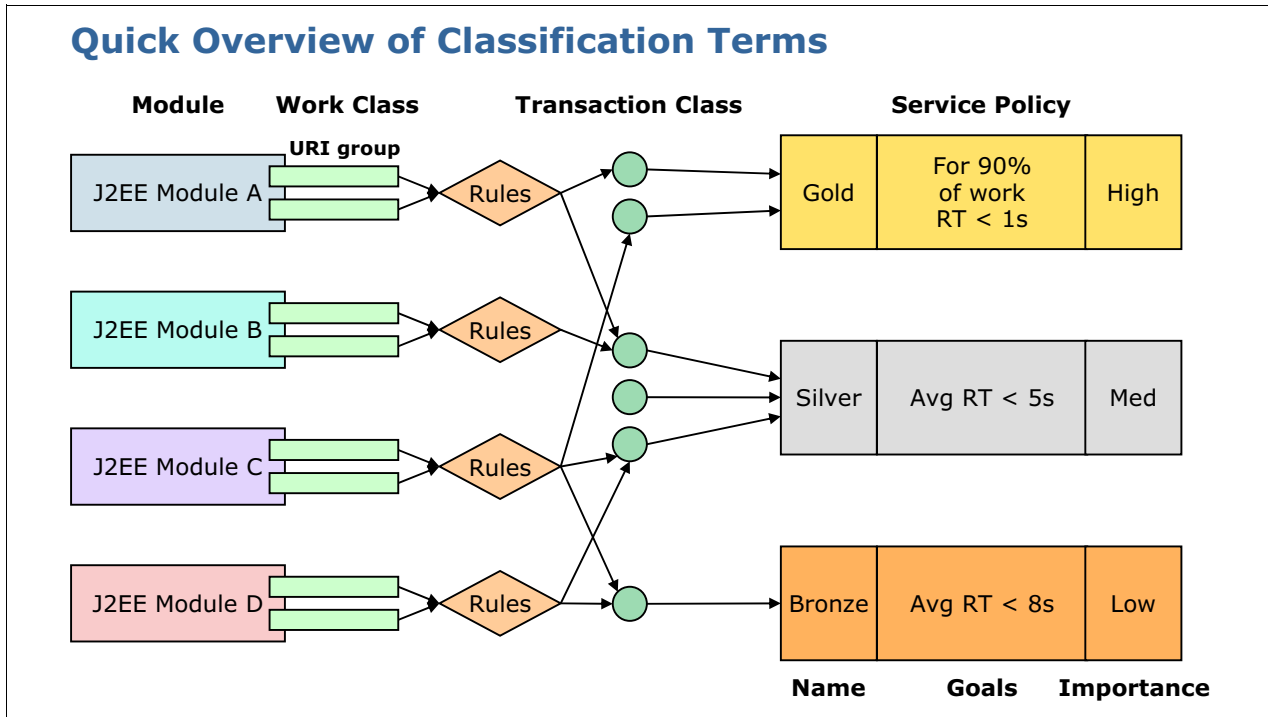


Figure 13-4 Work classes, transaction classes, and service policies

13.2.3 Creating service policies

Use the following steps to create a service policy:

1. In the administrative console, select **Operational policies** → **Service policies** and then click **New**.
2. Define the general property values for the service policy (Figure 13-5 on page 479). Enter a name and description for the new service policy, and select a goal type:
 - Average response time
 - Discretionary
 - Percentile response time

Click **Next**.

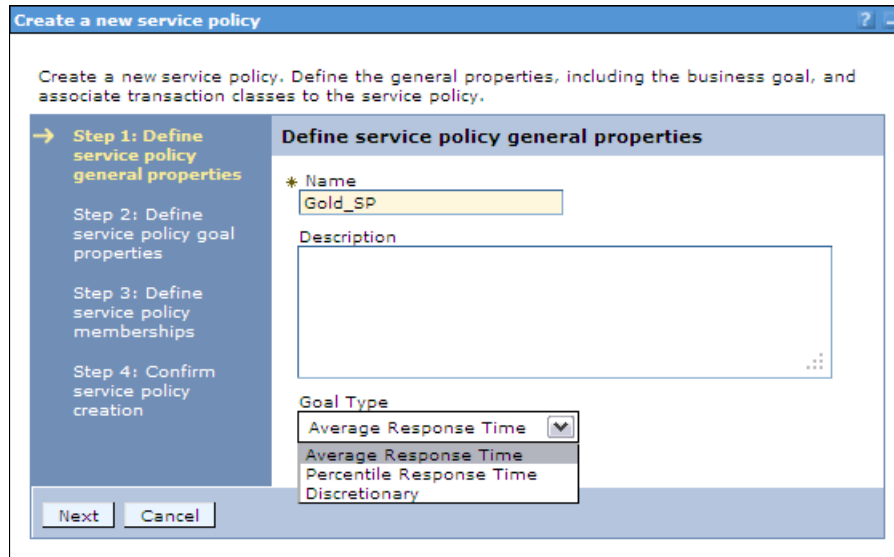


Figure 13-5 Define the service policy general properties

3. Optional: If you select a goal type of Average Response Time, or Percentile Response Time, you are prompted to define the specifics and select an importance. For the Average Response Time goal type, enter the following information (Figure 13-6 on page 480):
 - a. Enter a goal value.
 - b. Select the importance level.
 - c. If you want to monitor for persistent service policy violations and have a runtime task created, select the **Monitor for persistent violation** option, and enter values for the goal delta and time period:
 - Goal Delta Value: This is the allowable amount of time difference between the configured goal value and the actual average response time of requests that are served.
 - Time Period Value: This value signifies how long that goal delta value can be violated before it is considered breached and a runtime task is generated.

For information regarding percentile response time, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.wve.doc/ae/twve_odrpolicy.html

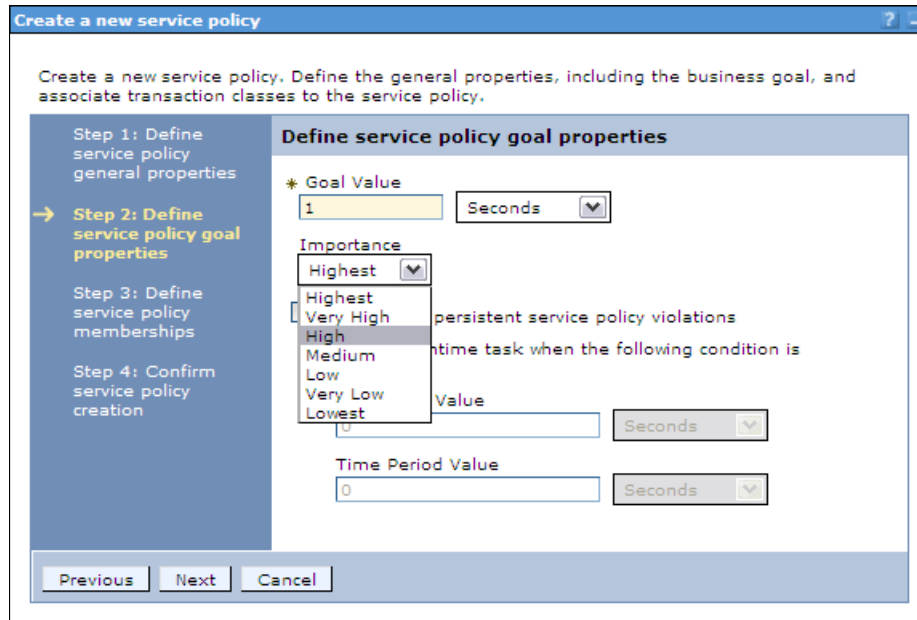


Figure 13-6 Define the service policy goal properties

Click **Next**.

- Optional: Define new transaction classes that are associated with this service policy. Note that a default transaction class is defined, as shown in Figure 13-7.

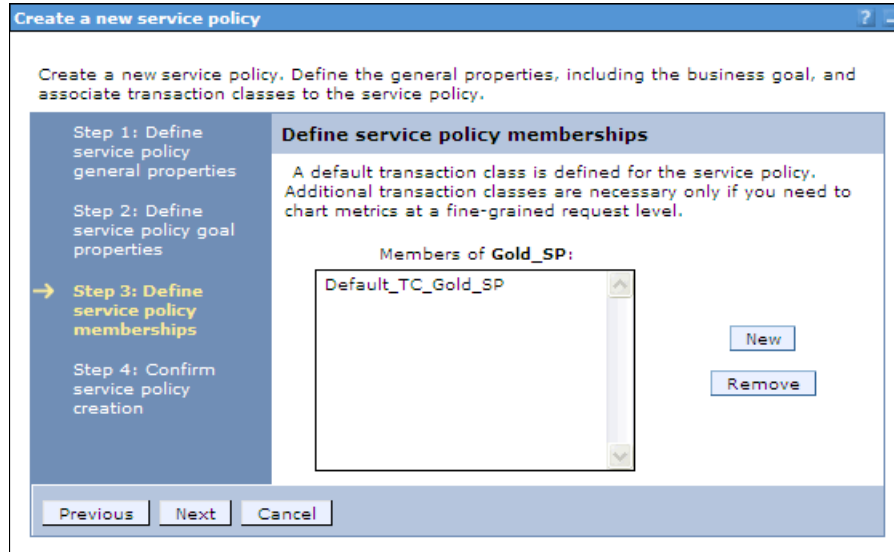


Figure 13-7 Define new service policies

Click **Next**.

- Review the summary and then click **Finish**. Save the changes to the master repository. A new service policy is created.
- Repeat steps 1-5 if you want to create more service policies.

Note: To define goal values for service policies, you have to complete the benchmarking for performance phase of the application development cycle. After that, you will know the response time of your applications under normal workloads and can assign a realistic goal value above this time.

13.2.4 Associating service policies with an application

With the service policies and transaction classes created, the next step is to define work classes for each application and associate the work class with the transaction class for the service policy. Work classes are associated with each application. We use the default application (DefaultApplication.ear) that comes with WebSphere Application Server to show the creation of work classes.

Complete the following steps to associate service policies with an application:

1. In the administrative console, select **Applications** → **Enterprise Applications** → **application_name** and then click the **Service Policies** tab (Figure 13-8).

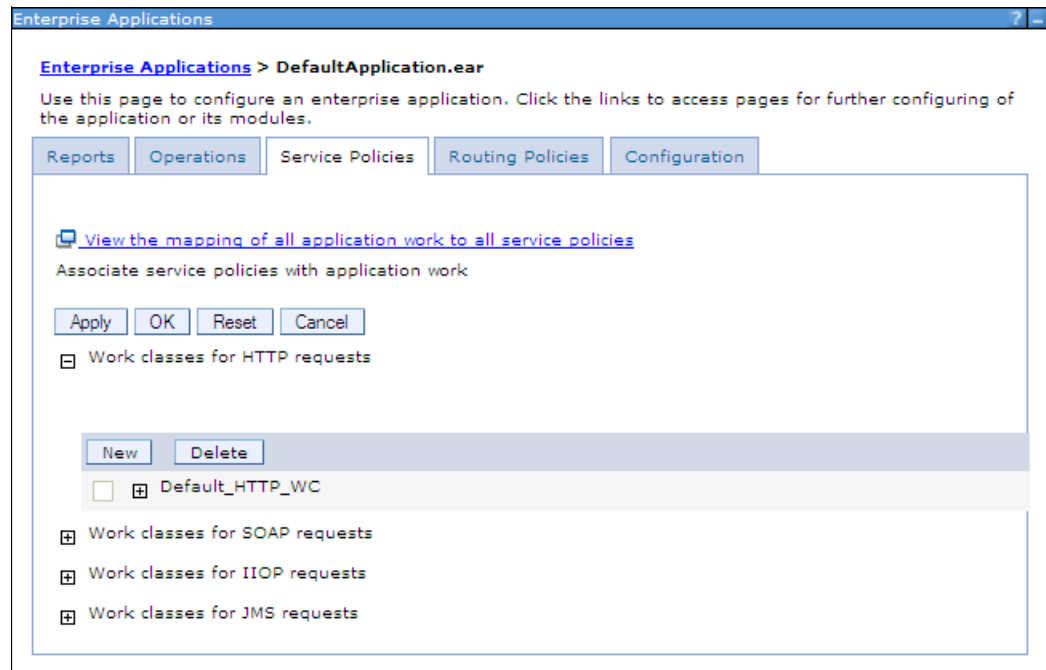


Figure 13-8 Specifying service policy settings for the application

2. Click **New** to define a new work class for HTTP requests.
3. Enter a name for the new work class and then click **Next** (Figure 13-9 on page 482).

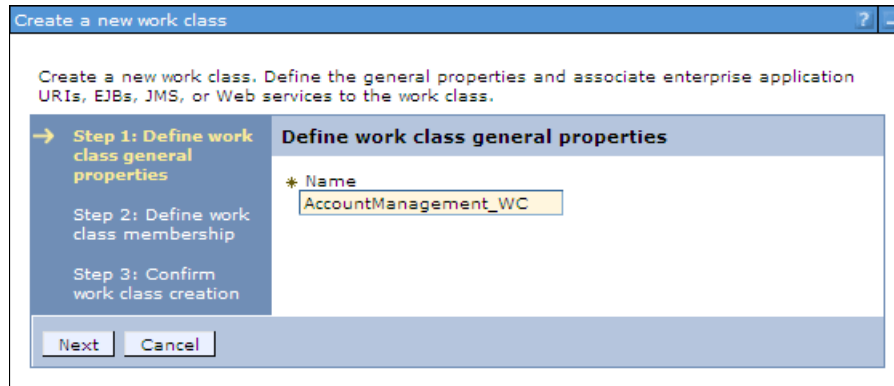


Figure 13-9 Define the work class name

4. Define the HTTP patterns that will be mapped to this work class (Figure 13-10):
 - a. Select the application module.
 - b. Select the HTTP patterns and then click **Add**.
 - c. Click **Next**.

Note that you can add custom HTTP patterns using the **Add Pattern** button.

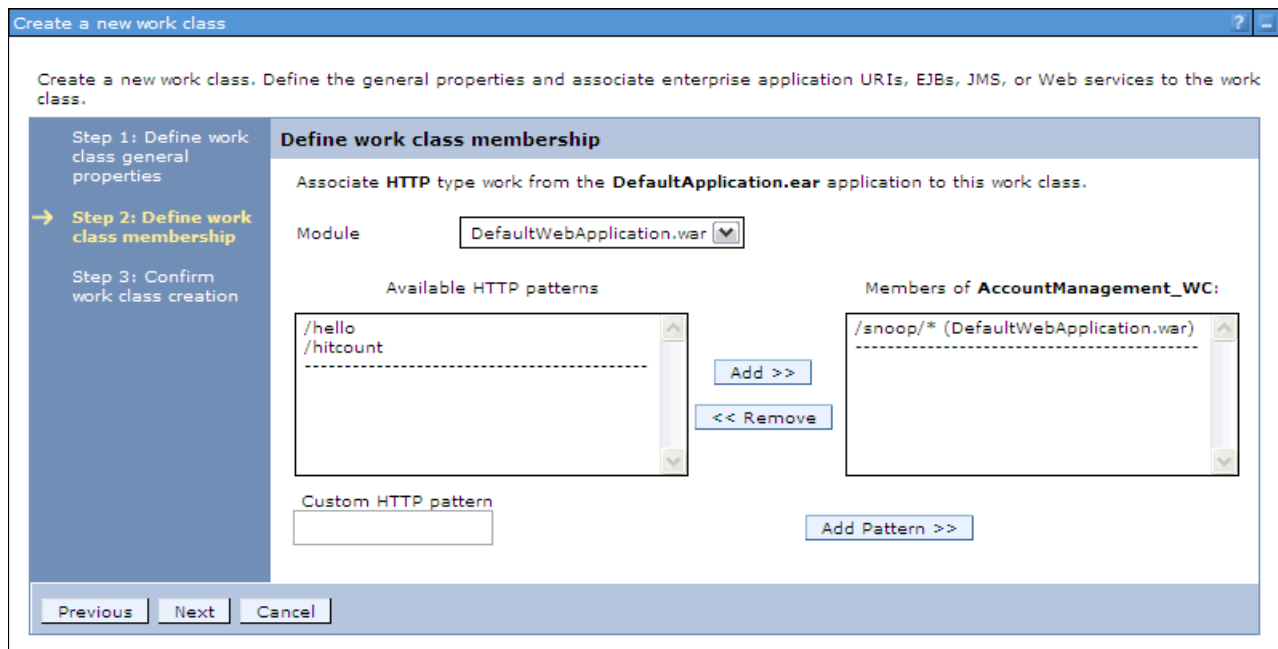


Figure 13-10 Define the HTTP patterns for the work class

5. Confirm work class creation and then click **Finish**.
6. Select an appropriate transaction class for this work class. You can apply different classification rules to the requests. Click **Add Rule** to configure additional classification rules, shown in Figure 13-11 on page 483.

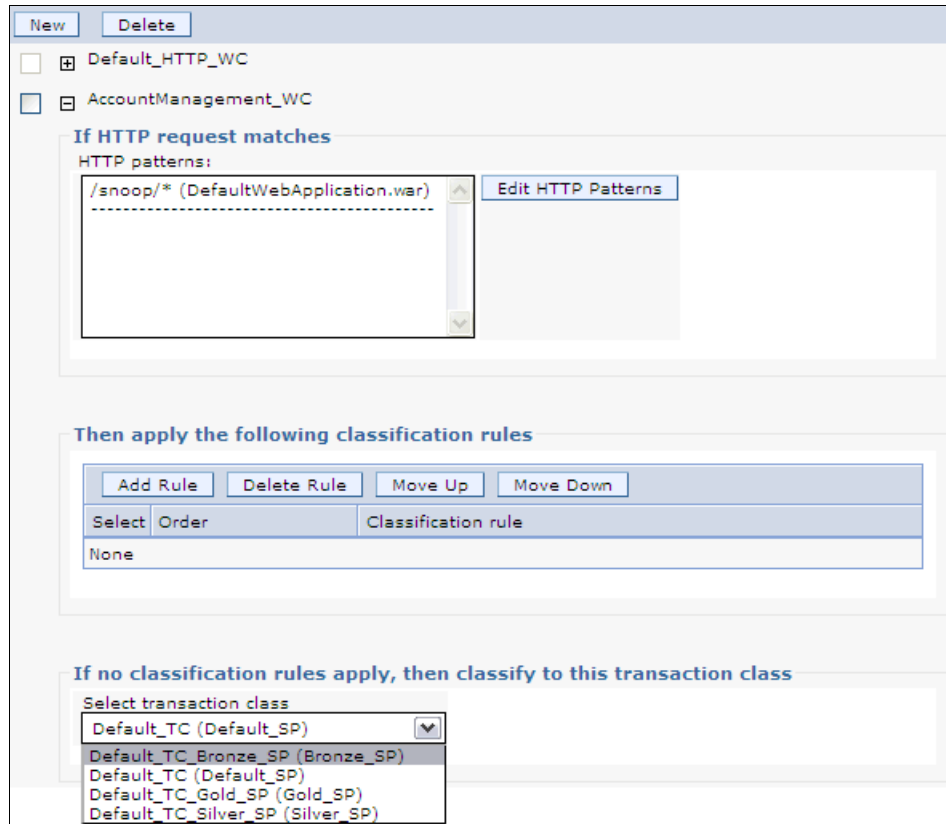


Figure 13-11 Work class specification

7. Repeat steps 2-6 to map another URI pattern to a service policy.
8. Click **OK**, and save the configuration.

If you click the link **View the mapping of all application work to all service policies** in the Service Policies tab (Figure 13-8 on page 481), you can see all the mappings defined for your application, as shown in Figure 13-12.



Figure 13-12 Mappings for all application

You can also see the mappings of service policies for all applications in the menu option by clicking **Operational policies** → **Service policy topology**.

Now that you defined the service policies for your application, WebSphere Application Server will make every effort to accomplish these policies.

13.3 Configuring health management

The health management subsystem provided with WebSphere Application Server Network Deployment V8.5 allows you to take a policy-driven approach to monitoring the application server environment and to define actions to be taken when certain criteria is discovered.

The health management subsystem consists of two main elements:

- ▶ Health policies define specific health criteria that can indicate a problem:
 - Where to monitor for this problem
 - The action to take
 - Whether the action is done automatically or by an operator
- ▶ A health controller monitors the WebSphere Application Server environment for conditions defined by the health policies and performs the appropriate actions.

Use health monitoring carefully, and only define and assign to servers if you think a particular health policy is needed. Health monitoring can make your environment more reliable, but it can also have performance impacts on the environment. Understanding the environment, including its capacity, usage, and loads will help you plan your policies.

Health monitoring is not meant to replace the testing and benchmarking phases of the application development lifecycle. The recommendation is that you test and benchmark for performance every application prior to being deployed in a WebSphere Application Server environment.

13.3.1 Health conditions

Health conditions define the variables that you want to monitor in your environment. Several categories of health policy conditions exist. You can choose from the following predefined health conditions:

- ▶ Age-based condition
Tracks the amount of time that the server is running. If the amount of time exceeds the defined threshold, the health actions run.
- ▶ Excessive request timeout condition
Specifies a percentage of HTTP requests that can time out. When the percentage of requests exceeds the defined value, the health actions run. The timeout value depends on your environment configuration.
- ▶ Excessive response time condition
Tracks the amount of time that requests take to complete. If the time exceeds the defined response time threshold, the health actions run.
- ▶ Memory condition: Excessive memory usage
Tracks the memory usage for a member. When the memory usage exceeds a percentage of the heap size for a specified time, health actions run to correct this situation.

- ▶ **Memory condition: Memory leak**
Tracks consistent downward trends in free memory that are available to a server in the Java heap. When the Java heap approaches the maximum configured size, you can perform either heap dumps or server restarts.
- ▶ **Storm drain condition**
Tracks requests that have a significantly decreased response time. This policy relies on change point detection on given time series data.
- ▶ **Workload condition**
Specifies a number of requests that are serviced before policy members restart to clean out memory and cache data.
- ▶ **Garbage collection percentage condition**
Monitors a Java virtual machine (JVM) or set of JVMs to determine whether they spend more than a defined percentage of time in garbage collection during a specified time period.

You can define custom conditions for your health policy if the predefined health conditions do not fit your needs. You define custom conditions as a subexpression that is tested against metrics in your environment. When you define a custom condition, consider the cost of collecting the data, analyzing the data, and if needed, enforcing the health policy. This cost can increase depending on the amount of traffic and the number of servers in your network. Analyze the performance of your custom health conditions before you use them in production.

For further information about creating custom conditions for your health policy, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.wve.doc/ae/cwve_hconditionsubex.html

13.3.2 Enabling and disabling health management

Health management is enabled by default. Use health management to protect your system from user application malfunctions, including memory leaks and application hangs. Health management uses health policies to define a set of conditions. Intelligent Management uses the health conditions to monitor the health of the system.

To enable or disable the health management:

1. In the administrative console, click **Operational policies** → **Autonomic managers** → **Health controller** (Figure 13-13 on page 486).

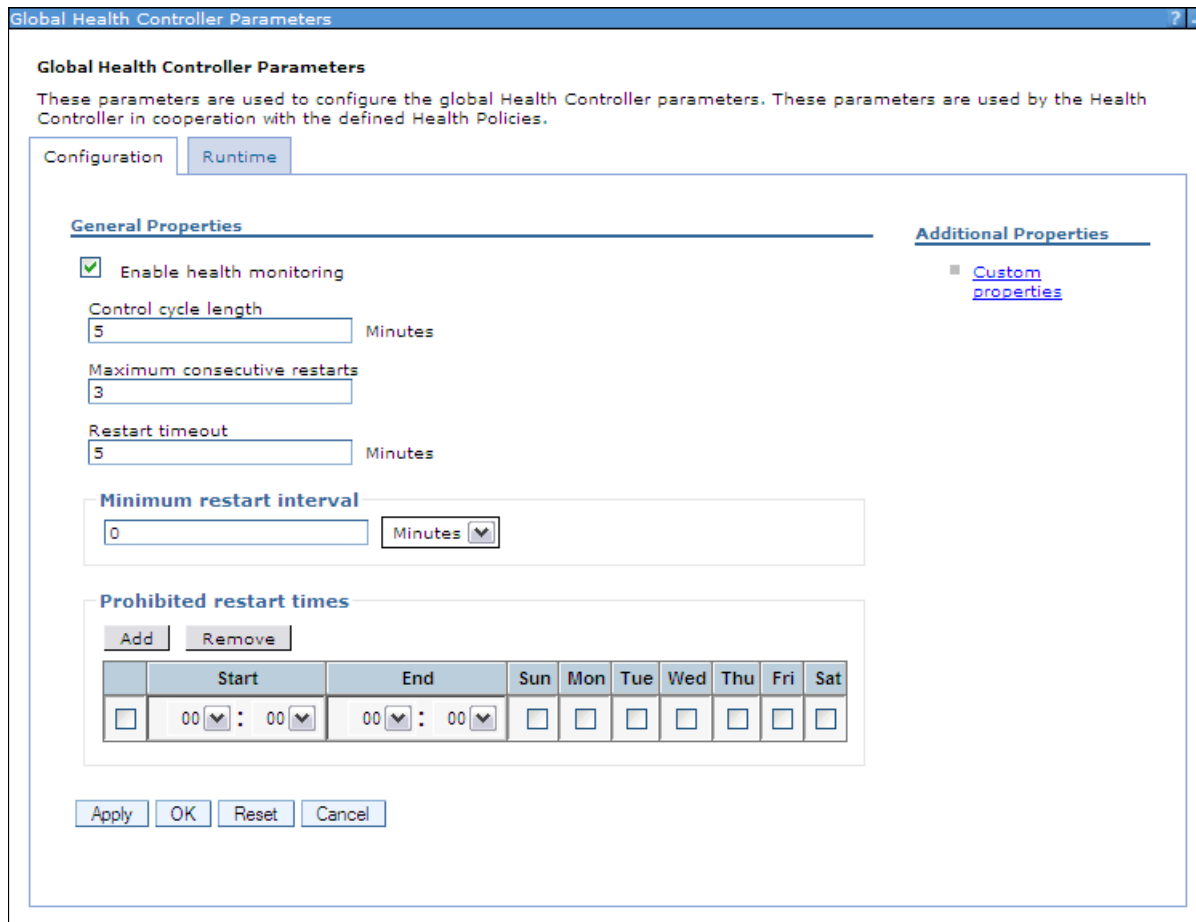


Figure 13-13 Configuring health controller

2. Enable or disable health monitoring. When the check box is selected, the health condition of the environment is monitored. When the check box is not selected, health monitoring is disabled.

For further information about the options you can configure in this window, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.wve.doc/ae/twve_odtunehealth.html

13.3.3 Health policy actions

There are different actions that are performed if a policy breach is detected. The possible actions that can be included into an action plan are:

- ▶ Restarting the application server

When a server is a member of a dynamic cluster, another instance of the dynamic cluster is started to serve user requests before the server that triggered the policy breach is shutdown. This allows WebSphere Application Server to handle potential problems with the least amount of impact to its consumers.
- ▶ Taking a thread dump (javacore)

The option to take thread dumps is only supported for application servers running in IBM JVMs.

- ▶ Taking JVM heap dumps on IBM Java Development Kit (JDK)
This option works for IBM JVMs only.
- ▶ Put server into maintenance mode
Maintenance mode is used to perform diagnostics, maintenance, or tuning on a node or server without disrupting incoming traffic. Putting a server into maintenance mode allows the remaining requests on the server to be processed.

Any requests that have an open session on the server are routed to the server until the session ends or times out. After all requests are completed, the server is moved to maintenance mode. Any new requests are routed to servers that are not in maintenance mode.
- ▶ Put server into maintenance mode, and break affinity
The HTTP and SIP session affinity is broken, and the session is moved to another server running in normal mode.
- ▶ Take server out of maintenance mode
After the server reaches a healthy state, it can be reinstated to serve requests. For example, if a server exceeds a memory threshold, putting the server in maintenance mode gives the server a chance to recover through garbage collection while no new requests are being sent to it. After heap utilization is below the threshold, the server can be taken out of maintenance mode.
- ▶ Custom action
With a custom action, you define a Java or non-Java executable file to define corrective actions to run when a health condition is broken.

13.3.4 Reaction mode

The health management subsystem functions in reaction mode, defined by the level of user-interaction when the health condition determines corrective action is needed. There are two possible reaction modes:

- ▶ Automatic mode
When the reaction mode on the policy is set to automatic, the health management system takes action when a health policy violation is detected. The logging data and the defined reaction are performed automatically.
- ▶ Supervised mode
The health management system creates a runtime task that proposes one or more reactions. The system administrator can approve or deny the proposed actions. The recommendations on actions are sent to the administrator. If the administrator follows the recommendations, the only action required is selecting a button, and the actions are performed. This option is widely preferred by the administrators who are not yet comfortable with giving to WebSphere Application Server with Intelligent management total control in performing autonomic actions.

13.3.5 Creating health policies

A health policy is the definition of specific health criteria that you want WebSphere Application Server to protect against. The health management function uses the defined policy to search the environment for software malfunctions.

To define a health policy using the administrative console:

1. Select **Operational policies** → **Health policies** and then click **New**.
2. Enter a name for the health policy and the health condition that will trigger the actions, (Figure 13-14).

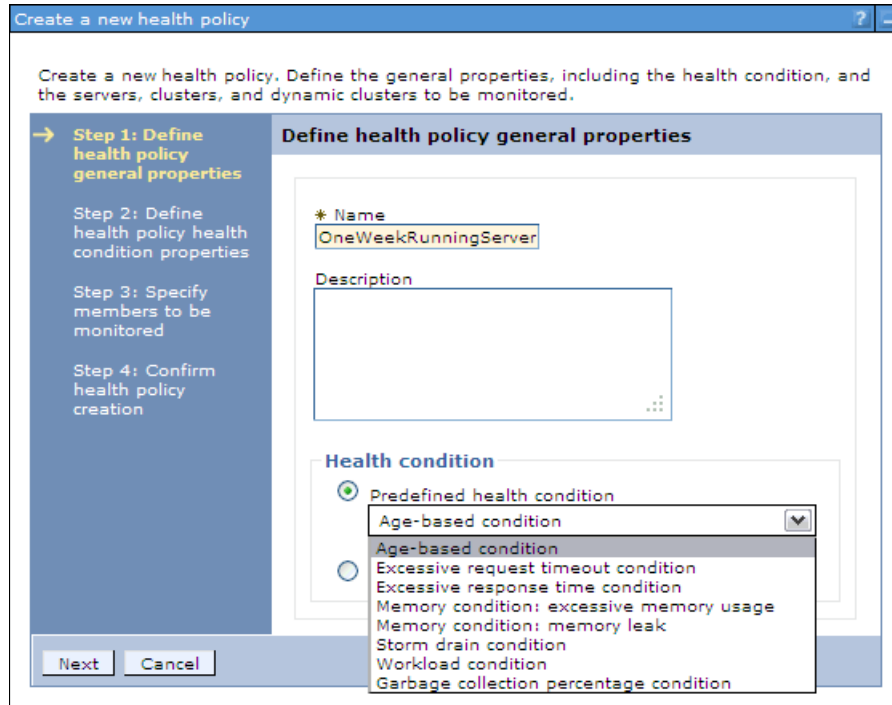


Figure 13-14 Define health condition

3. Depending on the health condition selected, enter the health condition general properties. Select the reaction mode, and configure the actions to be taken (Figure 13-15 on page 489).

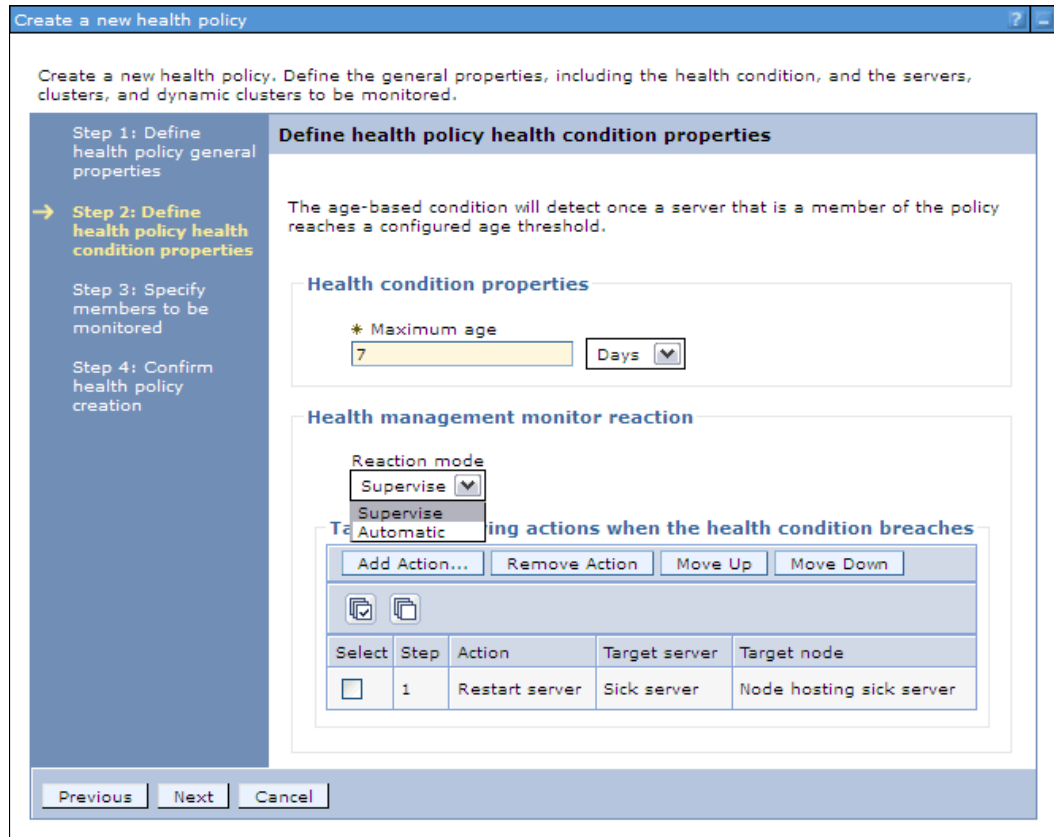


Figure 13-15 Define health condition properties

Click **Next**.

4. Select the members to monitor with this health policy. Specify the filter by option, click the member you want to add, and click **Add** (Figure 13-16).

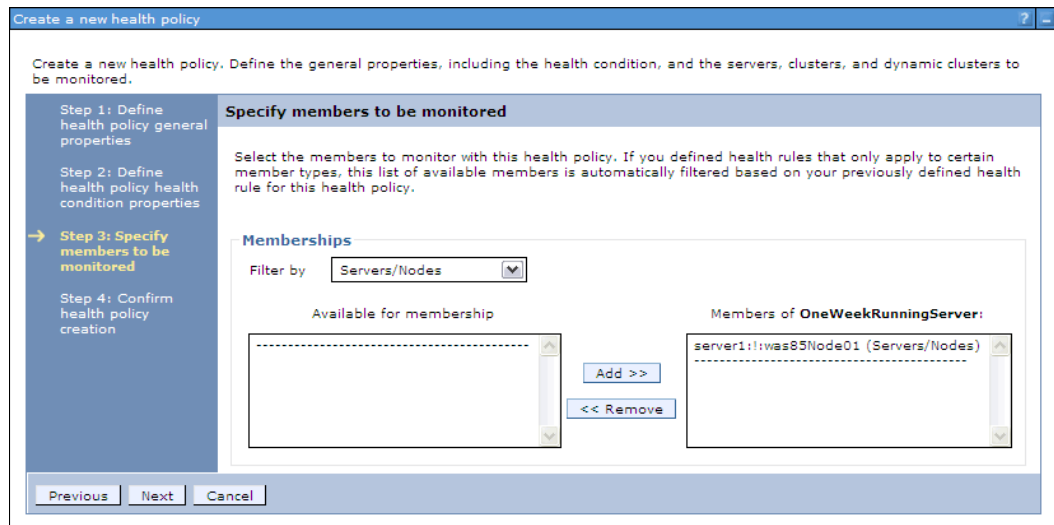


Figure 13-16 Specify members to be monitored

Click **Next**.

5. Review the summary and then click **Finish**. Save the changes to the master repository.

If the reaction mode of the health policy is set to supervised, and the health condition is breached, you will get a runtime task. To review the Runtime tasks, select **System administration** → **Task Management** → **Runtime Tasks**, (Figure 13-17).

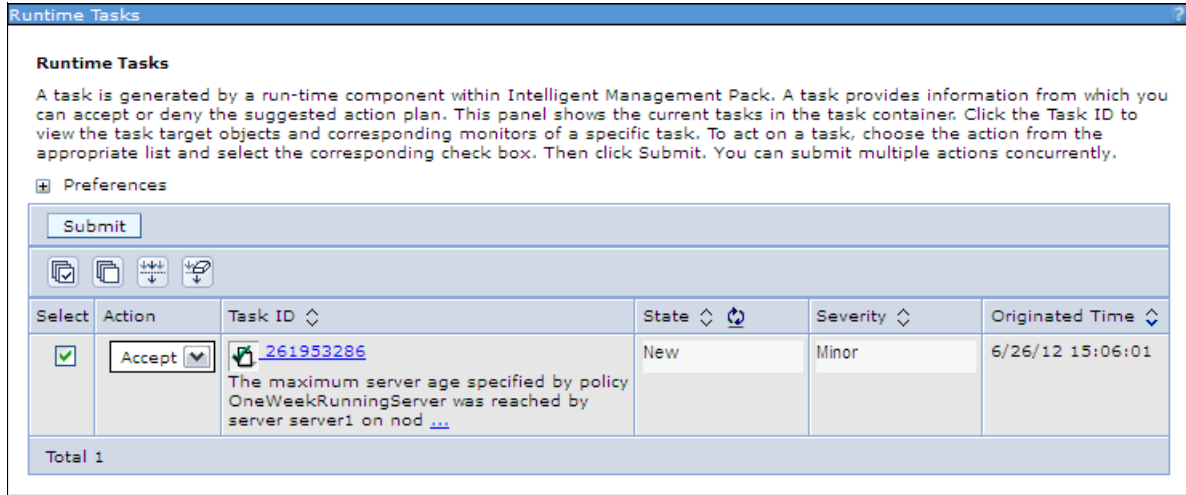


Figure 13-17 Runtime Taks list

To accept the runtime task and run the action plan for the health policy, select the task, select the **Accept** action and then click **Submit**.

For more information about monitoring operations, refer to 16.5, “Monitoring operations” on page 584.



Part 3

Managing distributed systems



Performance tuning on distributed environments

Performance tuning is a complex task that spans multiple components and areas with a goal of improving system performance. It is heavily dependent on application architecture, system infrastructure, and the amount of load on your system. WebSphere Application Server V8.5 delivers a scalable and highly available platform for applications and provides multiple methods for tuning options to optimize runtime performance.

This chapter provides information about methods and provides a list of tunable parameters by which you can improve performance. WebSphere provides an extensive number of tunable parameters. We include those parameters that might have the most noticeable benefits for your system. We use the queue analogy to represent WebSphere resource pools and provide a methodological approach to tuning these pools. We also include methods to tune Java virtual machines (JVM) and other various components.

This chapter includes the following topics:

- ▶ Performance tuning overview
- ▶ Using the queue analogy to tune WebSphere resource pools
- ▶ JVM tuning
- ▶ Other tuning considerations
- ▶ Tools
- ▶ Case Study

14.1 Performance tuning overview

Application architecture, system infrastructure, and the amount of load on the system are three essential factors that determine the optimum values for the parameters that we introduce in this chapter. There is no single value for any parameter that will work optimally on all systems. You need to go through capacity planning, stress test, collect information, and analyze results to reach a set of optimum values for your system.

To tune for performance, you need to have a performance test system that is identical to your production system. For the results of the test to be meaningful for the production, the hardware and software on the test system must be identical to the production.

To measure the success of your tests, generate a workload that meets the following characteristics:

- ▶ **Measurable:** Use a metric that can be quantified, such as throughput and response time.
- ▶ **Reproducible:** Ensure that the results can be reproduced when the same test is executed multiple times. Execute tests in the same conditions to define the real impacts of the tuning changes. Change only one parameter at a time.
- ▶ **Static:** Determine whether the same results can be achieved no matter for how long you execute the run.
- ▶ **Representative:** Ensure that the workload realistically represents the stress to the system under normal operating considerations. Execute tests in a production-like environment with the same infrastructure and the same amount of data.

To determine performance targets, you need a clear understanding of your system architecture and requirements. Investigate architectural documents, use cases, and functional and non-functional requirements. With a clear understanding, you can define performance success criteria.

Define your own performance success criteria. Without a goal or target, you cannot determine if the performance campaign was successful. You have to avoid non-figured success criteria, for example, aiming for the “best” performance that you can have. Keep in mind that performance testing can be endless if you do not have target figures to reach. Each time you test, you will find a new bottleneck to solve with a new solution. In the end, you cannot determine whether the test is a success or failure.

Do not attempt to conduct performance tuning on production systems with live load because there is a high chance that the server will be recycled for the new parameters to take effect. This recycling process can cause downtime for the production environment.

14.2 Using the queue analogy to tune WebSphere resource pools

WebSphere Application Server functions similarly to a queuing network, with a group of interconnected queues that represent various resources. Resource pools are established for the network, web server, web container, EJB container, Object Request Broker (ORB), data source, and possibly a connection manager to a custom back-end system. Each of these resources represents a queue of requests waiting to use that resource. Queues are load-dependent resources. As such, the average response time of a request depends on the number of concurrent clients. The tuning of these queues is an essential task to tune for performance.

As an example, think of an application, which consists of servlets and EJB beans, that accesses a database. Each of these application elements reside in the appropriate WebSphere component (for example, servlets in the web container), and each component can handle a certain number of requests in a given time frame.

A client request enters the web server and travels through WebSphere components to provide a response to the client. Figure 14-1 illustrates the processing path that this application takes through the WebSphere components as interconnected pipes that form a large tube.

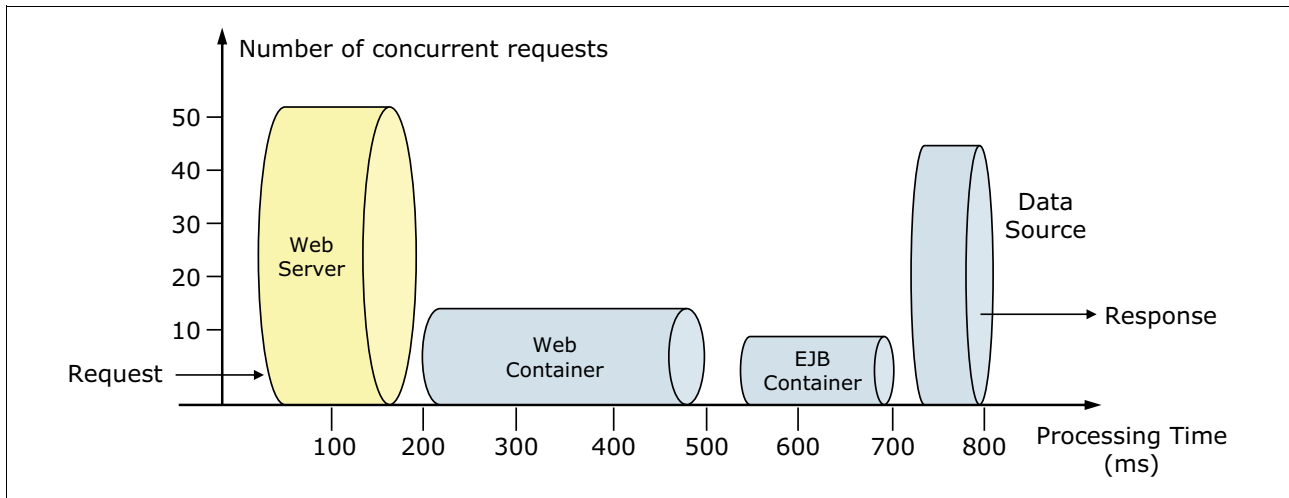


Figure 14-1 Queuing network

The width of the pipes (illustrated by height) represents the number of requests that can be processed at any given time. The length represents the processing time that it takes to provide a response to the request.

Figure 14-1 shows that at any given time, a web server can process 50 requests and a web container can process 18 requests. This difference implies that under peak load, requests are queued on the web server side, waiting for the web container to be available. In addition, the EJB container can process nine requests at a given time. Therefore, half of the requests in the web container are also queued, waiting for an ORB thread pool to be available. The database seems to have enough processing power; therefore, requests in the EJB container do not wait for database connections.

Suppose that we have adequate CPU and memory in the WebSphere Application Server system. Thus, we can increase the ORB pool size to better use the available database connections. Because the web server processes 50 requests at a given time, we can increase the web container thread pool size to more efficiently use the web container threads and to keep up with the increased number of ORB threads.

However, how do you determine the maximum amount of threads and database connections? If requests are queued due to processing differences, do you queue them closer to the data layer or closer to the client for best performance? We provide information to help answer these questions in the sections that follow.

14.2.1 Upstream queuing

The golden rule of resource pool tuning is to minimize the number of waiting requests in the WebSphere Application Server queues and to adjust resource pools in a way that resources wait in front of the web server. This configuration allows only requests that are ready to be processed to enter the queuing network. To accomplish this configuration, specify that the queues furthest upstream (closest to the client) are slightly larger and that the queues further downstream (furthest from the client) are progressively smaller. This approach is called *upstream queuing*.

Figure 14-2 shows an example of this type of queuing. When 200 client requests arrive at the web server, 125 requests remain queued in the network because the web server is set to handle 75 concurrent clients. As the 75 requests pass from the web server to the web container, 25 requests remain queued in the web server and the remaining 50 requests are handled by the web container. This process progresses through the data source until 25 user requests arrive at the final destination, which is the database server. Because there is work waiting to enter a component at each point upstream, no component in this system must wait for work to arrive. The bulk of the requests wait in the network, outside of WebSphere Application Server. This type of configuration adds stability because no component is overloaded.

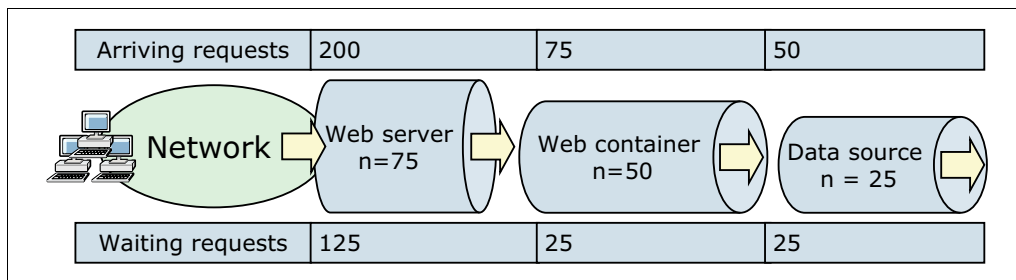


Figure 14-2 Upstream queuing

Because you do not have an infinite amount of physical resources for your system, do not use equal-sized queues. If you have infinite resources, you can tune the system such that every request from the web server has an available application server thread and every application server thread has an available database connection. However, for most real-world systems, this configuration is not possible.

Another important rule for accurate resource tuning is that you need to have a clear understanding of your system infrastructure, application architecture, and requirements. Different systems can have varying access and utilization patterns. For example, in many cases, only a fraction of the requests passing through one queue enters the next queue downstream. However, in a site with many static pages, many requests are fulfilled at the web server and are not passed to the web container.

In this circumstance, the web server queue can be significantly larger than the web container queue. In the previous example, the web server queue was set to 75 requests rather than closer to the value of the Max Application Concurrency parameter. You need to make similar adjustments when different components have different execution times. As the percentage of static content decreases, a significant gap in the web server queue and the application server queue can create poorly performing sites overall. Thus, before you begin any tuning activities, understand your system infrastructure, application architecture, and requirements.

For another example, in an application that spends 90% of its time in a complex servlet and only 10% of its time making a short JDBC query, on average 10% of the servlets are using database connections at any time. Thus, the database connection pool can be significantly smaller than the web container pool. Conversely, if much of a servlet execution time is spent making a complex query to a database, consider increasing the pool sizes at both the web container and the data source. Always monitor the CPU and memory utilization for both WebSphere Application Server and the database servers to ensure that the CPU or memory are not being overutilized.

In the following sections, we provide information about how to tune each resource pool with each pool starting from the furthest downstream, traveling upstream.

14.2.2 Data source tuning

When determining data source queues, consider tuning the following settings:

- ▶ Connection pool size
- ▶ Prepared statement cache size

Connection pool size

When accessing any database, the initial database connection is an expensive operation. WebSphere Application Server provides support for connection pooling and connection reuse. The connection pool is used for direct JDBC calls within the application and for enterprise beans that use the database.

IBM Tivoli Performance Viewer can help you determine the optimal size for the connection pool. Use a standard workload that represents a typical number of incoming client requests, a fixed number of iterations, and a standard set of configuration settings. Watch the Pool Size, Percent Used, and Concurrent Waiters counters of the data source entry under the JDBC Connection Pools module. The optimal value for the pool size is the value that reduces the values for these monitored counters. If the Percent Used counter is consistently low, consider decreasing the number of connections in the pool.

Better performance is generally achieved if the value for the connection pool size is set lower than the value for the Max Connections parameter in the web container. Lower settings for the connection pool size (10-30 connections) typically perform better than higher settings (more than 100). On UNIX platforms, a separate DB2 process is created for each connection. These processes affect performance on systems with low memory, which causes errors.

Each entity bean transaction requires an additional connection to the database specifically to handle the transaction. Be sure to take this connection into account when calculating the number of data source connections. The connection pool size is set from the administrative console by completing the following steps:

1. In the console navigation tree, click **Resources** → **JDBC Providers**.
2. Select the appropriate scope (cell, node, or server), depending on your configuration.
3. Open the JDBC provider configuration by clicking the name of the provider.
4. Under Additional Properties, select the **Data Sources** entry.
5. Open the data source configuration by clicking the data source name.
6. Click **Connection pool properties**.
7. Use the Minimum connections and Maximum connections fields to configure the pool size.
8. Save the configuration, and restart the affected application servers for the changes to take effect.

The default values are 1 for Minimum connections and 10 for Maximum connections.

A deadlock can occur if the application requires more than one concurrent connection per thread and if the database connection pool is not large enough for the number of threads. Suppose each of the application threads requires two concurrent database connections and the number of threads is equal to the maximum connection pool size. Deadlock can occur when both of the following statements are true:

- ▶ Each thread has its first database connection, and all connections are in use.
- ▶ Each thread is waiting for a second database connection, and no connections will become available because all threads are blocked.

To prevent the deadlock in this case, the value set for the database connection pool must be at least one higher than the number of waiting threads to have at least one thread complete its second database connection. To avoid deadlock, code the application to use, at most, one connection per thread. If the application is coded to require concurrent database connections per thread, the connection pool must support at least the following number of connections, where T is the maximum number of threads:

$$T * (C - 1) + 1$$

The connection pool settings are directly related to the number of connections that the database server is configured to support. If you raise the maximum number of connections in the pool and if you do not raise the corresponding settings in the database, the application fails and SQL exception errors are displayed in the SystemErr.log file.

Prepared statement cache size

The data source optimizes the processing of prepared statements to help make SQL statements process faster. It is important to configure the cache size of the data source to gain optimal statement execution efficiency. A prepared statement is a precompiled SQL statement that is stored in a prepared statement object. This object is used to efficiently execute the given SQL statement multiple times. If the JDBC driver specified in the data source supports precompilation, the creation of the prepared statement sends the statement to the database for precompilation. Some drivers might not support precompilation, and the prepared statement might not be sent until the prepared statement is executed.

If the cache is not large enough, useful entries are discarded to make room for new entries. In general, the more prepared statements that your application has, the larger the cache must be. For example, if the application has five SQL statements, set the prepared statement cache size to 5 so that each connection has five statements.

Tivoli Performance Viewer can help tune this setting to minimize cache discards. Use a standard workload that represents a typical number of incoming client requests, a fixed number of iterations, and a standard set of configuration settings. Watch the PrepStmtCacheDiscardCount counter of the JDBC Connection Pools module. The optimal value for the statement cache size is the setting used to get either a value of zero or the lowest value for the PrepStmtCacheDiscardCount counter.

As with the connection pool size, the statement cache size setting requires resources at the database server. Specifying too large a cache can have an impact on database server performance. Consult your database administrator to determine the best setting for the prepared statement cache size.

Note: The statement cache size setting defines the maximum number of prepared statements cached per connection.

You can set the cache size from the administrative console using these steps:

1. In the console navigation tree, select **Resources** → **JDBC Provider**.
2. Select the appropriate scope (cell, node or server), depending on your configuration.
3. Open the JDBC provider configuration by clicking the name of the provider.
4. Under Additional Properties, select the **Data Sources** entry.
5. Open the data source configuration by clicking the data source name.
6. Select **WebSphere Application Server data source properties**.
7. Use the Statement cache size field to configure the total cache size.
8. Save the configuration, and restart the affected application servers for the change to take effect.

14.2.3 EJB container

The Enterprise JavaBeans (EJB) container can be another source of potential scalability bottlenecks. The inactive pool cleanup interval is a setting that determines how often unused EJB beans are cleaned from memory. Set this interval too low, and the application spends more time instantiating new EJB beans when an existing instance can be reused. Set the interval too high, and the application has a larger memory heap footprint with unused objects that remain in memory. EJB container cache settings can also create performance issues if not properly tuned for the system.

In the sections that follow, we describe the parameters that you can use to make adjustments that might improve performance for the EJB container.

Cache settings

To determine the cache absolute limit, multiply the number of enterprise beans that are active in any given transaction by the total number of concurrent transactions expected. Next, add the number of active session bean instances. Use Tivoli Performance Viewer to view bean performance information. The cache settings consist of the following parameters:

- ▶ The cache size: The cache size specifies the number of buckets in the active instance list within the EJB container.
- ▶ The cleanup interval: The cleanup interval specifies the interval at which the container attempts to remove unused items from the cache to reduce the total number of items to the value of the cache size.

To change these settings, click **Servers** → **Application servers** → **<AppServer_Name>** → **EJB Container Settings** → **EJB container** → **EJB cache settings**.

The default values are Cache size=2053 buckets and Cache cleanup interval=3000 milliseconds.

ORB thread pool size

Method invocations to enterprise beans are only queued for requests coming from remote clients going through the RMI activity service. An example of such a client is an EJB client running in a separate JVM (another address space) from the enterprise bean. In contrast, no queuing occurs if the EJB client (either a servlet or another enterprise bean) is installed in the same JVM that the EJB method runs on and the same thread of execution as the EJB client.

Remote enterprise beans communicate by using the RMI/IIOP protocol. Method invocations initiated over RMI/IIOP are processed by a server-side ORB. The thread pool acts as a queue

for incoming requests. However, if a remote method request is issued and there are no more available threads in the thread pool, a new thread is created. After the method request completes, the thread is destroyed. Therefore, when the ORB is used to process remote method requests, the EJB container is an open queue because of the use of unbounded threads.

Tivoli Performance Viewer can help tune the ORB thread pool size settings. Use a standard workload that represents a typical number of incoming client requests, a fixed number of iterations, and a standard set of configuration settings. Watch the PercentMaxed counter of the Thread Pools module. If the value of this counter is consistently in the double digits, the ORB might be a bottleneck, and you need to increase the number of threads in the pool.

The degree to which you need to increase the ORB thread pool value is a function of the number of simultaneous servlets (that is, clients) that are calling enterprise beans and the duration of each method call. If the method calls are longer or if the applications spend a lot of time in the ORB, consider making the ORB thread pool size equal to the web container size. If the servlet makes only short-lived or quick calls to the ORB, servlets can potentially reuse the same ORB thread. In this case, the ORB thread pool can be small, perhaps even one-half of the thread pool size setting of the web container.

You can configure the ORB thread pool size from the administrative console by completing the following steps:

1. To change these settings, click **EJB cache settings**.
2. Click **Servers** → **Application servers** → **<AppServer_Name>** → **Container Services**.
3. Click **ORB Service** → **Thread Pool**.
4. Use the Maximum Size field to configure the maximum pool size. Note that this setting affects only the number of threads that are held in the pool. The actual number of ORB threads can be higher.
5. Save the configuration, and restart the affected application server for the change to take effect.

14.2.4 Web container tuning

Monitor the web container thread pool closely during initial performance runs. This bottleneck is the most common bottleneck in an application environment. Define the web container size, considering all the infrastructure chain in close cooperation with the web server number of threads and the number of sessions of the database. If you adjust the number of threads to be too low, the web server threads can wait for the web container. If you adjust the number of threads to be too high, the back end can be inundated with too many requests. For both circumstances, the consequence is an increase of the response time or even a hang.

To route servlet requests from the web server to the web containers, a transport connection between the web server plug-in and each web container is established. The web container manages these connections through transport channels and assigns each request to a thread from the web container thread pool.

Web container thread pool

The web container maintains a thread pool to process inbound requests for resources in the container (that is, servlets and JSP pages).

Tivoli Performance Viewer can help tune the web container thread pool size settings. Use a standard workload that represents a typical number of incoming client requests, a fixed number of iterations, and a standard set of configuration settings. Watch the PercentMaxed

and ActiveCount counters of the Thread Pools module. If the value of the PercentMaxed counter is consistently in the double digits, the web container can be a bottleneck, and you need to increase the number of threads. Alternatively, if the number of active threads are significantly lower than the number of threads in the pool, consider lowering the thread pool size for a performance gain.

You can configure the web container thread pool size from the administrative console by completing the following steps:

1. Click **Servers** → **Application servers** → **<AppServer_Name>**.
2. Under Additional Properties, click the **Thread Pools** entry.
3. In the thread pools list of the workspace, click the **WebContainer** entry.
4. Use the Maximum Size field to configure the maximum pool size. Note that in contrast to the ORB, the web container uses threads only from the pool. Thus, this configuration is a closed queue. The default value is 50.
5. Save the configuration, and restart the affected application server for the change to take effect.

Note: Selecting the **Allow thread allocation beyond maximum thread size** option on the Web Container Thread Pool Configuration window allows for an automatic increase of the number of threads beyond the maximum size that is configured for the thread pool. As a result, the system can become overloaded.

HTTP transport channel maximum persistent requests

The maximum persistent requests is the maximum number of requests that are allowed on a single keep-alive connection. This parameter can help prevent denial of service attacks when a client tries to hold on to a keep-alive connection. The web server plug-in keeps connections open to the application server as long as it can, providing optimum performance. A good starting value for the maximum number of requests that are allowed is 100 (which is the default value). If the application server requests are received from the web server plug-in only, increase this value.

You can configure the maximum number of requests that are allowed from the administrative console by completing the following steps:

1. Click **Servers** → **Application servers** → **<AppServer_Name>**.
2. Under Container Settings, click **Web Container Settings** → **Web container transport chains**.
3. Select the transport chain you want to modify, for example, WCInboundDefault.
4. In the Transport Channels pane, click **HTTP Inbound Channel (HTTP #)**.
5. Enter a value in the Maximum persistent requests field.
6. Click **OK**.
7. Save the configuration, and restart the affected application server for the change to take effect.

14.2.5 Web server tuning

There are several configuration options that impact the performance of the web server, such as the number of concurrent requests, keep-alive settings, or SSL parameters. For the scope of this section, we focus on the number of concurrent requests.

The web server must allow for sufficient concurrent requests to make full use of the application server infrastructure. The web server also acts as a filter and keeps users waiting in the network to avoid flooding the servers if more requests than the system can handle are incoming. As a rough initial start value for testing the maximum concurrent threads (one thread can handle one request at a time), use the following setting:

$$\text{MaxClients} = (((TH + MC) * WAS) * 1.26) / WEB$$

In the setting, the following definitions apply:

| | |
|------------|---|
| TH | Number of threads in the web container |
| MC | MaxConnections setting in the <code>plugin-cfg.xml</code> |
| WAS | Number of WebSphere Application Server servers |
| WEB | Number of web servers |

Monitoring the web server using tools, such as `server-status`, can help tune the maximum concurrent thread processing setting for the web server. Use a standard workload that represents a typical number of incoming client requests, use a fixed number of iterations, and use a standard set of configuration settings. Watch the number of web server threads going to the web container and the number of threads accessing static content locally on the web server. We explain how to enable `server-status` in 14.5.5, “IBM HTTP server status monitoring page” on page 515.

For additional information about IBM HTTP Server performance tuning, refer to the following websites:

- ▶ <http://www-01.ibm.com/support/docview.wss?uid=swg21167658>
- ▶ http://publib.boulder.ibm.com/httserv/ihsdiag/ihs_performance.html

A simple way to determine the correct queue size for any component is to perform a number of load runs against the application server environment at a time when the queues are large, ensuring maximum concurrency through the system. For example, one approach might be as follows:

- ▶ Set the queue sizes for the web server, web container, and data source to initial values. Set the values as listed in 14.2.6, “Estimating web container and ORB thread pool initial sizes” on page 504 to estimate initial values for the web container and ORB thread pools.
- ▶ Simulate a large number of typical user interactions entered by concurrent users in an attempt to fully load the WebSphere environment. In this context, *concurrent users* means simultaneously active users that send a request, wait for the response, and immediately re-send a new request upon response reception. You can use any stress tool to simulate this workload, such as IBM Rational Performance Tester.
- ▶ Measure overall throughput, and determine at what point the system capabilities are fully stressed (the saturation point).
- ▶ Repeat the process, each time increasing the user load. After each run, record the throughput (requests per second) and response times (seconds per request), and plot the throughput curve.

The throughput of WebSphere Application Server is a function of the number of concurrent requests that are present in the total system. At some load point, congestion develops due to a bottleneck and throughput increases at a much lower rate until reaching a saturation point (maximum throughput value). The throughput curve can help you identify this load point.

It is desirable to reach the saturation point by driving CPU utilization close to 100% because this point gives an indication that a bottleneck is not caused by something in the application. If the saturation point occurs before system utilization reaches 100%, it is likely that another bottleneck is being aggravated by the application. For example, the application might be

creating Java objects that are causing excessive garbage collection mark phase bottlenecks in Java. You might notice these bottlenecks because only one processor is being used at a time on multi-processor systems. On uniprocessor systems, you will not notice the symptom but will notice only the problems that the symptom causes.

Figure 14-3 shows an example throughput curve.

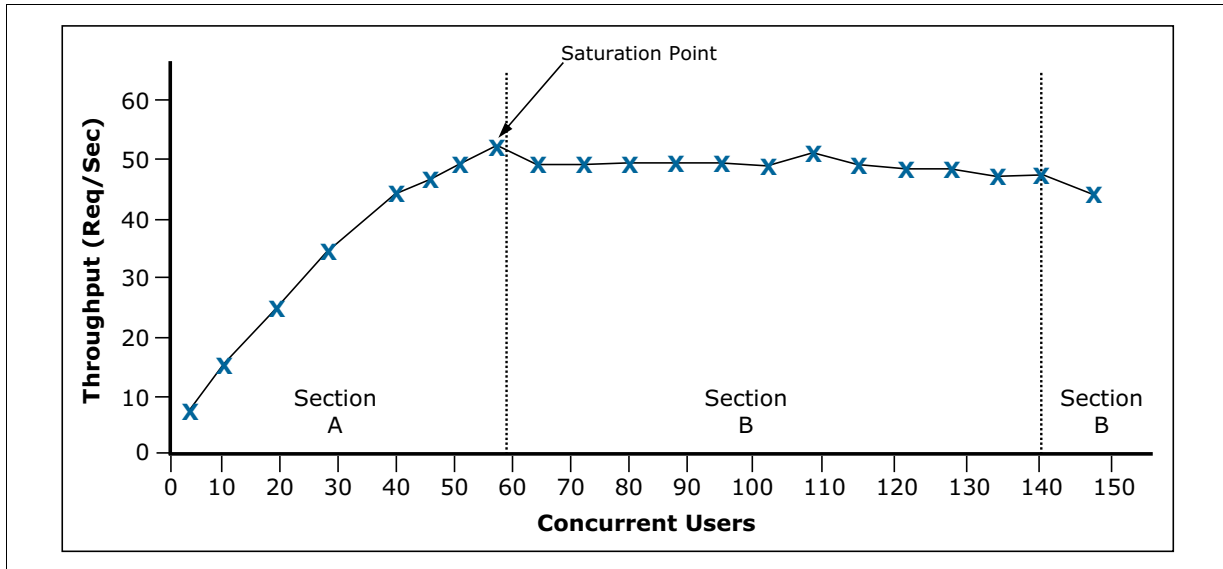


Figure 14-3 Throughput curve

In Figure 14-3, Section A contains a range of users that represent a light user load. The curve in this section illustrates that as the number of concurrent user requests increase, the throughput increases almost linearly with the number of requests. You can interpret this increase to mean that at light loads, concurrent requests face little congestion within the WebSphere Application Server system queues.

In the heavy load zone, Section B, as the concurrent client load increases, throughput remains relatively constant. However, the response time increases proportionally to the user load. That is, if the user load is doubled in the heavy load zone, the response time doubles.

In Section C (the buckle zone), one or more of the system components became exhausted and throughput degrades. For example, the system might enter the buckle zone if the network connections at the web server exhaust the limits of the network adapter or if the requests exceed operating system limits for file handles.

14.2.6 Estimating web container and ORB thread pool initial sizes

Tuning WebSphere thread pools is a critical activity that has impact on the response time of the applications that run on WebSphere. Although allocating less than an optimum number of threads can result in higher response times, allocating higher than the optimum number can result in over use of system resources, such as CPU, databases, and other resources. A load test can determine the optimum number for thread pool sizes.

When tuning for performance, you are most frequently tuning web container thread and ORB thread pools. As a starting point, you need to know the possible use cases of your applications and the kind of workload that is generated by these use cases. Suppose that you have an online banking application. One use case for a user is to log in, check account details, transfer money to another account, and then log out. You need to estimate how many web and EJB requests are generated by each activity in this use case. (You might also want to gather statistics about how many data source connections this use case uses and for how long it uses these connections.) Next, you need to calculate an estimate of requests for all use cases and determine the total requests per second for your system.

Suppose that your application receives 50 HTTP requests and 30 EJB requests per second. The expected average response time for web requests is 1 second and for EJB requests the response time is 3 seconds. In this case, you need the following *minimum* number of threads:

- ▶ Web Container Thread Pool: $50/1 = 50$
- ▶ EJB Container Thread Pool: $30/3 = 10$

The *maximum* number of threads that you need depends on the difference between the load in peak time and the average load. Suppose that you have 1.5 times the average load in peak time. Then, you need the following *maximum* number of threads:

- ▶ Web Container Thread Pool = $50 \times 1.5 = 75$
- ▶ EJB Container Thread Pool = $10 \times 1.5 = 15$

14.2.7 WebSphere Plug-in tuning

In the Web server plug-in, the following parameters can be tuned to obtain the best performance in your environment:

- ▶ LoadBalanceWeight

Is a starting "weight". The value is dynamically changed by the Plug-in during runtime. The "weight" of a server (or clone) is lowered each time a request is assigned to that clone. When all weights for all servers drop to 0 or below, the Plug-in has to readjust all of the weights so that they are above 0. The usual configuration for LoadBalanceWeight is to set all application servers with the same value, except one, which is configured with a value off by one (for example: 20, 20, 19).

- ▶ MaxConnections

This parameter is used to determine when a server is "starting to become overwhelmed" but not to determine when to fail-over (mark a server "down").

When a request is sent to an active server, the request is PENDING until handled by the back-end application server. Requests that are handled quickly, will only be PENDING for a short time. Therefore, under ideal conditions, the MaxConnections setting should be set to its default value (-1). However, when application servers are pushed handling a large volume of requests and the handling of incoming requests is taking more time to complete, PENDING requests start to build up. In these cases, the parameter MaxConnections can be used to put a limit on the number of PENDING requests per server. When this limit is met, the server is unavailable for additional new requests but is not marked down.

The optimal value for MaxConnections depends on how quickly the application and appserver respond to each request. Values for this parameter are usually defined in the range of 20 - 100, depending on application response times.

► ConnectTimeout

Defines how long the plug-in waits when trying to open a socket to the back-end application server.

The plug-in attempts to use connection streams that are already open and available to the application server. However, new streams to the application server might be needed. In a normal environment, opening streams does not take long; therefore, the ConnectTimeout parameter can be set at a small value. The usual configuration for this parameter is to define it with a small value, such as 5.

Note: A ConnectTimeout value of 0 means never time-out. In that case, the time-out is left up to the operating system TCP layer, which is NOT ideal. It is better to specify a small positive number for the value.

► ServerIOTimeout

Defines how long the plug-in waits for a response from the application server. Incoming requests are passed by the plug-in to the application server, which processes these requests, sending responses back to the client, through the plug-in.

If the application is quick to respond, use a lower value for ServerIOTimeout, usually 60 seconds. For applications that demand longer time to respond back to requests, use a higher number for ServerIOTimeout. Using a value of 0 means that the Plug-in will NOT time-out the request. This is often NOT ideal. The usual starting value is 60 seconds and adjusts up or down as necessary.

Configuring the ServerIOTimeout with a positive value means that the plug-in will NOT mark the application server down after a ServerIOTimeout happens, which means that the plug-in will continue sending requests to the timed-out application server. Negative values mean that the plug-in WILL mark the application server as down after the ServerIOTimeout criteria is met and fail-over the requests to another application server.

► RetryInterval

Defines how long the plug-in waits before trying again to use an application server that was marked down. The optimal value for RetryInterval depends on the number of application servers in the cluster and the value used for ServerIOTimeout. The following formula can help you determine the maximum RetryInterval value for your plug-in configuration:

- Recommended value = (number of application servers in cluster - 1) x (absolute ServerIOTimeout) - 1

For example, if there are four application servers in a cluster, and the value of ServerIOTimeout is -60, the maximum RetryInterval setting is:

- $(4 - 1) \times (60) - 1 = 179$ seconds or less

Note: Setting RetryInterval to a value higher than the recommended maximum, based on the formula above, can lead to an undesirable situation where all of the application servers in the cluster can be marked down simultaneously, resulting in all requests temporarily failing.

Session affinity

Session affinity means that all requests of the same JSESSIONID are sent to the same application server, regardless of the LoadBalanceWeight.

When using round robin for the LoadBalance option, the following options are available to handle affinity requests:

- ▶ Configure the parameter IgnoreAffinityRequests to true. The affinity requests do not lower the weight. This behavior might cause an uneven distribution of requests across the servers in environments that make use of session affinity.
- ▶ Configure the IgnoreAffinityRequests to false. The weight is lowered by each affinity request, leading to a more balanced round robin environment.

Fail-over

Fail-over occurs when the plug-in marks an application server (or clone) as down and then sends the pending requests to another member of the same cluster. This can happen if the plug-in is unable to open a new connection to the appserver within the ConnectTimeout. Or fail-over can happen if the plug-in already sent the request to the application server but does not receive a response from the application within ServerIOTimeout.

While an application server is marked as down, the plug-in no longer sends any requests to it until the interval defined in the RetryInterval parameter is met. After that, the plug-in checks to see if that application server can be used successfully again, removing the "down" flag if a positive response is received.

For more information about plug-in tuning and configuration, refer to the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg27021301>

To understand how load balancing works in the Web server plug-in, see Understanding IBM HTTP Server plug-in Load Balancing in a clustered environment, at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg21219567>

To understand how fail-over works in the Web server plug-in, see Understanding HTTP plug-in failover in a clustered environment, at the following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg21219808>

14.3 JVM tuning

In this section, we provide information about the JVM parameters that you can tune to increase system performance. You can set the majority of parameters that we explain in this section from the administrative console. Complete the following steps:

1. Navigate to **Servers** → **Application servers** → **<AppServer_Name>**.
2. Under System Infrastructure, expand **Java and Process Management** → **Process definition** → **Java Virtual Machine**.
3. Use the Configuration window to specify the JVM settings.
4. Click **Apply**.

For detailed information about the concepts that we explore in this section, refer to the *Java Diagnostics Guide* at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/topic/com.ibm.java.doc.diagnostics.60/homepage/plugin-homepage-java6.html>

14.3.1 Garbage collection

Garbage collection has the following phases:

- ▶ Mark phase

All live objects are marked in this phase. All reachable objects in the heap are identified, and every other object is treated as garbage. The marking process is also known as *tracing*. The marking phase creates and uses a *mark bit array*, whose bits identify reachable objects in the memory.

- ▶ Sweep phase

The sweep phase uses the mark bit array that is created in the mark phase to identify those chunks of heap storage that can be reclaimed for future allocations. These chunks of heap are added to the pool of free space.

The mark and sweep phases can have threads running in parallel or running concurrently. *Parallel* activities run while application threads are halted, and *concurrent* activities run without stopping the application threads.

- ▶ Compaction phase

In this phase, the resulting set of objects are compacted to remove the spaces between them. Even if the heap has enough total free space, allocation failures can occur because there is not enough contiguous free space. This state is called *fragmentation* of the Java heap.

Compaction is a remedy for fragmentation because it moves objects to the beginning of the heap, with no spaces between them, resulting in a contiguous free space for allocations. When an object is moved, the garbage collection changes all the references to that object, which is a complicated and time expensive process that can result in high pause times. By default, compaction is triggered only in certain circumstances, such as not having enough free space to satisfy the allocation request after sweeping.

Setting the `-Xnoccompactgc` parameter as a JVM argument prevents compaction and the `-Xcompactgc` parameter forces compaction to occur in every global garbage collection cycle.

The different garbage collection policies are compared by:

- ▶ Pause time

In some garbage collection policies, certain activities can acquire exclusive access on the JVM, and all other application threads can pause (such as compaction). The amount of pause time is determined by the size of the heap and by the amount of the garbage objects in the heap.

The JVM uses the following techniques to reduce pause times:

- Concurrent garbage collection
- Generational garbage collection

Pause time has a direct effect on response time of the applications because lower pause times imply higher response times.

- ▶ Throughput

The throughput is the amount of data that is processed by an application in a specific time window. For example, if an application can handle 10 client requests simultaneously and if each request takes one second to process, this site can have a potential throughput of 10 requests per second.

► Response time

The response time is the period from entering a system at a defined entry point until exiting the system at a defined exit point. In a WebSphere Application Server environment, this time is usually the time that it takes from when a request is submitted by a web browser until the response is received at the web browser.

The sections that follow list the garbage collection policies and provide a means to compare the policies. However, to determine the optimum policy for your system, conduct load tests using different policy options. You can set these policies as a JVM argument using `-Xgcpolicy:policy_name` as follows:

`-Xgcpolicy:optthruput`

The optthruput policy

This policy includes parallel mark and sweep phases. The collector can use multiple threads to run these two phases. By default, the number of threads that are used is limited by the number of CPUs. Compaction occurs if required. As the name implies, this policy favors a higher throughput than the other policies. Therefore, this policy is more suitable for systems that run batch jobs. However, because it does not include concurrent garbage collection phases, the pause times can be longer than the other policies.

The optavgpause policy

This policy uses concurrent garbage collection to reduce pause times. Concurrent garbage collection reduces the pause time by performing some garbage collection activities concurrently with normal program execution. This method minimizes the disruption that is caused by the collection of the heap. This policy limits the effect of increasing the heap size on the length of the garbage collection pause. Therefore, it is most useful for configurations that have large heaps. This policy provides reduced pause times by sacrificing throughput.

The gencon policy

This policy is the default garbage collection policy for WebSphere Application Server V8.5, and it is the short name for concurrent garbage collection and generational garbage collection combined. This policy includes both approaches to minimize pause times. The idea behind generational garbage collection is splitting the Java heap into two areas, nursery and tenured. New objects are created in a nursery area and, if they continue to be reachable for the tenure age (a defined number of garbage collections), they are moved into the tenured area.

This policy dictates a more frequent garbage collection of only the nursery area rather than collecting the whole heap as the other policies do. The local garbage collection of the nursery area is called *scavenge*. The gencon policy also includes a concurrent mark phase (not a concurrent sweep phase) for the tenured space, which decreases the pause times of a global garbage collection. The gencon policy can provide shorter pause times and more throughput for applications that have many short-lived objects. It is also an efficient policy against heap fragmentation problems because of its generational strategy.

The balanced policy

You can use the balanced policy only on 64-bit platforms that have compressed references enabled. This policy involves splitting the Java heap into equal sized areas called *regions*. Each region can be collected independently, allowing the collector to focus on the regions that return the largest amount of memory for the least processing effort. Objects are allocated into a set of empty regions that are selected by the collector. This area is known as an *eden space*.

When the eden space is full, the collector stops the application to perform a partial garbage collection. The collection might also include regions other than the eden space, if the collector

determines that these regions are worth collecting. When the collection is complete, the application threads can proceed, allocating from a new eden space, until this area is full. Balanced garbage collection incrementally reduces fragmentation in the heap by compacting part of the heap in every collection. By proactively tackling the fragmentation problem in incremental steps, the balanced policy eliminates the accumulation of work that is sometimes incurred by generational garbage collection, resulting in less pause times.

From time to time, the collector starts a global mark phase to detect if there are any abandoned objects in the heap that are unrecognized by previous partial garbage collections. During these operations, the JVM attempts to use under utilized processor cores to perform some of this work concurrently while the application is running. This behavior reduces any stop-the-world time that the operation might require.

The subpool policy

The subpool policy works on SMP systems (AIX, Linux PPC, and IBM eServer™ zSeries, and z/OS and i5/OS™ only) with 16 or more processors. This policy aims to improve performance of object allocations by using multiple free lists called *subpools* rather than the single free list used by the optavgpause and optthruput policies. Subpools have varying sizes. When allocating objects on the heap, a subpool with a “best fit” size is chosen, as opposed to the “first fit” method used in other algorithms. This policy also tries to minimize the amount of time for which a lock is held on the Java heap. This policy does not use concurrent marking. The subpool policy provides additional throughput optimization on the supported systems.

Comparison of garbage collection policies

Table 14-1 summarizes characteristics and results for different garbage collection policies. Note that some of the results, such as yielding to higher throughput times, is common to more than one policy because all policies that have this result achieve it by using different algorithms. If your aim is to have higher throughput for your system, you might want to test with all the policies that have this result, as listed in Table 14-1, and come up with the best policy for your system.

Table 14-1 Comparison of garbage collection policies

| Policy name | Results |
|-------------|--|
| Optthruputt | Higher throughput, longer response times for applications with GUI |
| Optavgpause | <ul style="list-style-type: none"> ▶ Less pause times, shorter response times for applications with GUI ▶ Shorter pause times for large heaps |
| Gencon | <ul style="list-style-type: none"> ▶ High throughput when application allocates short-lived objects ▶ Shorter response times due to local garbage collection ▶ Effective against heap fragmentation |
| Balanced | <ul style="list-style-type: none"> ▶ Reduces pause times by incremental compactions ▶ Uses NUMA hardware for higher performance ▶ Dynamically unload unused classes and class loaders on every partial collect ▶ Uses under used cores for the global mark phase |
| Subpool | Additional throughput optimization on the supported systems |

14.3.2 Sizing the JVM heap

You can set minimum and maximum heap sizes that are equal, instead of setting a minimum heap size that is smaller than the maximum heap size. This approach prevents heap

expansions and compactions that might occur if the minimum heap size is set smaller than the maximum heap size. However, this approach can have the following drawbacks:

- ▶ Because the minimum heap size is larger, it takes more time for the collections.
- ▶ Because many compactions are omitted, this approach can lead to a more fragmented heap than the minimum less than maximum approach, especially if you are not using generational garbage collection policies.

You can set the initial and maximum heap sizes of your application server through the deployment manager's administrative console:

1. Navigate to **Servers** → **Application servers** → **<AppServer_Name>**.
2. Under System Infrastructure, expand **Java and Process Management** → **Process definition** → **Java Virtual Machine**.
3. Set the initial heap value in the Initial heap size field.
4. Set the maximum heap value in the Maximum heap size field.
5. Click **Apply**.
6. Save the configurations to the master repository, synchronize the nodes, and recycle the application server that was reconfigured.

You can also use the following JVM arguments to set the minimum and maximum heap sizes:

| | |
|-------------------------|----------------------------------|
| -Xms<size> | Sets the initial Java heap size. |
| -Xmx<size> | Sets the maximum Java heap size. |

In your load tests, the best approach is to first set the minimum and maximum values to be equal and then determine the optimum heap size for your system by trying different sizes. After you find the optimum number, set the minimum and maximum values around this number, and then test to find the best performing minimum and maximum values. You can compare performance with these different values to see which settings are optimal for your system.

Note: Avoid setting low values for the initial heap size, especially when the maximum heap is configured to high values. Example: When you have a maximum heap size of **1536MB**, do not set the initial heap size value to 50MB (default value for the initial heap size) because this causes performance degradation due to an excessive load on the garbage collector. As a general rule, setting the initial heap size to at least 50% of the maximum heap can be a good starting point to determine the optimum values for your environment.

14.3.3 Sizing the nursery and tenured space when using the gencon policy

The duration of the nursery collect is determined by the amount of data that is copied from the *allocate* space to the *survivor* space, which are two different regions in the nursery space. (We do not provide information about these regions in detail for the scope of this book.) The size of the nursery does not have an increasing effect on scavenges. Instead, increasing the nursery size increases the time between scavenges, which decreases the amount of data that is copied.

The amount of live data that can be copied to the nursery at any time is limited by the amount of transactions that can work concurrently, which is fixed by the container pool settings. Thus, the nursery size can be fixed to an optimum large value. By default, the nursery size is a quarter of the total heap size or 64 MB, whichever is smaller. The nursery can shrink and expand within this size.

You can use the following parameters to define the nursery size:

- Xmns<size>** Sets the initial size of the new area to the specified value. By default, this option is set to 25% of the value of the **-Xms** option.
- Xmnx<size>** Sets the maximum size of the new area to the specified value. By default, this option is set to 25% of the value of the **-Xmx** option.
- Xmn<size>** Sets the initial and maximum size of the new area to the specified value. This parameter is equivalent to setting both **-Xmns** and **-Xmnx**. If you set either **-Xmns** or **-Xmnx**, you cannot set **-Xmn**.

You can use the following parameters to define the tenured space size:

- Xmos<size>** Sets the initial size of the old (tenure) heap to the specified value. By default, this option is set to 75% of the value of the **-Xms** option.
- Xmox<size>** Sets the maximum size of the old (tenure) heap to the specified value. By default, this option is set to the same value as the **-Xmx** option.
- Xmo<size>** Sets the initial and maximum size of the old (tenured) heap to the specified value. Equivalent to setting both **-Xmos** and **-Xmox**. If you set either **-Xmos** or **-Xmox**, you cannot set **-Xmo**.

For a starting point, you can use **-Xmn** to fix the size of the nursery space to a large enough value. Then, you can treat the tenured space as a Java heap with a non-generational policy, and use the **-Xmos** and **-Xmox** parameters to replace **-Xms** and **-Xmx**.

14.3.4 Using compressed references

The IBM SDK for Java 64-bit stores object references as 64-bit values. The **-Xcompressedrefs** command-line option causes object references to be stored as 32-bit representations, which reduces the 64-bit object size to be the same as a 32-bit object. Because the 64-bit objects with compressed references are smaller than default 64-bit objects, they occupy a smaller memory footprint in the Java heap and improve data locality. Generally, this parameter results in better memory utilization and improved performance. You can perform load testing and compare the results to see whether using this parameter improves performance for your application.

To change to compressed references mode for the JVM:

1. Click **Environment** → **WebSphere variables**. You can also enable this variable on a process basis by clicking **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Java and process management** → **ProcessDefinition** → **Environment entries**.
2. Select a scope based on your desired affected environment.
3. In the name field, click **New**, and specify **IBM_JAVA_OPTIONS**.
4. In the Value field, add or append the **-Xcompressedrefs**.
5. Click **Apply**.
6. For the Network deployment configuration, select **Review**, and select the **Synchronize changes with nodes** option. Click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected.
7. Navigate to **Servers** → **Server types** → **WebSphere Application servers** and then click **Restart** for the affected server.

14.4 Other tuning considerations

This section describes other tuning considerations that can help and guide you when tuning for performance.

14.4.1 Dynamic caching

The dynamic cache service improves performance by caching the output of servlets, commands, and JavaServer Pages (JSP) files. WebSphere Application Server consolidates several caching activities, including servlets, web services, and WebSphere commands, into one service called the *dynamic cache*. These caching activities work together to improve application performance and share many configuration parameters that are set in an application server's dynamic cache service.

The dynamic cache works within an application server Java Virtual Machine (JVM), intercepting calls to cacheable objects, for example, through a servlet's `service()` method or a command's `execute()` method. The dynamic cache either stores the object's output to or serves the object's content from the dynamic cache. Because J2EE applications have high read/write ratios and can tolerate small degrees of latency in the currency of their data, the dynamic cache can create an opportunity for significant gains in server response time, throughput, and scalability.

Refer to the article found at the following website for detailed information about using the dynamic cache service:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/ccontainer_dyn_admin.html

14.4.2 The pass by reference parameter

Passing EJB beans by value can be expensive in terms of resource use because of the forever remote method call. The parameters are copied onto the stack before the call is made. You can use the pass by reference parameter, which passes the original object reference without making a copy of the object, to avoid this impact.

For EJB 2 or later and 3 or later beans, interfaces can be local or remote. For local interfaces, method calls use the pass by reference parameter by default. If the EJB client and EJB server are installed in the same WebSphere Application Server instance and if the client and server use remote interfaces, specifying the pass by reference parameter can improve performance up to 50%.

Note that using the pass by reference parameters helps performance only when non-primitive object types are being passed as parameters. Therefore, int and floats are always copied, regardless of the call model.

Also, be aware that using the pass by reference parameter can lead to unexpected results. If an object reference is modified by the remote method, the change might be seen by the caller. As a general rule, any application code that passes an object reference as a parameter to an enterprise bean method or to an EJB home method must be scrutinized to determine if passing that object reference results in loss of data integrity or in other problems.

To set this value, use the administrative console to complete the following steps:

1. Click **Servers** → **Application servers** → **<AppServer_Name>** → **Container Services** → **ORB Service**.

2. Select the **Pass by reference** parameter.
3. Click **OK** and then click **Apply** to save the changes.
4. Stop and restart the application server.

14.4.3 Large page support

Many operating systems provide the ability to use a larger memory page size than the default memory page size of 4 KB. Having larger page sizes decreases CPU consumption because the CPU has to manage fewer pages. Therefore, Java applications often benefit from using large pages.

To enable large page utilization, configure the value on your operating system. For more information, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/tprf_tuneopsys.html

After you configure the operating system, add `-Xlp<size>` as a generic JVM argument to support large pages.

14.4.4 Application tuning

The most important part of your tuning activities is spent on the application. The majority of performance-related problems are related to application design and development implementations. Only a well-designed application, developed with the preferred practices for programming, can give you good throughput and response times. Although environment-related tuning is important to optimize resource use and to avoid bottlenecks, it cannot compensate for a poorly written application.

Review the application code itself as part of the regular application life cycle to ensure that it is using the most efficient algorithms and the most current APIs that are available for external dependencies. For example, take care to use optimized database queries or prepared statements instead of dynamic SQL statements. To help you in this task, you can optimize the application performance using application profiling.

For additional information about the application design considerations, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/cprf_appdesign.html

14.5 Tools

This section provides information about the tools that can help and guide you when tuning for performance.

14.5.1 Tivoli Performance Viewer

Tivoli Performance Viewer is included with WebSphere Application Server V8.5 and is used to record and display performance data. Using Tivoli Performance Viewer, you can perform the following tasks:

- ▶ Display PMI data collected from local and remote application servers
Summary reports show key areas of contention. It also provides graphical and tabular views of raw PMI data.
- ▶ Provide configuration advice through the performance advisor section
You can formulate tuning advice from gathered PMI and configuration data.
- ▶ Log performance data
Using Tivoli Performance Viewer, you can log real-time performance data and review the data at a later time.
- ▶ View server performance logs
You can record and view data that is logged using Tivoli Performance Viewer in the Integrated Solutions Console.

Refer to *Chapter 16, "Monitoring distributed systems"* on page 553 for detailed information about Tivoli Performance Viewer and the underlying monitoring architecture.

14.5.2 Collecting Java dumps and core files using the administrative console

With WebSphere Application Server V8.5, you can produce a Java dump, Java core, or system dump files directly using the administrative console. These files are useful when you have performance issues that you need to analyze, such as memory, thread, and system behaviors.

To collect Java dump and core files:

1. Click **Troubleshooting** → **Java dumps and cores**.
2. Select the server or servers.
3. Click **System Dump**, **Java Core**, or **Heap Dump**, as appropriate.

14.5.3 IBM Pattern Modelling and Analysis Tool for Java Garbage Collector

IBM Pattern Modelling and Analysis Tool for Java Garbage Collector (PMAT) is a useful tool that analyzes verbose garbage collection trace. It provides crucial information for garbage collection tuning, such as verbose garbage collection analysis, verbose garbage collection graphics, list of errors, and recommendations.

You can enable verbose garbage collection by selecting the option in the JVM Settings window (refer to 14.3, "JVM tuning" on page 506).

For more information about PMAT and to download the tool, go to the following website:

<http://www.alphaworks.ibm.com/tech/pmat>

14.5.4 IBM Monitoring and Diagnostic tools for Java

IBM Monitoring and Diagnostic tools for Java are available using IBM Support Assistant, which is a workbench that offers a single point to access these tools. Using IBM Monitoring and Diagnostic tools for Java, you can analyze applications, garbage collection files, Java heap dump files, and Java core files.

For additional information about IBM Monitoring and Diagnostic tools for Java, refer to the following website:

<http://www.ibm.com/developerworks/java/jdk/tools/>

The following sections describes the components of the IBM Monitoring and Diagnostic tools for Java.

Health center

Health center allows you to monitor the real-time running applications and provides useful information about memory, class loading, I/Os, object allocations, and the system. This tool can help you to identify application memory leaks, I/O bottlenecks, and lock contentions and can help you to tune the garbage collector. The health center is designed to minimize the performance impact of the monitoring.

Memory analyzer

This tool analyzes the Java heap of a JVM process, identifies potential memory leaks, and provides the application memory footprint. Memory analyzer provides a useful object tree browsing function to focus on the objects' interactions and to analyze the memory usage.

Dump analyzer

This tool determines the causes of Java crashes by analyzing the operating system dump. This analysis can be useful to better understand the application failures.

Garbage collection and memory visualizer

This tool helps you analyze and tune the garbage collection, similar to PMAT. It also provides recommendations to optimize the garbage collector and to find the best Java heap settings. Garbage collection and memory visualizer allow you to browse the garbage collection cycles and to better understand the memory behavior of the application.

14.5.5 IBM HTTP server status monitoring page

To monitor IBM HTTP Server, a useful web page called server-status is available. This page is disabled by default, but you can enable it in the httpd.conf configuration file. This web page displays a real-time view of the current IBM HTTP Server state, which includes the following information:

- ▶ The CPU usage
- ▶ The total number of requests served since the server is up
- ▶ The total traffic size since the server is up
- ▶ Some average about the response time
- ▶ The number of requests currently running
- ▶ The number of idle threads
- ▶ And the list of the requests being processed

To enable the server status monitoring page, complete the following steps:

1. Edit the IBM HTTP Server `httpd.conf` file, and remove the comment character (`#`) from the following lines in this file:

```
#LoadModule status_module, modules/ApacheModuleStatus.dll,  
#<Location/server-status>  
#SetHandler server-status  
#</Location>
```

2. Save the changes, and restart IBM HTTP Server.
3. In a web browser, go to: `http://your_host/server-status`. Click **Reload** to update the status. If the browser supports refresh, go to `http://your_host/server-status?refresh=5` to refresh every 5 seconds.

14.5.6 WebSphere performance advisors

When gathering runtime information, the WebSphere performance advisors provide diagnostic advice about the environment. The advisors can determine the current configuration for an application server. Then, by trending the runtime data over time, the advisors provide advice about potential environmental changes that can enhance the performance of the system. The advice is hard coded into the system and is based on IBM preferred practices for tuning and performance.

The advisors do not implement any changes to the environment. Instead, they identify the problem and allow the system administrator to make the decision as to whether to implement. Perform tests after any change is implemented.

WebSphere provides the following types of advisors:

- ▶ Performance and Diagnostic Advisor
- ▶ Performance Advisor in Tivoli Performance Viewer

Performance and Diagnostic Advisor

This advisor is configured through the Integrated Solutions Console. It writes to the application server log files and to the console while in monitor mode. To minimize the performance impact, configure the server to use HPEL instead of using the `systemOut.log` file.

The interface is configurable to determine how often data is gathered and how often advice is written. It offers advice about the following components:

- ▶ J2C Connection Manager:
 - Thread pools
 - LTC Nesting
 - Serial reuse violation
 - Plus various different diagnostic advises
- ▶ Web Container Session Manager:
 - Session size with overflow enabled
 - Session size with overflow disabled
 - Persistent session size
- ▶ Web Container:
 - Bounded thread pool
 - Unbounded thread pool

- ▶ Orb Service:
 - Unbounded thread pool
 - Bounded thread pool
- ▶ Data Source:
 - Connection pool size
 - Prepared statement cache size
- ▶ Java virtual machine (JVM):
 - Memory leak detection

If you need to gather advice about items that are outside of this list, use the Performance Advisor in Tivoli Performance Viewer.

The Performance Advisor in Tivoli Performance Viewer

This advisor is slightly different from the Performance and Diagnostic Advisor. The Performance Advisor in Tivoli Performance Viewer is invoked only through the Tivoli Performance Viewer interface of the Integrated Solutions Console. It runs on the application server that you are monitoring, but the refresh intervals are based on selecting refresh through the console. Also, the output is routed to the user interface instead of to an application server output log file. This advisor captures data and gives advice about more components. Specifically, this advisor can capture the following types of information:

- ▶ ORB service thread pools
- ▶ Web container thread pools
- ▶ Connection pool size
- ▶ Persisted session size and time
- ▶ Prepared statement cache size
- ▶ Session cache size
- ▶ Dynamic cache size
- ▶ JVM heap size
- ▶ DB2 performance configuration

Running the Performance Advisor in Tivoli Performance Viewer requires resources and can impact performance. Use it with care in production environments.

Refer to *Chapter 16, "Monitoring distributed systems"* on page 553 for detailed information about performance advisors.

14.6 Case Study

A complete performance tuning case study for WebSphere Application Server, is at the following website:

http://www.ibm.com/developerworks/websphere/techjournal/0909_blythe/0909_blythe.html

The case study describes how different configurations behave, does a comparison between the different settings, and shows how to extract valuable data from the tests. This information can help you apply the optimum configuration values into application servers settings.



Clustering, workload management, and high availability

Clustering is a fundamental approach for accomplishing high availability. IBM WebSphere Application Server Network Deployment offers a built-in application server clustering function and the high availability (HA) manager for protecting WebSphere singleton services. Clustering application servers provide workload management (WLM) and failover for applications that reside on the application server cluster.

This chapter provides information about the three closely related topics of clustering:

- ▶ Clustering
- ▶ Workload management
- ▶ High availability and failover

This chapter also provides an overview of the concepts and configuration steps and investigates how features and components work together to establish high availability and workload management.

15.1 Clustering

A *cluster* is a group of servers that are managed together. In the case of WebSphere Application Server, it is a logical grouping of application server processes, running the same set of enterprise applications and the weighted workload capacity that is associated with these servers.

It also provides two key aspects for enterprise applications infrastructure:

- ▶ Scalability

Scalability provides enterprise applications with the ability to properly handle an increase in load volumes and achieve better throughput by making use of more infrastructure resources.

- ▶ High availability

High availability means the capacity that enterprise applications have to continue to process work and avoid impacts in the occurrence of failure of one or several components. It is achieved by making use of redundancy, therefore eliminating single points of failure (SPOF) in the environment.

15.1.1 Clustering for scalability and failover

Clustering can be achieved using vertical scaling, horizontal scaling, or a combination of both approaches.

Vertical scaling

In *vertical scaling*, as shown in Figure 15-1, multiple cluster members for an application server are defined on the same physical machine or node, which might allow the machine's processing power to be more efficiently allocated.

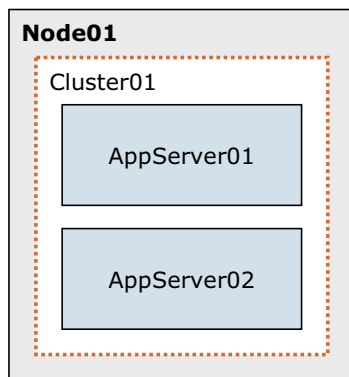


Figure 15-1 Vertical scaling

Horizontal scaling

In *horizontal scaling*, as shown in Figure 15-2 on page 521, cluster members are created on multiple physical or virtual machines. This configuration allows a single WebSphere application to run on several machines while still presenting a single system image, making the most effective use of the resources of a distributed computing environment.

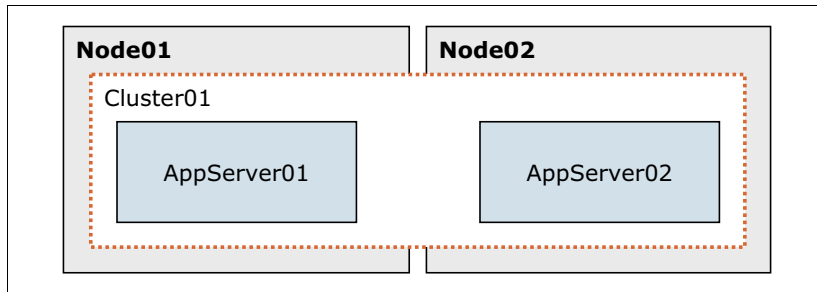


Figure 15-2 Horizontal scaling

Failover is another important benefit of horizontal scaling because it can handle application server process failures and hardware failures (or maintenance) without significant interruption to client services. If a machine becomes unavailable, its workload can be routed to other machines that contain cluster members.

Combining vertical and horizontal scaling

WebSphere applications can combine horizontal and vertical scaling to reap the benefits of both scaling techniques, as shown in Figure 15-3.

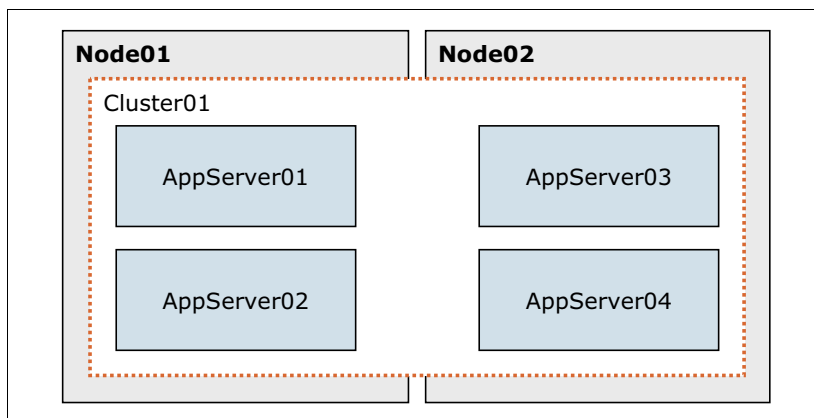


Figure 15-3 Hybrid scaling using both vertical and horizontal topologies

15.1.2 Intelligent Management

WebSphere Application Server Network Deployment provides the Intelligent Management feature that extends the quality of services provided by your middleware infrastructure. It provides the environment with application server virtualization capabilities and a host of advanced operational facilities, such as performance virtualization. This allows WebSphere to enhance operational efficiency by providing continuously available application server environments that can service high-volume transactional workloads with linear scalability and high availability.

The main features in Intelligent Management are:

- ▶ Intelligent routing: Provides the ability to prioritize the routing of requests to business critical applications, following administrator-defined rules.
- ▶ Health management: Provides the ability to specify conditions to be monitored and the corresponding corrective actions to be taken when the conditions are observed.

- ▶ Performance management: Provides a self-optimizing middleware infrastructure based on dynamic clusters, that can automatically scale up and down the number of application server instances to meet optimum response times and business needs.

Autonomic managers

Intelligent Management introduces autonomic capabilities into your infrastructure. These capabilities are delivered through a set of components known as *autonomic managers*. Autonomic managers monitor performance and health statistics through a series of sensors and turn various internal control knobs to optimize system performance and perform traffic shaping.

The Intelligent Management component includes the following autonomic managers:

- ▶ Autonomic request flow manager (ARFM) to perform traffic shaping of incoming requests
- ▶ Dynamic workload controller to dynamically adjust server weights to even out and minimize response times across the cluster
- ▶ Application placement controller to manage an application's location within a node group in the WebSphere cell
- ▶ On demand configuration manager to maintain cell topology information, track updates applied into the cell topology, and keep the ARFM and other controllers aware of its environment

The on demand router

The on demand router (ODR) server is an intelligent Java-based HTTP proxy server and SIP proxy server built on the WebSphere run time. In your environment's architecture, it is a component that sits in front of your application servers and is responsible for managing the flow of requests into the WebSphere environment. The ODR supports health, application edition, and performance management features.

The ODR is completely asynchronous, high performing, and scalable. It can be clustered for high availability, and it handles the queuing and dispatching of requests according to operational policies. As well as functioning similarly to the web server plug-in for WebSphere Application Server, it uses session affinity for routing work requests to application servers. This allows the ODR to queue requests for less important applications so that requests from more important applications are handled quickly. Another important aspect is that it can protect systems from becoming overloaded by limiting the rate at which traffic is forwarded to application servers.

In WebSphere Application Server V8.5.5, the ODR function can be implemented in the web server tier instead of as an ODR server. For more information about this topology, see Chapter 13, "Intelligent management" on page 469.

15.1.3 Dynamic cluster

A *dynamic cluster* is a server cluster that enables application server virtualization. It uses weights and workload management to balance the workloads of its cluster members dynamically, based on performance information that is collected from the cluster members. This allows the application environment to dynamically expand and contract depending on the amount of workload it needs to handle at any given time.

The autonomic managers, including the application placement controller and the dynamic workload manager maximize the use of your computing resources. Dynamic clusters are required for many of the Intelligent Management autonomic functions, including high availability and service policies.

WebSphere provides two options for adding members to a dynamic cluster:

- ▶ **Automatically define cluster members with rules**
By automatically defining cluster members with rules, you can create a subexpression that automatically selects nodes to host dynamic cluster members based on different node properties. This subexpression is called a membership policy.
- ▶ **Manually define cluster members**
When you manually define cluster members, you statically define which servers are cluster members by selecting servers to add to the cluster. You use this option instead of the membership policy for the following reasons:
 - You have an existing static cluster that you want to convert to a dynamic cluster.
 - You are using assisted life-cycle management servers, which cannot be created from the administrative console. With this option, you create representations of the servers as cluster members, which must be homogeneous. That is, be all of the same server type, same version of the middleware software and the same enterprise applications.

Keep in mind the following key points about dynamic clustering:

- ▶ Dynamic clusters grow and shrink depending on the workload demand.
- ▶ Dynamic clusters work closely with the ODR to ensure even distribution of workload among the cluster members.

WebSphere Application Server V8.5 provides support for the following types of dynamic clusters:

- ▶ WebSphere application server
- ▶ Apache server
- ▶ Custom HTTP server
- ▶ ODR
- ▶ PHP server
- ▶ WebSphere Application Server Community Edition server

15.1.4 Static cluster versus dynamic cluster

WebSphere Application Server Network Deployment V8.5 supports the creation of both static and dynamic clusters. Despite the fact that both approaches provide workload balancing capabilities, their configuration differs from each other in the sense that dynamic clusters are controlled by autonomic managers that can optimize the performance of the cluster. See Table 15-1 for the differences between static and dynamic clusters.

Table 15-1 Static clusters versus dynamic clusters

| Characteristic | Static clusters | Dynamic clusters |
|----------------|--|--|
| Membership | You must manually add application servers into a static cluster. | Membership is based on policy rules. For example, you can define a nodegroup that contains the nodes on which a dynamic cluster can add or remove cluster members. |

| Characteristic | Static clusters | Dynamic clusters |
|----------------------------|---|--|
| Management | Management is done manually by an operator or administrator, to stop and start any application server instances within the cluster. | A dynamic cluster can stop and start instances as required when configured to operate in automatic mode. It also provides a supervised mode in which the administrator is advised to perform runtime tasks to stop and start server instances. Finally, there is also a manual mode, which will always require the intervention of an operator to stop and start the application server instances. |
| Application server weights | The dynamic workload manager is disabled by default. Therefore, you explicitly assign a weight value to each application server instance. | The dynamic workload manager is enabled by default and assigns weights to the application server instances. |

15.1.5 Creating a static application server cluster

To create a new static cluster from the administrative console:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters** and then click **New**.
2. Enter basic cluster information:
 - a. Enter a cluster name.
 - b. Select one of the following options:
 - The Prefer local option routes EJB requests to the same host as the host of the requesting EJB client.
 - The Configure HTTP session memory-to-memory replication option creates a replication domain for the cluster.
 - c. Click **Next**.
 - d. Create the first cluster member, as shown in Figure 15-4 on page 525. Enter a member name. Then, select the node, and enter the weight.
 - e. Select the **Generate unique HTTP ports** option to ensure that the server is created with unique port numbers.
 - f. The “Select how the server resources are promoted in the cluster” drop-down menu defines in which scope resources, such as data sources, are initially created in the cluster. Possible options are cluster, server, and both.
 - g. Create the first cluster member based on one of the following options:
 - Using an application server template
 - Using an existing application server
 - Converting an existing server
 - Creating an empty cluster
 - h. Click **Next**.

Create a new cluster

Step 1: Enter basic cluster information

→ **Step 2: Create first cluster member**

Step 3: Create additional cluster members

Step 4: Summary

Create first cluster member

The first cluster member determines the server settings for the cluster members. A server configuration template is created from the first member and stored as part of the cluster data. Additional cluster members are copied from this template.

* Member name

Select node

* Weight
 (0..100)

Generate unique HTTP ports

Select how the server resources are promoted in the cluster.

Select basis for first cluster member:

Create the member using an application server template.

Create the member using an existing application server as a template.

Create the member by converting an existing application server.

None. Create an empty cluster.

Figure 15-4 Creating the initial cluster member

3. Add additional cluster members by specifying the following information for each new member:
 - Member name
 - Node
 - Weight

To add a new member, click **Add Member**, as shown in Figure 15-5 on page 526. Repeat steps 3a-e to add other members.

When you are finished adding new members, click **Next**.

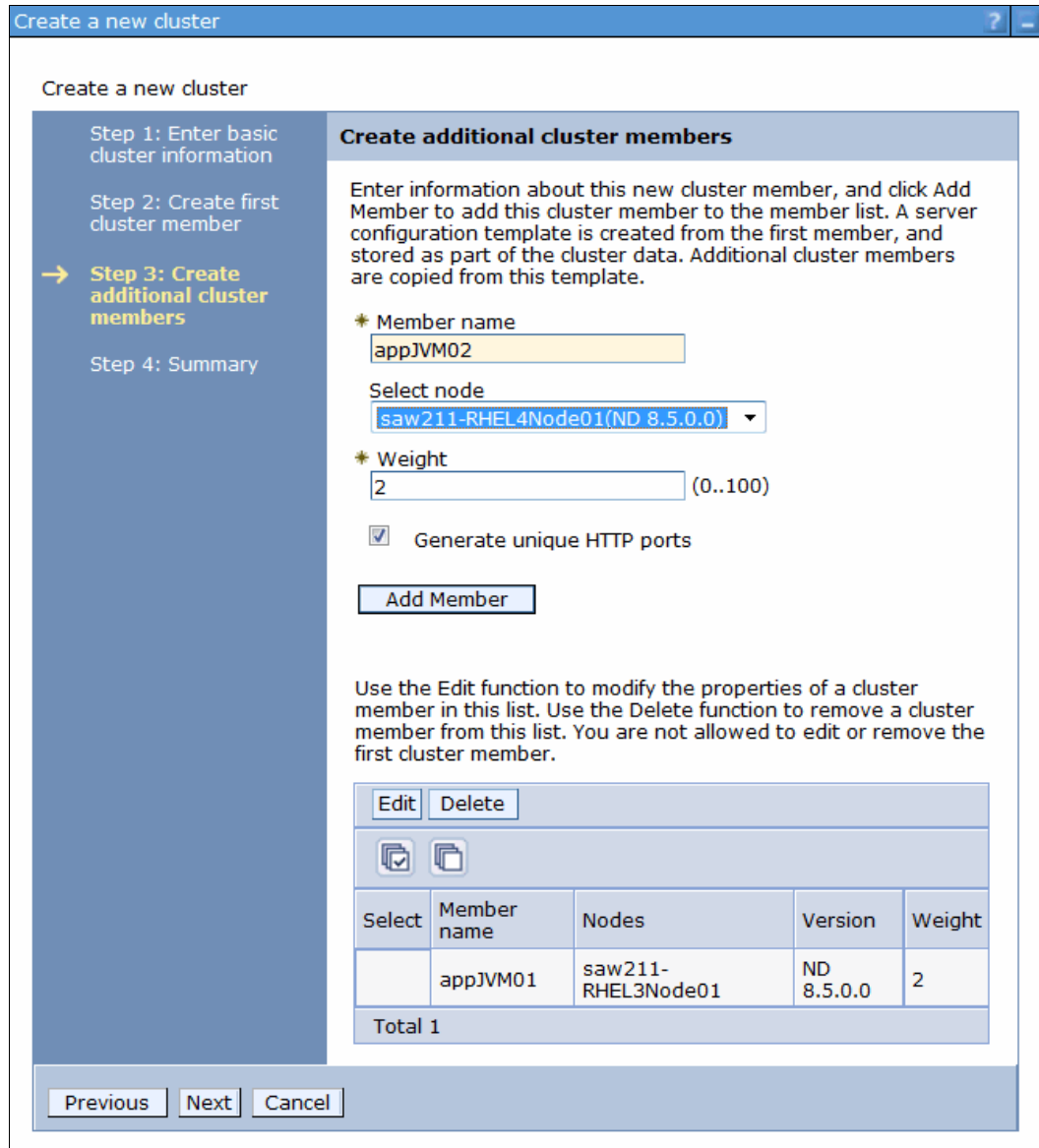


Figure 15-5 Creating new cluster members

- In the summary window, click **Finish** to create the cluster.

15.1.6 Creating a dynamic application server cluster

To create a new dynamic cluster, you must first verify that the application placement controller feature is enabled. It is enabled by default in WebSphere Application Server Network Deployment V8.5. From the deployment manager's administrative console, navigate to **Operational policies** → **Autonomic managers** → **Application placement controller** to enable this functionality, as shown in Figure 15-6.

Application Placement Controller

Use this page to configure the application placement controller. The application placement controller manages dynamic clusters in supervised mode and automatic mode.

Configuration **Runtime**

General Properties

Enable

Approval timeout: Minutes

Server operation timeout: Minutes

Minimum time between placement change: Minutes

Additional Properties

Elasticity Operation

Elasticity Custom Actions

Custom Properties

Elasticity Mode

Enable Elasticity Operation

Supervised

Elasticity operations time out

Minutes

Apply OK Reset Cancel

Figure 15-6 Application Placement Controller

WebSphere Application Server Network Deployment V8.5 essentially provides two options to create a new dynamic cluster:

- ▶ Creating a new dynamic cluster with automatic membership.
- ▶ Creating a new dynamic cluster with manual membership, which allows for the conversion of static clusters into a dynamic clusters.

Creating a new dynamic cluster with automatic membership

To create a cluster from the administrative console, complete the following steps:

1. Click **Servers** → **Clusters** → **Dynamic clusters** and then click **New**.
2. Set the type of the new dynamic clusters that are being created to **WebSphere application server**, and click **Next**.

3. Configure the new cluster with automatic membership by selecting **Automatically define cluster members with rules**, as shown in Figure 15-7.
 - a. Enter a cluster name.
 - b. Enable the options best suited for your environment:
 - Prefer local enabled, enabled by default, indicates that Enterprise Java Beans (EJB) requests are routed to the same host as the host of the requesting EJB client.
 - Create a replication domain for this cluster. Specifies that a replication domain is created for the dynamic cluster. The name of the replication domain is set to the name of the dynamic cluster that uses this domain.
 - c. Click **Next**.

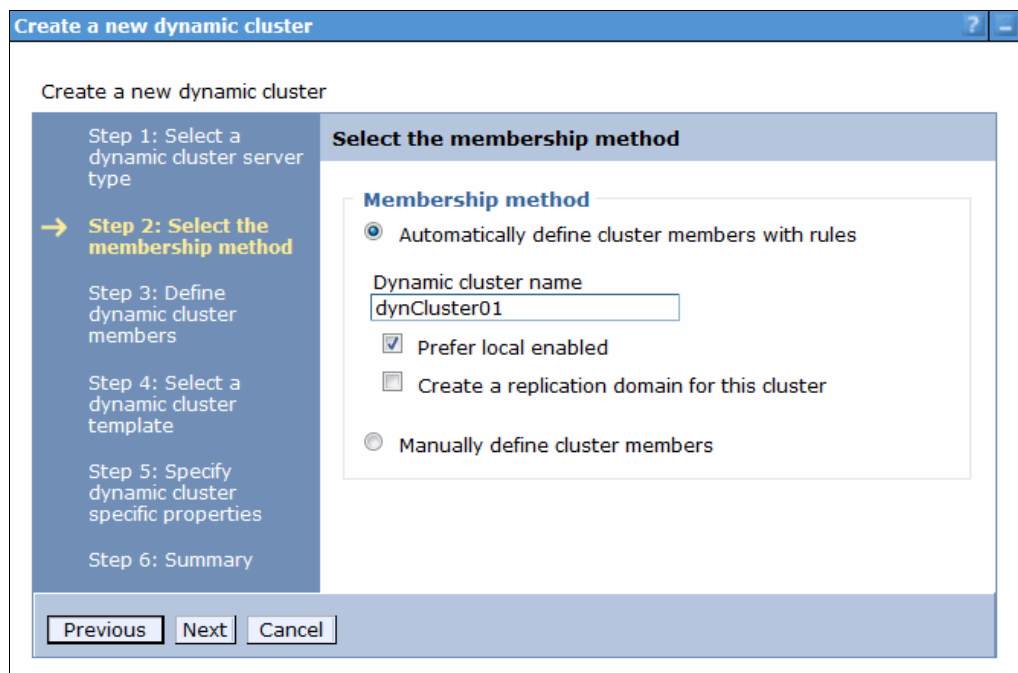


Figure 15-7 Setting up the dynamic cluster membership method

4. Define dynamic cluster members:
 - a. By default, the value of the Membership policy field is set to `node_nodegroup = 'DefaultNodeGroup'`, which means that all of the nodes belonging to this node group are members of the new dynamic cluster. To modify the membership policy, click the **Subexpression builder** link under Edit rule.
 - b. From the membership policy Subexpression builder window, modify the initial policy by setting the following options, as shown in Figure 15-8 on page 529:
 - Define the appropriate value for Logical operator as either AND or OR.
 - Set the expression's operand value, which can be Nodegroup, Node name, Node host name and Node property.
 - Choose the desired expression Operator.
 - Enter the appropriate information in the value field, matching the chosen operand value, for the expression's operation.
 - Click **Generate subexpression**.

- Click **Append**, which modifies the value of the Membership policy field, as shown in Figure 15-9.

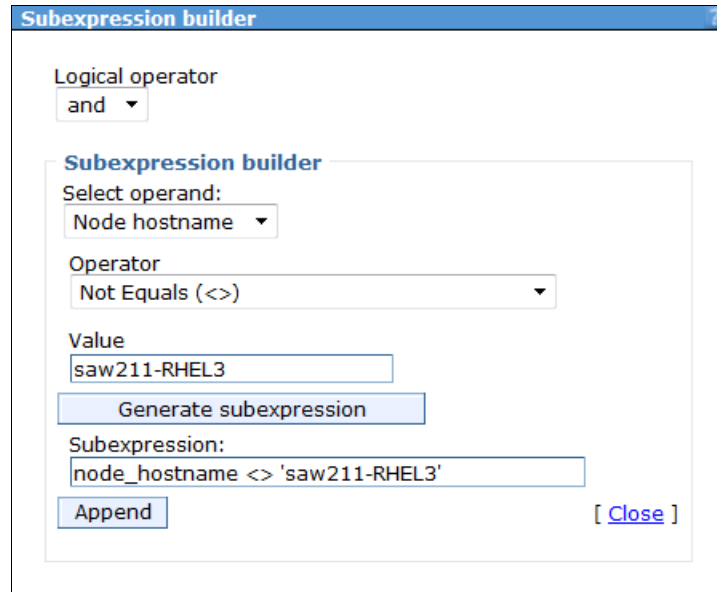


Figure 15-8 Subexpression builder for cluster membership policy

- To see where the new dynamic cluster members will reside, click the **Preview membership** link.
- Click **Next**.

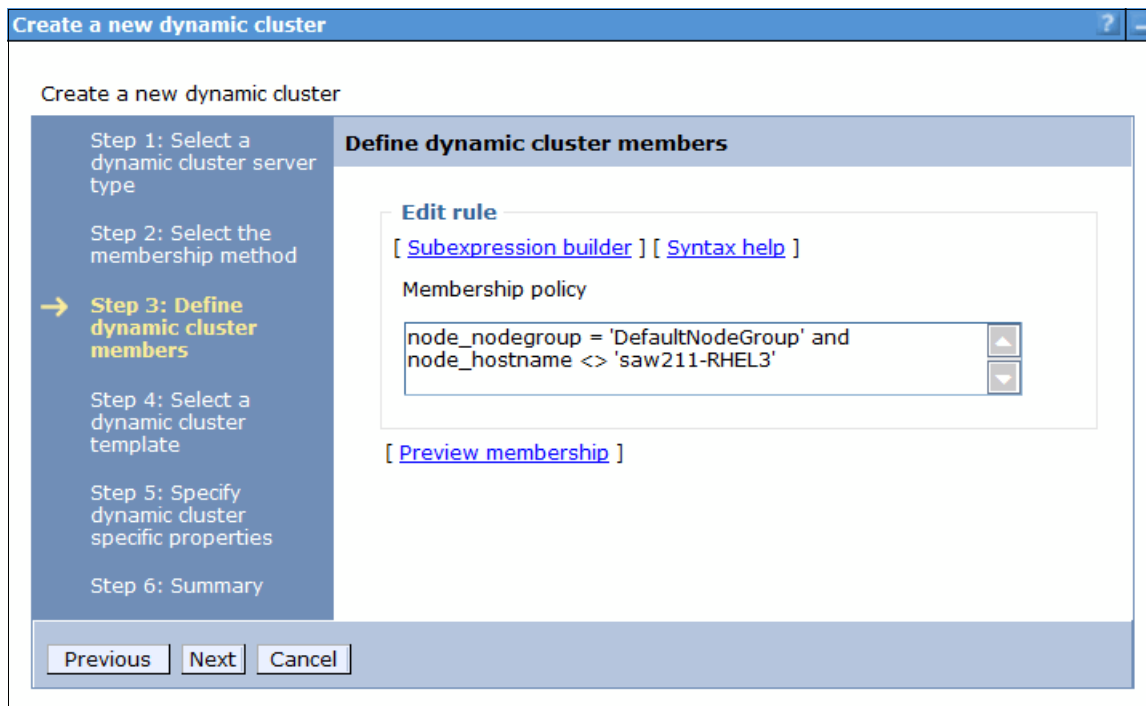


Figure 15-9 Dynamic cluster membership policy definition

5. Create the cluster members based on one of the following options:
 - a. Using an application server template:

- defaultXD, is the default template for the Intelligent Management.
 - default, is identical to the defaultXD template except that the Web container thread pool maximum size is 50 while it is 500 in the defaultXD template.
 - DeveloperServer, is the default template used when you are developing your applications.
 - Using an existing application server as template.
- b. Specifying a target core group for the new cluster members.
 - c. Click **Next**.
6. Complete the dynamic cluster configuration by specifying its runtime and creation-specific properties, as shown in Figure 15-10 on page 531:
- The minimum number of running instances for the dynamic cluster.
 - The maximum number of cluster instances that can be started at any given time.
 - The vertical stacking, or number of cluster instances that will be created per node.
 - The isolation mode, to specify if you want the dynamic cluster to run on the same nodes as other instances of different dynamic clusters, or if you want the dynamic cluster to be the only instances running on a node. Options for isolation mode are:
 - No isolation: Target nodes can be shared between different dynamic clusters. This option is the default.
 - Strict isolation: Target nodes are dedicated to a specific dynamic cluster, running only application server instances of it.
 - Associate with an isolation group: Requires the creation of an isolation group, which determines which dynamic clusters can share the same target nodes. If only one cluster is added into an isolation group, the behavior will be the same as the behavior of the strict isolation.
- d. Click **Next**.

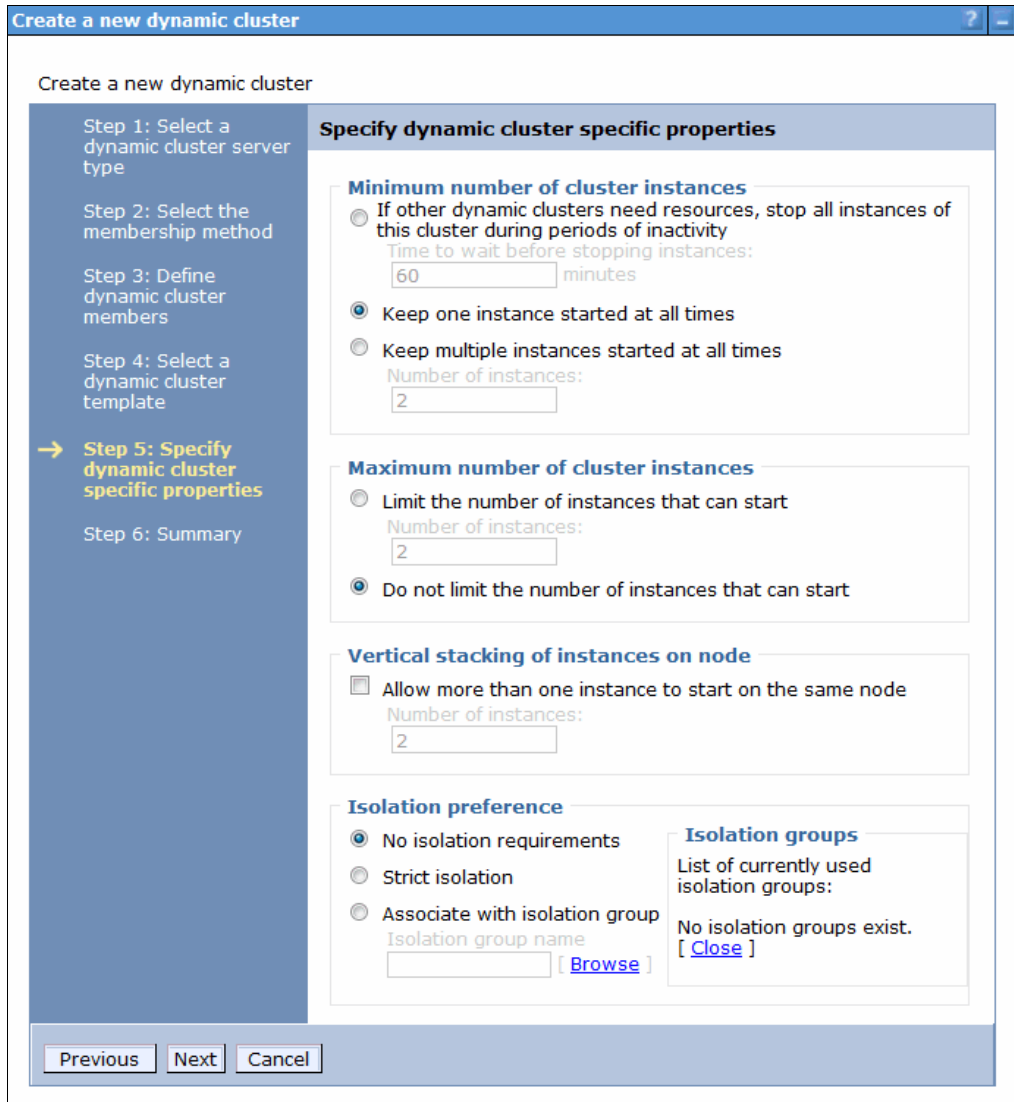


Figure 15-10 Configuring dynamic cluster specific properties

7. In the summary window, click **Finish** to create the cluster.

Creating a new dynamic cluster by manually defining members

To create a cluster from the administrative console:

1. Click **Servers** → **Clusters** → **Dynamic clusters** and then click **New**.
2. Set the type of the new dynamic clusters that is being created to **WebSphere application server**, and click **Next**.
3. For membership method, set the option to **Manually define cluster members**.
4. Click **Next**.
5. Select the pre-existing static cluster that will be converted into a dynamic cluster from the Convert from an existing static cluster drop-down list, and click **Next**.
6. Complete the dynamic cluster configuration by specifying its runtime and creation-specific properties:
 - The minimum number of running instances for the dynamic cluster.

- The maximum number of cluster instances that can be started at any given time.
 - The isolation mode, to specify if you want the dynamic cluster to run on the same nodes as other instances of different dynamic clusters, or if you want the dynamic cluster to be the only instances running on a node. Options for isolation mode are:
 - No isolation: Target nodes can be shared between different dynamic clusters. This option is the default.
 - Strict isolation: Target nodes are dedicated to a specific dynamic cluster, running only application server instances of it.
 - Associate with an isolation group: Requires the creation of an isolation group, which determines which dynamic clusters can share the same target nodes. If only one cluster is added into an isolation group, the behavior is the same as the strict isolation.
 - e. Click **Next**.
7. In the summary window, click **Finish** to create the cluster.

Note: Dynamic cluster isolation can be used to isolate applications deployed within the same cell. For example, you might want to isolate critical applications that an external customer uses from internal applications, which can tolerate some instability.

15.1.7 Setting the operational mode for dynamic clusters

Dynamic clusters act differently depending on the operating mode that is defined for them. To configure the operational mode for dynamic clusters in WebSphere Application Server Network Deployment V8.5, execute the following steps:

1. Login to the deployment manager's administrative console with an administrative ID.
2. Navigate to **Servers** → **Clusters** → **Dynamic clusters**.
3. Select the desired dynamic cluster from the existing clusters list.
4. Set the desired operational mode to:
 - Manual: In this mode, the dynamic cluster is no different from a static cluster because all cluster instances need to be manually started through administration or operator intervention.
 - Supervised: In this mode, the environment provides runtime information to system administrators on whether or not actions, such as stopping or starting application server instances, need to be taken. The administrator, in turn, can manage this run time task from the administrative console by selecting **System Administration** → **Task management** → **Runtime tasks**.
 - Automatic: In this mode, the environment automatically takes corrective actions, such as starting and stopping application server instances.
5. Click **Set Mode**.

15.2 Workload management

The ability to route a request to any server in a group of clustered application servers allows the servers to share work and to improve throughput of client requests. Requests can be distributed evenly to servers to prevent workload imbalances in which one or more servers

has idle or low activity while other servers are overburdened. This load balancing activity is a benefit of workload management.

Thus, this configuration ensures that each machine or server in the configuration processes a fair share of the overall client load that is being processed by the system as a whole. In other words, it is inefficient to have one machine overloaded while another machine is mostly idle. If all machines have roughly the same capacity (for example, CPU power), each machine can process a roughly equal share of the load. Otherwise, consider a provision for the workload to be distributed in proportion to the processing power that is available on each machine.

Using weighted definitions of cluster members allows nodes to have different hardware resources and still participate in a cluster. The weight specifies that the application server with a higher weight is more likely to serve the request faster, and workload management consequently sends more requests to that node.

With several cluster members available to handle requests, it is more likely that failures will not negatively affect throughput and reliability. With cluster members distributed to various nodes, an entire machine can fail without any application downtime. Requests can be routed to other nodes if one node fails. Clustering also allows for maintenance of nodes without stopping application functionality.

This section provides an overview of WebSphere WLM. The available WLM policies and how requests are distributed among available servers is described in great detail in *WebSphere Application Server V6 Scalability and Performance Handbook*, SG24-6392.

15.2.1 Dynamic workload management

Dynamic workload management is a feature of the ODR that applies the same principles as workload management (WLM). Principles, such as routing based on a weight system, establish a prioritized routing system. With dynamic workload management, the system can dynamically modify the weights to stay current with the business goals as compared to manually setting static weights in WLM. It also balances requests across the available nodes to regulate response times.

For more information about dynamic workload management, refer to 5.4, “Dynamic workload management” in the *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022, at the following website:

<http://www.redbooks.ibm.com/abstracts/sg248022.html?open>

15.2.2 Components that can be workload managed

In a typical WebSphere Application Server topology, the components that we describe in this section can be workload managed.

HTTP requests and web servers

An IP sprayer component, such as the Edge Components Load Balancer or a network appliance, can be used to perform load balancing and workload management functionality for incoming HTTP requests.

HTTP requests and static clusters

When incoming requests reach an HTTP server, the web server plug-in, which runs in-process with the HTTP server, decides how to handle these service requests. While some requests for static content can be serviced directly by the HTTP server, any requests for dynamic content, and some requests for static content, are sent to the back-end application servers. We refer to this process as *plug-in WLM*, as illustrated in Figure 15-11. For these WebSphere requests, high availability for the web container becomes an important piece of the failover solution.

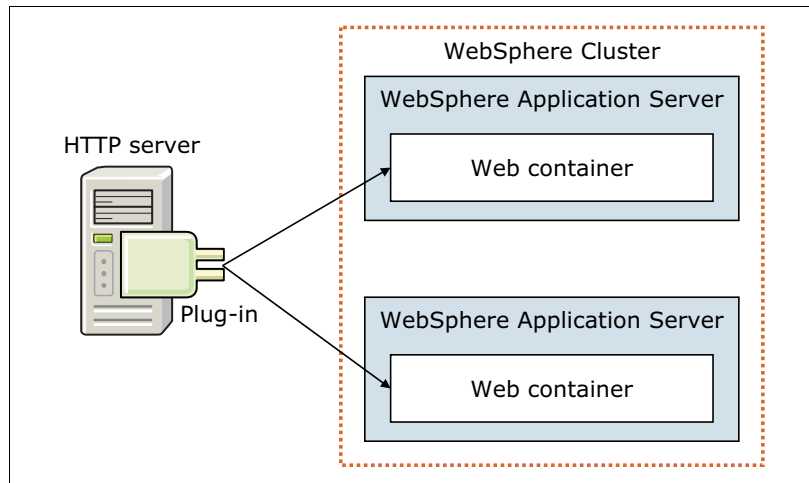


Figure 15-11 Plug-in workload management

WebSphere provides the following load balancing options:

- ▶ Round-robin

This routing is based on the weight that is associated with the cluster members. If all cluster members have identical weights, the plug-in sends equal requests to all members of the cluster, assuming no strong affinity configurations. If the weights are scaled in the range 0 - 20, the plug-in routes requests to those cluster members with the higher weight value more often. No requests are sent to cluster members with a weight of zero (0) unless no other servers are available. Round-robin is the default load balance policy.

Use the following formula as a guideline for determining routing preference:

$$\% \text{ routed to Server1} = \text{weight1} / (\text{weight1} + \text{weight2} + \dots + \text{weightn})$$

Where there are n cluster members in the cluster.

- ▶ Random

The plug-in picks a member of the cluster randomly.

The load balancing options are impacted by the session affinity. After a session is created at the first request, all the following requests have to be served by the same member of the cluster. The plug-in retrieves the application server that serves the previous request by analyzing the session identifier and tries to route to this server. We describe session management concepts in detail in Chapter 28, "Session management" on page 961.

HTTP requests and dynamic clusters

With the dynamic workload management features, the ODR becomes an important figure in workload management. It handles queuing and dispatching of incoming requests to the dynamic application server clusters, according to defined operational policies for optimum results and performance.

Figure 15-12 illustrates how the ODR server distributes traffic between application servers in a dynamic cluster.

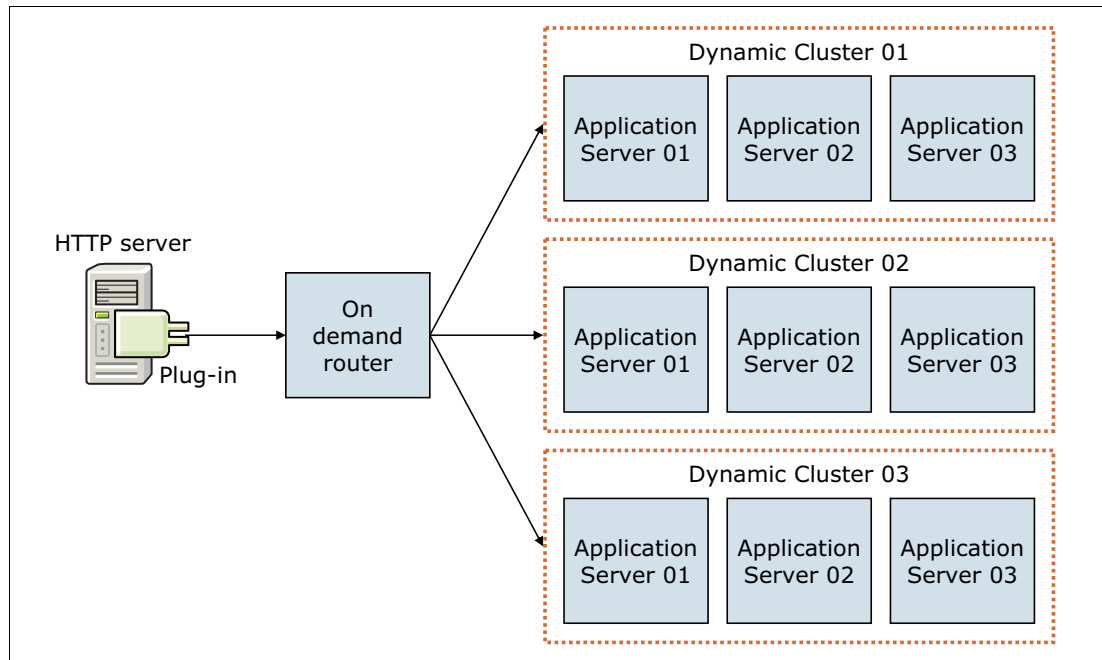


Figure 15-12 ODR workload management

The components in this topology:

- ▶ The IBM HTTP Server is placed as an entry point of the system, with the plug-in configured to route requests that are destined for an application server, to the ODR. The plug-in file is generated by the ODR.
- ▶ The ODR is placed between the IBM HTTP Server and the application servers. With the help of the autonomic managers, the ODR categorizes and prioritizes the incoming work before routing it to the appropriate application servers in the dynamic clusters.
- ▶ Each dynamic cluster consists of multiple application servers on multiple nodes. The workload for an application is spread across the servers. If one server fails, incoming work for the application is simply routed to other servers in the dynamic cluster.

When using this topology with a web server in the DMZ and the ODR, you must generate a special web server plug-in. This new plug-in replaces the one that is usually generated for the web servers and routes any incoming requests to the ODRs instead of directly to the application servers. To eliminate the ODR as a single point of failure, WebSphere Application Server supports ODR static and dynamic clustering. For more information, see 15.4, “ODR server considerations” on page 549.

A similar configuration can be seen in Figure 15-13 on page 536. This configuration uses the web server enabled for Intelligent Management versus the ODR server and eliminates the additional server and network hops used in Figure 15-12.

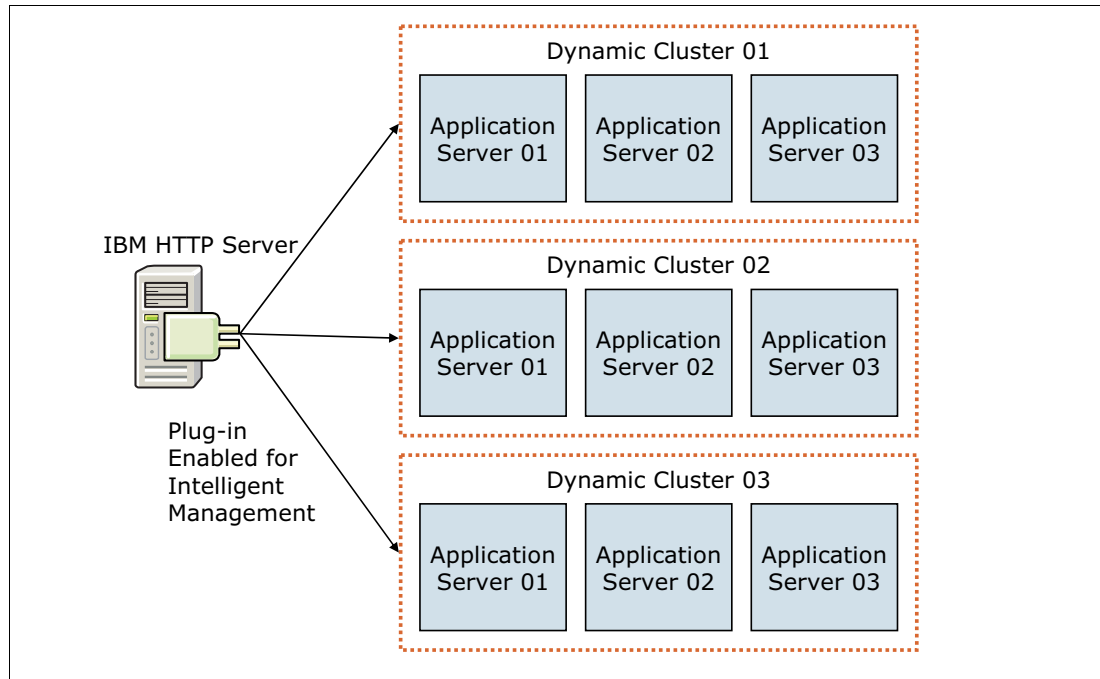


Figure 15-13 Intelligent Management topology with the web server enabled for Intelligent Management

The components in this topology:

- ▶ The IBM HTTP Server is placed as an entry point of the system, with the plug-in enabled for Intelligent Management.
- ▶ With the help of the autonomic managers, the plug-in categorizes and prioritizes the incoming work before routing it to the appropriate application servers in the dynamic clusters.
- ▶ Each dynamic cluster consists of multiple application servers on multiple nodes. The workload for an application is spread across the servers. If one server fails, incoming work for the application is simply routed to other servers in the dynamic cluster.

This topology uses the web server plug-in that is usually generated for the web servers. When the web server is enabled for Intelligent Management, additional information is added to the plug-in that allows it to connect to a REST service to dynamically gather routing information for one or more WebSphere cells.

For more information about the ODR, refer to the following sources:

- ▶ Chapter 13, “Intelligent management” on page 469
- ▶ Chapter 7, “Working with ODRs and autonomic managers” in the *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422, at the following website:
<http://www.redbooks.ibm.com/abstracts/sg247422.html?Open>

RMI/IIOP (EJB) requests

EJB requests can be distributed across multiple EJB containers. When an EJB client makes calls from the web or client container or from outside, the request is handled by the EJB container in one of the clustered application servers. If that server fails, the client request is redirected to another available server. We refer to this process as *EJB WLM*, as illustrated in Figure 15-14 on page 537.

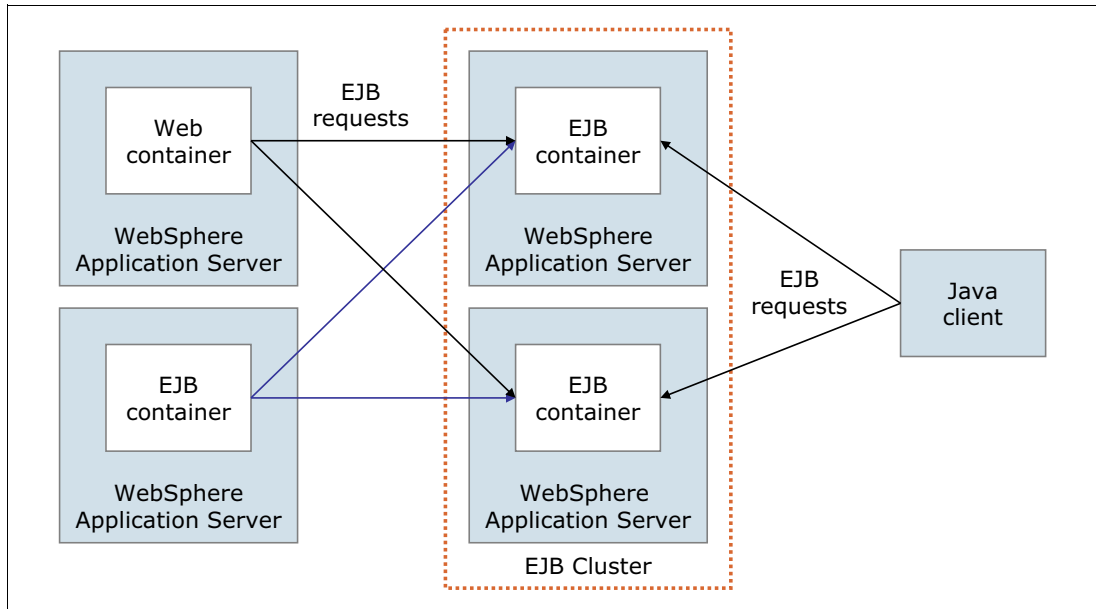


Figure 15-14 Workload management for Enterprise Java Beans

To route the EJB requests, WebSphere provides the following main routing policies:

- ▶ Server weighted round-robin

In this configuration, EJB client requests are routed to available EJB servers in a round-robin fashion based on assigned server weights. The EJB clients can be servlets operating within a web container, stand-alone Java programs using RMI/IIOP, or other EJB beans. The server weighted round-robin routing policy ensures a distribution based on the set of server weights that are assigned to the members of a cluster. For example, if all servers in the cluster have the same weight, the expected distribution for the cluster is that all servers receive the same number of requests. If the weights for the servers are not equal, the distribution mechanism sends more requests to the higher weight value servers than the lower weight value servers.

- ▶ Prefer local

You can also choose that EJB requests are routed preferably to the same host as the host of the requesting EJB client. In this case, only cluster members on that host are chosen (using the round-robin weighted method). Cluster members on remote host are chosen only if a local server is not available.

The following affinity policies can also impact routing:

- ▶ Process affinity

If an EJB is available in the same cluster member as the client, all requests from that client are directed to the EJB in the same JVM process. One of the advantages of this policy is that there is no need for serialization for method calls.

- ▶ Transaction affinity

All the requests from the same transaction are directed to the same cluster member. This policy overwrites all other policies.

15.2.3 Workload management benefits

Workload management provides the following benefits to WebSphere applications:

- ▶ It balances client processing requests, allowing incoming work requests to be distributed according to a configured WLM selection policy.
- ▶ It provides failover capability by redirecting client requests to a running server when one or more servers are unavailable. This redirection improves the availability of applications and administrative services.
- ▶ It enables systems to be scaled up to serve a higher client load than provided by the basic configuration. With clusters and cluster members, additional instances of servers can be added easily to the configuration.
- ▶ It enables servers to be maintained and upgraded transparently while applications remain available for users.
- ▶ It centralizes administration of application servers and other objects.

15.3 High availability and failover

High availability is also known as *resiliency*. High availability is the description of the system's ability to tolerate a certain amount of failures and to remain operational. This section provides information about WebSphere Application Server high availability concepts and features.

15.3.1 Overview

High availability (HA) means that your infrastructure continues to respond to client requests no matter what the circumstances are. Depending on the errors or failures, the infrastructure can run in a degraded mode. HA is achieved by adding redundancy in your infrastructure to support the system when failures occur. Availability impacts both performance and scalability. Depending on your needs, you have to define the level of HA for your infrastructure.

The most common method of describing availability is by the “nines” or the percentage availability for the system. For example, 99.9% of system availability represents 8.76 hours of outage in a single year. Table 15-2 shows the level of availability and the calculated downtime per year.

Table 15-2 Availability matrix

| Availability % | Downtime per year |
|-------------------|-------------------|
| 99% (two 9s) | 87.6 hours |
| 99.9% (three 9s) | 8.76 hours |
| 99.99% (four 9s) | 56.56 minutes |
| 99.999% (five 9s) | 315.36 seconds |

You can calculate availability using the following formula, where MTBF is the mean time between failure and MTTR is the maximum time to recovery:

$$\text{Availability} = (\text{MTBF}/(\text{MTBF} + \text{MTTR})) \times 100$$

Keep in mind that the overall infrastructure is available only if all the components are available. For a WebSphere infrastructure that is composed of several components, such as

load balancers, HTTP servers, application servers, database servers, and so on, availability is determined by the weakest component.

For most of the environment's components, several degrees of HA implementation exist. The cost of the infrastructure is directly linked to the level of availability. Evaluate the business loss of the infrastructure downtime, and ensure that the business case justifies the costs. Moving a system availability from 99.9% to 99.99% can be expensive. It can also be true that the system is used only during regular business hours on regular working days. This use implies that an availability of 99.9% is more than adequate to meet the operational window.

For additional information about this topic, go to the following website:

http://www.ibm.com/developerworks/websphere/techjournal/0312_polozoff/polozoff.html#sec1

Because it is likely that the complete environment is made up of multiple systems, the goal is to make the whole system as available as possible by minimizing the number of single points of failure (SPOFs) throughout the system by adding redundancy. Redundancy can be added at different layer, such as hardware, process, and data.

15.3.2 WebSphere Application Server high availability and failover

This section provides information about the WebSphere Application Server HA features. It can help you to understand how the HA features work and can assist you in planning and configuring for HA.

Figure 15-15 represents a typical WebSphere Application Server topology, where a request travels through the load balancer to the HTTP server to the web container and finally to the EJB container.

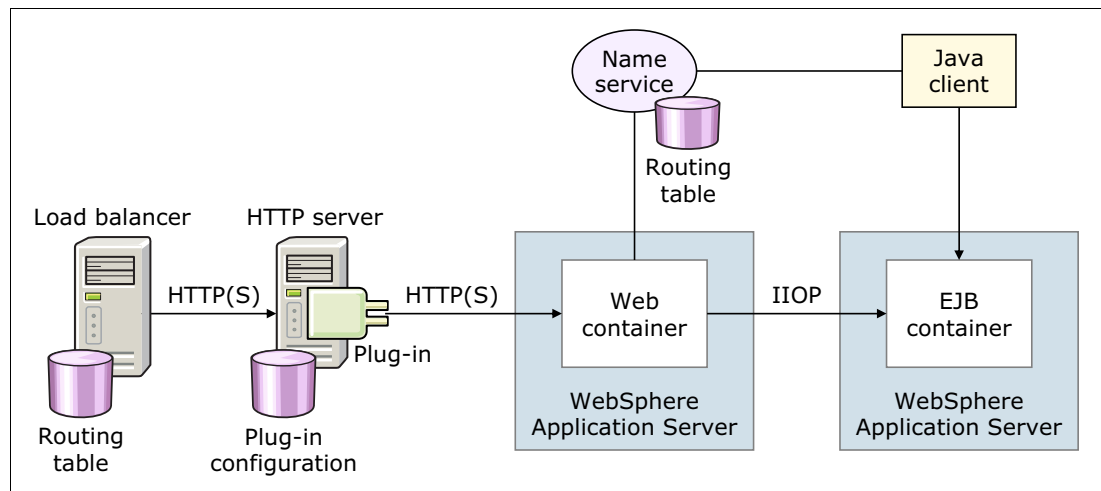


Figure 15-15 WebSphere topology and incoming request

The sections that follow provide information about HA and failover options for components and services that are involved in the processing of the request.

Load balancing

You can configure the Edge Components Load Balancer to run as an active-passive pair. The active instance is the single point of distribution for HTTP requests. If it fails, the passive instance routes the requests.

If you use a third-party load balancer, consult with the vendor for load HA features.

Web server high availability and failover

When a web server fails, the Edge Components Load Balancer detects the failure and routes the requests around the failing web server. If there is a session affinity to a server, it still holds, because all web servers have a copy of the plug-in that can route the requests to servers with affinities.

Web container high availability and failover

A typical WebSphere environment can include several web server instances and several WebSphere Application Server instances. Each HTTP server is configured to run the WebSphere web server plug-in. Each request coming into the web server is passed through the plug-in, which uses its configuration information to determine if the request is routed to WebSphere and, if so, to which application server (that is, to which web container) the request is routed. The communication between the plug-in and the application servers can be either HTTP or HTTPS. The plug-in, which runs in-process with the web server itself, determines to which web container the request is passed. The plug-in also distributes requests around cluster members that are not available.

WebSphere Application Server provides the following mechanisms for web container WLM and failover:

- ▶ Application server clustering creates server process redundancy for failover support. All application servers in a cluster host the same application or applications.
- ▶ The workload management routing technique and the failure detection mechanism is built into the WebSphere web server plug-in. The plug-in can detect a failed web container and mark it down. Then the plug-in continues to route the request to available server processes.
- ▶ Session management and a failover mechanism provide HTTP session data for redundant server processes.

Satisfactory failover support for web clients can be achieved only by using all three mechanisms, as illustrated in Figure 15-16.

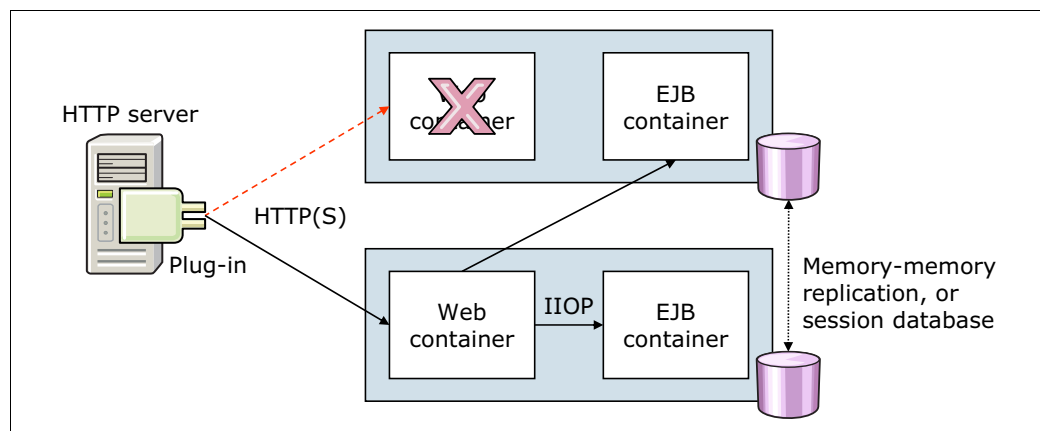


Figure 15-16 Web container high availability

EJB container clustering and failover

Many JEE applications rely on Enterprise JavaBeans (EJB) to implement key business logic. Therefore, providing a resilient and highly available EJB runtime system is a critical task for any EJB container provider. WebSphere Application Server V8.5 satisfies this requirement for

EJB applications by providing an advanced HA solution, which guarantees that EJB requests can be serviced continuously even during various types of failures.

EJB clients can be servlets, JSP pages, a JEE client, stand-alone Java applications, or other EJB beans. When an EJB client makes calls from within the WebSphere or client container or outside of a container, the request is handled by the EJB container in one of the cluster members. If that cluster member fails, the client request is redirected automatically to another available server, as illustrated in Figure 15-17.

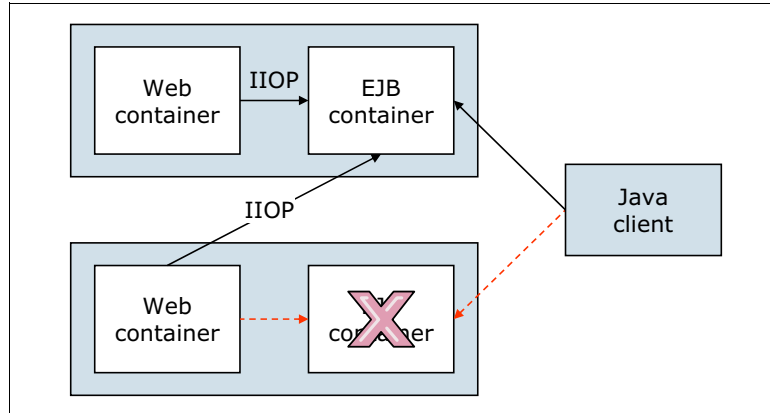


Figure 15-17 EJB container High Availability

In IBM WebSphere Application Server Network Deployment V8.5, EJB HA is achieved using a combination of the following WebSphere services:

- ▶ The HA manager
- ▶ The EJB server cluster
- ▶ EJB WLM

The mechanisms for routing workload-managed EJB requests to multiple cluster members are handled on the client side of the application. In WebSphere Application Server V8.5, this functionality is supplied by a workload management plug-in to the client ORB and the routing table in the Location Service Daemon (LSD) that is hosted in the node agent. The WLM failover support for EJB beans maintains the routing table and modifies the client ORB to redirect traffic in case of a server failure.

EJB failover depends on whether the type of EJB can be workload-managed by the container. Table 15-3 summarizes failover support for different EJB types.

Table 15-3 Summary of EJB types and failover support

| EJB type | Component | Failover capable |
|-------------------------------|---|-------------------|
| Entity bean (Option A) | Home CMP bean instance BMP bean instance | Yes No No |
| Entity bean (Options B and C) | Home CMP bean instance BMP bean instance | Yes Yes Yes |
| Session Bean | Home Stateless bean instance Stateful bean instance | Yes Yes Yes |

EJB WLM failover scenarios

EJB WLM failover can happen in the following scenarios:

- ▶ A failure occurs during EJB processing
This failure triggers an exception. Normally, WebSphere WLM catches the exception and resends the failed request to another application server in the cluster. This method is the normal failover function of WebSphere EJB WLM. If, however, WLM cannot determine whether the transaction completed, it sends an exception to the application.
- ▶ A failure occurs during WLM processing
If the WLM mechanism cannot handle the request, it sends an exception to the application.

Exceptions that cause WLM to fail over

The EJB WLM catches most of the exceptions that can occur during execution. WLM handles these exceptions and decides whether the original request is redirected to another available cluster member. The EJB WLM includes the following exceptions:

- ▶ `org.omg.CORBA.COMM_FAILURE`
- ▶ `org.omg.CORBA.NO_RESPONSE`

These exceptions have a `COMPLETION_STATUS` of `NO`, `YES`, or `MAYBE`, and these values actually determine whether the request fails over to another available member as follows:

- ▶ With a `COMPLETION_STATUS` of `COMPLETED_NO`, automatic failover occurs because the request was not completed. The request is then rerouted to an available cluster member.
- ▶ With a `COMPLETION_STATUS` of `COMPLETED_YES`, there is no need to fail over because the transaction completed successfully. Communication errors might have occurred during the marshalling of the answer.
- ▶ With a `COMPLETION_STATUS` of `COMPLETED_MAYBE`, WLM cannot verify whether the transaction was completed and, thus, cannot redirect the request. WLM, according to the programming model, sends this exception to the client application. It is the applications' responsibility to handle this exception and to decide whether to retry the request. This status does not mean that something is broken. The application has to check whether the transaction was successful and, depending on the result, then issues the request.

Exceptions during failover

During the failover process, WLM might face problems and then try to redirect the request again or issue a CORBA exception. If a CORBA exception issues, something unusual occurred while WLM was trying to redirect the original request to another server. There are two reasons for such an exception to occur. Both events are contained in an `org.omg.CORBA.TRANSIENT` exception, with different minors:

- ▶ Communication problem with one of the other servers
The following exception means that an error occurred during the communication and that the client must retry the request:
`TRANSIENT_SIGNAL_RETRY (minor=1229066306)`
- ▶ No reachable server
The following exception means that WLM did not find a cluster member that can answer the request:
`NO_IMPLEMENT_NO_USEABLE_TARGET (minor=1229066304)`

This exception is the worst case scenario. The application cannot retry the request because it will fail again. This failure then leads to a message to the user, for example, asking the user to contact the system administrator.

Session persistence

For information about session persistence, refer to 28.3.2, “Persistent sessions management” on page 973.

Default messaging provider availability

The service integration bus (SIBus) is used to provide HA to the messaging system process. WebSphere Application Server provides the ability to configure the following policies to achieve message engine HA based on the cluster utilization:

- ▶ High availability

One message engine is created in the cluster and can fail over to any other server in the cluster. The message engine does not fail back to the previous server if this server becomes available again.

- ▶ Scalability with high availability

One message engine is created for each application server on the cluster. Each message engine can fail over to any other server in the cluster. All the messages set for high reliability that were being processed or queued continue to be processed when the message engine is available on another server. Each message engine can fail back to the previous server when this server is available again.

To accomplish the failover seamlessly, the queue information and message data must be stored in a shared location that is reachable by all the members of the cluster, either using an external database or a shared disk environment.

For additional information about the availability policy, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/cjt1005_.html

Using embedded Derby: If you use embedded Derby as a messaging data store, concurrent access can be a concern. The embedded Derby does not support multiple servers running the Derby engine. Thus, there is no ability to have multiple servers communicating with the same shared file system.

Resources high availability

With WebSphere Application Server V8.5, you can configure failover resources for a data source and connection factory. Resource workload routing improves the availability of the applications. A data source and connection factory can fail over when a default occurs and can fail back when the situation comes back to normal. Only one resource can be used at a time, and the alternate resource is available only when the primary resource fails. To use the alternate resource, you have to create alternate resources for the data source and connection factory. These resources must be identical to the primary resources and must be compatible with applications. Then to finish, add custom properties to configure the availability behavior.

For further details about this topic, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.base.doc/ae/cdat_dsfailover.html

Deployment manager high availability and failover

Although it is not required to have deployment manager running at all times, you might require highly available administrative capability, especially in environments that have a significant number of new application deployments or updates and server monitoring.

Multiple instances of deployment manager removes the single point of failure (SPOF) for cell administration, assuring the attainability of the administrative console, wsadmin, and scripting features to manage your environment.

WebSphere Application Server provides a mechanism for cloning your existing deployment manager, thus achieving high availability, by employing redundant deployment managers with a hot-standby model and the use of a shared file system.

In this paradigm, one of the deployment managers is elected as primary. As primary it is considered an active deployment manager hosting the cell-wide endpoints for the administrative functions. Other deployment managers are considered backups and are kept in standby mode and are available to take over the active role in case of failure or termination of the primary manager.

A highly-available deployment manager component runs in each deployment manager to control which deployment manager is elected as the active one.

The Figure 15-18 illustrates a common topology for the highly available deployment manager.

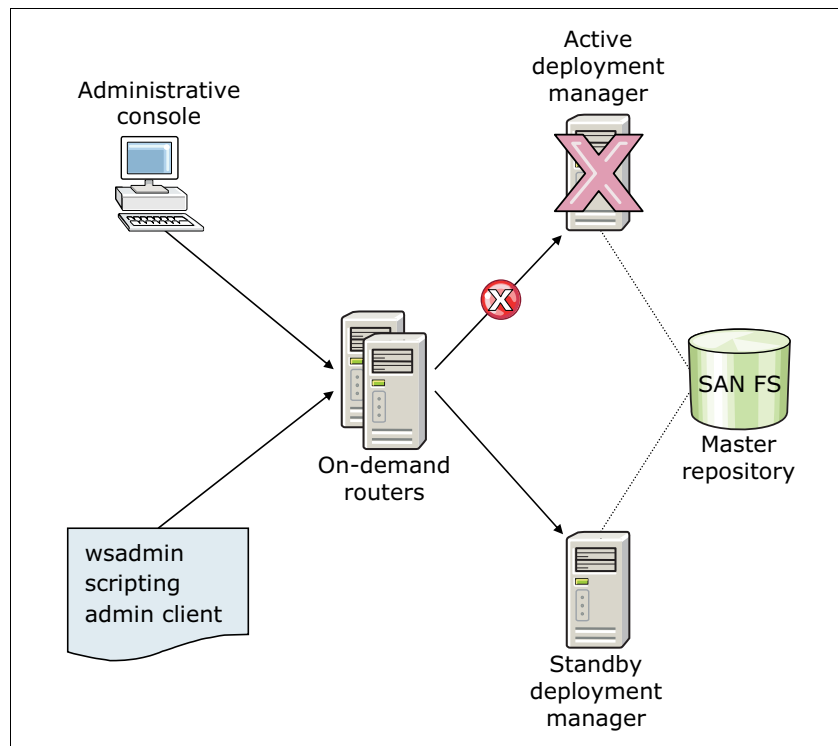


Figure 15-18 High availability deployment manager topology

Note that this topology requires the use of an ODR server. It is not supported when the web server is enabled for Intelligent Management.

For more information, refer to the section “10.10 Highly available deployment manager” in the *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022, at the following website:

15.3.3 How high availability features work

This section provides information about the WebSphere Application Server components and concepts that facilitate HA.

High-availability manager

WebSphere Application Server uses an HA manager to eliminate SPOFs. The HA manager runs key services on available application servers rather than on a dedicated server (such as the deployment manager). It continually polls the core group members to verify that they are active and healthy. The HA manager service runs by default in each server, as shown in Figure 15-19.

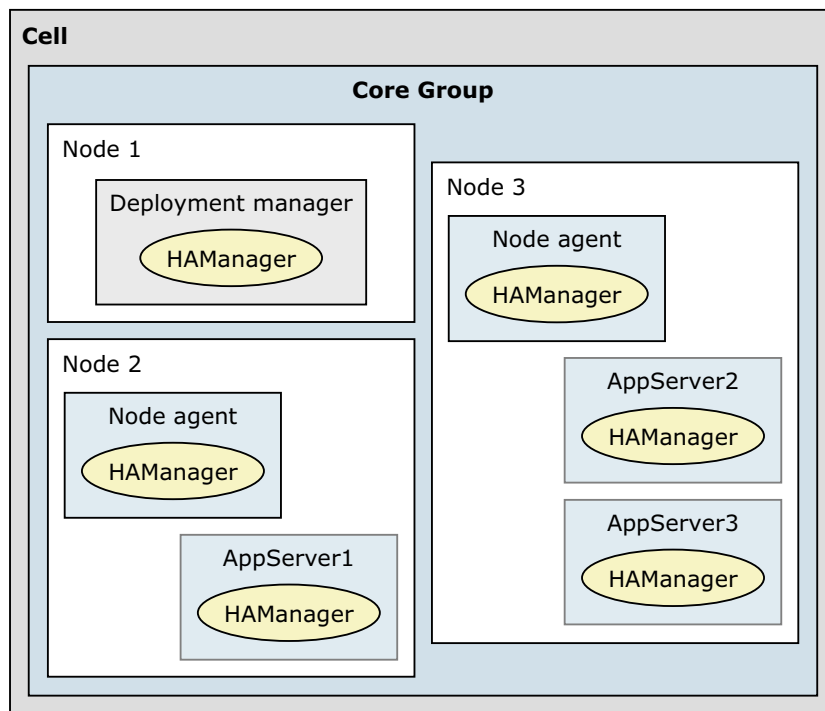


Figure 15-19 Conceptual diagram of a core group

For certain functions (such as transaction peer recovery), the HA manager takes advantage of fault tolerant storage technologies, such as Network Attached Storage (NAS), which lowers the cost and complexity of HA configurations. The HA manager also provides peer-to-peer failover for critical services by maintaining a backup for these services. WebSphere Application Server supports other HA solutions such as Power HA, IBM Parallel Sysplex®, and so on.

An HA manager monitors the application server environment continually. If an application server component fails, the HA manager takes over the in-flight and in-doubt work for the failed server. This process introduces some processing impact, but improves application server availability.

An HA manager focuses on recovery support and scalability in the following areas:

- ▶ Embedded messaging
- ▶ Transaction managers

- ▶ Workload management controllers
- ▶ Application servers
- ▶ WebSphere partitioning facility instances
- ▶ On-demand routing
- ▶ Memory-to-memory replication through Data Replication Service (DRS)
- ▶ Resource adapter management

To provide this focused failover service, the HA manager supervises the JVMs of the application servers that are core group members. The HA manager uses one of the following methods to detect failures:

- ▶ An application server is marked as failed if the socket fails.

This method uses the KEEP_ALIVE function of TCP/IP and is tolerant of poor performing application servers, which might happen if the application server is overloaded, swapping, or thrashing. This method is best for determining a JVM failure if you are using multicast emulation and are running enough JVMs on a single application server to push the application server into extreme processor starvation or memory starvation.
- ▶ A JVM is marked as failed if it stops sending heartbeats for a specified time interval.

This method is referred to as *active failure detection*. When it is used, a JVM sends out one heartbeat, or pulse, at a specific interval. If the JVM does not respond to heartbeats within a defined time frame, it is considered down.

With WebSphere Application Server, you can configure an alternative protocol provider to monitor and manage communication between core group members. In general, alternate protocol providers, such as the z/OS Cross-System Coupling Facility (XCF)-based provider, uses less system resources than the default Discovery Protocol and Failure Detection Protocol, especially during times when the core group members are idle.

In either case, if a JVM fails, the application server on which it is running is separated from the core group, and any services running on that application server are failed over to the surviving core group members.

A JVM can be a node agent, an application server, or a deployment manager. If a JVM fails, any singletons running in that JVM are activated on a peer JVM after the failure is detected. This peer JVM is already running and eliminates the normal startup time, which potentially can be minutes.

This method of detecting failovers actually is a key difference to using operating system-based HA. With this method, the HA manager usually recovers in seconds, and operating system-based solutions take minutes.

When an application server fails, the HA manager assigns the failing application servers work to another eligible application server. Using shared storage for common logging facilities (such as the transaction logs) allows the HA manager to recover in-doubt and in-flight work if a component fails.

Note: Additional resource: The following website provides a testing routine that you can use to determine whether your shared file system is suitable for use with the HA manager:

http://www-01.ibm.com/support/docview.wss?rs=180&context=SSEQTP&q1=transaction+log+failover&uid=swg24010222&loc=en_US&cs=utf-8&lang=en

Coregroups

A *core group* is an HA domain that consists of a set of processes in the same cell that can establish HA relationships directly. Highly-available components can fail over only to another

process in the same core group, and replication can occur only between members of the same core group.

A cell must contain at least one core group, although multiple core groups are supported. Each core group contains a core group coordinator to manage its HA relationships and a set of HA policies that are used to manage the highly-available components within that core group.

WebSphere Application Server provides one standard core group, the DefaultCoreGroup, that is created during installation. New server instances are added to the default core group as they are created.

In most cases, one core group is sufficient for establishing an HA environment. However, certain topologies require the use of multiple core groups. A basic rule is that all members of a core group require full IP visibility. Therefore, you have to create multiple core groups if you spread the application servers of your cell across different firewall zones.

If you have more than 50 servers in a cell: A large number of application servers in a cell increases the processing impact of core group services and server startup times. Consider creating additional core groups when you have more than 50 servers in a cell.

If you are using a DMZ secure proxy server with dynamic routing, the routing information is exchanged using core groups. In this case, you need to create a tunnel access point group to establish a core group bridge tunnel between the core groups that are running on both sides of the firewall.

The core group contains a bridge service that supports cluster services that span multiple core groups. Core groups are connected by access point groups. A core group access point defines a set of bridge interfaces that resolve IP addresses and ports. It is through this set of bridge interfaces that the core group bridge provides access to a core group.

When moving core group members to new core groups, remember the following information:

- ▶ Each server process within a cell can only be a member of one core group.
- ▶ If a cluster is defined for the cell, all cluster members must belong to the same core group.

Network communication between all members of a core group is essential. The network environment must consist of a fast local area network with full IP visibility and bidirectional communication between all core group members. IP visibility means that each member is entirely receptive to the communications of any other core group member.

High-availability groups

HA groups are part of the HA manager framework. An HA group provides the mechanism for building a highly available component and enables the component to run in one of several different processes. An HA group cannot extend beyond the boundaries of a core group.

An HA group is associated with a specific component. The members of the group are the set of processes where it is possible to run that component. A product administrator cannot directly configure or define an HA group and its associated set of members. Instead, HA groups are created dynamically at the request of the components that need to provide a highly available function.

HA groups are dynamically created components of a core group. A core group contains one or more HA groups. However, members of an HA group can also be members of other HA groups, if all of these HA groups are defined within the same core group.

Every HA group has a policy associated with it. This policy is used to determine which members of an HA group are active at a given time. The policies that HA groups use are stored as part of the core group configuration. The same policy can be used by several HA groups, but all of the HA groups to which it applies must be part of the same core group.

Any WebSphere Application Server highly available component can create an HA group for its own usage. The component code must specify the attributes that are used to create the name of the HA group for that component. For example, establishing an HA group for the transaction manager is achieved by completing the following process:

1. The code included in the transaction manager component code specifies the attribute `type=WAS_TRANSACTION`s as part of the name of the HA group that is associated with this component.
2. The HA manager function includes the default policy Clustered TM Policy that includes `type=WAS_TRANSACTION`s as part of its match criteria.
3. Whenever transaction manager code joins an HA group, the HA manager matches the match criteria of the Clustered TM Policy to the HA group member name. In this example, the name-value pair `type=WAS_TRANSACTION`s that is included in the HA group name is matched to the same string in the policy match criteria for the Clustered TM Policy. This match associates the Clustered TM Policy with the HA group that was created by the transaction manager component.

After a policy is established for an HA group, you can change the policy attributes, such as quorum, failback, and preferred servers. However, you cannot change the policy type. If you need to change the policy type, create a new policy and then use the match criteria to associate that new policy with the appropriate group.

Match criteria: If you want to use the same match criteria, delete the old policy before defining the new policy. You cannot use the same match criteria for two different policies.

Dynamic high availability

WebSphere's Autonomic request flow manager (ARFM) classifies and prioritizes requests to application servers based on demand and policies. The dynamic workload manager (DWLM) then distributes the requests among the nodes to balance the work.

DWLM sets the load balancing weights for application servers dynamically to stay current with the business goals. The weights are then used by the router in the ODR to distribute the workload accordingly. This autonomic manager continuously monitors the response time and resource utilization of each server and uses feedback control techniques that change the dispatching weights to achieve balance across the clusters and nodes.

DWLM can also dynamically update application status, using the application placement controller, to make modifications to a running application infrastructure. The application placement controller receives information from the ARFM and together with the performance data, service policies, and service goals, it computes the optimal allocation of available resources to running applications.

DWLM is enabled by default. It can be enabled or disabled through the administrative console under:

- ▶ **Servers** → **Dynamic Clusters** → **cluster_name** → **Dynamic workload management (DWLM)**
- ▶ **Servers** → **Clusters** → **cluster_name** → **Dynamic workload management (DWLM)**

For more information, refer to the section “10.10 Highly available deployment manager” in the *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022, at the following website:

<http://www.redbooks.ibm.com/abstracts/sg248022.html?Open>

15.4 ODR server considerations

With WebSphere Application Server V8.5.5, the ODR function in an Intelligent Management topology can be implemented in one of two ways: by using an ODR server or by enabling the web server for Intelligent Management. This section describes additional considerations for administrators that use the ODR server option.

Creating an ODR server dynamic cluster

ODR servers support dynamic clusters of several server types. But the ODR servers can also participate in a dynamic cluster.

To create a cluster of ODR servers from the administrative console, follow these steps:

1. Click **Servers** → **Clusters** → **Dynamic clusters** and then click **New**.
2. Set the type of the new dynamic cluster being created to **On demand router**, and click **Next**.
3. Configure the new cluster with automatic membership by selecting **Automatically define cluster members with rules**:
 - a. Enter a cluster name.
 - b. Enable the options best suited for your environment:
 - Prefer local enabled: Enabled by default, indicating that Enterprise JavaBeans (EJB) requests will be routed to the same host as the host of the requesting EJB client.
 - c. Click **Next**.
4. Define dynamic cluster members:
 - a. By default, the value of the Membership policy field will be set to `node_nodegroup = 'DefaultNodeGroup'`, which means that all of the nodes belonging to this node group will be members of the new dynamic cluster. To modify the membership policy, click the **Subexpression builder** link under Edit rule.
 - b. From the membership policy Subexpression builder window, modify the initial policy by setting the following options:
 - Define the appropriate value for Logical operator as either AND or OR.
 - Set the expression's operand value, which can be Nodegroup, Node name, Node host name, and Node property.
 - Choose the desired expression Operator.
 - Enter the appropriate information in the value field, matching the chosen operand value for the expression's operation.
 - Click **Generate subexpression**.
 - Click **Append**.
 - c. Click **Next**.

5. Create the cluster members based on one of the following options, as shown in Figure 15-20:
 - a. Use an application server template:
 - http_sip_odr_server
 - odr
 - sip_odr_server
 - Using an existing ODR as a template
 - b. Specify a target core group for the new cluster members.
 - c. Click **Next**.

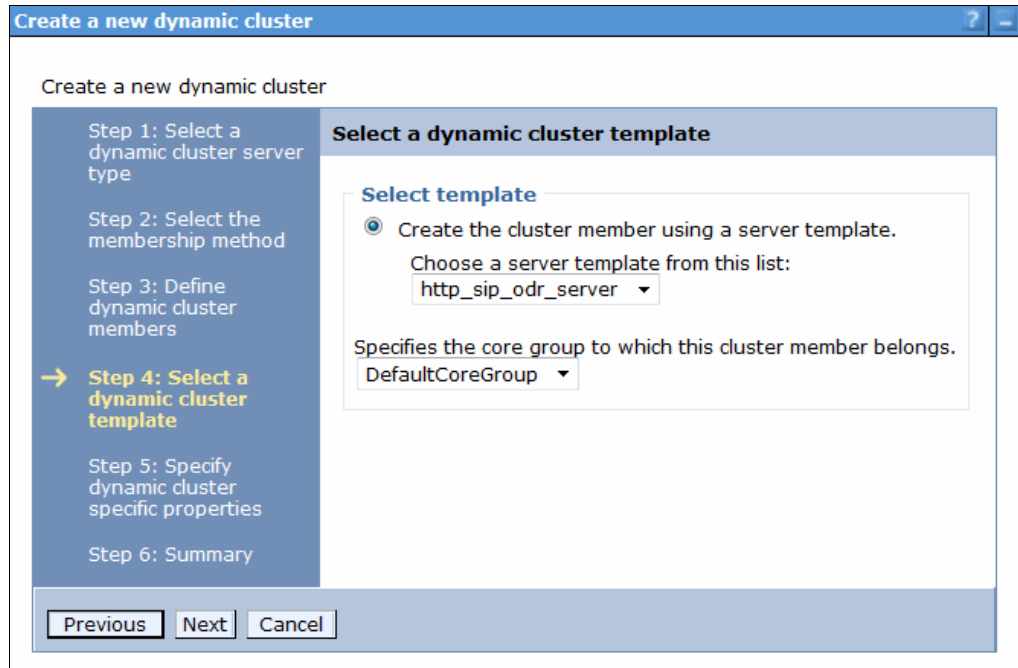


Figure 15-20 Specifying a target core group

6. Complete the ODR dynamic cluster configuration by specifying its runtime and creation-specific properties:
 - The minimum number of running instances for the dynamic cluster.
 - The maximum number of cluster instances that can be started at any given time.
 - The vertical stacking or number of cluster instances that are created per node.
 - The isolation mode, to specify whether you want the dynamic cluster to run on the same nodes as other instances of different dynamic clusters or whether you want the dynamic cluster to be the only instances running on a node. Options for isolation mode are:
 - No isolation: Target nodes can be shared between different dynamic clusters. This option is the default.
 - Strict isolation: Target nodes are dedicated to a specific dynamic cluster, running only application server instances of it.

- Associate with an isolation group: Requires the creation of an isolation group, which determines which dynamic clusters can share the same target nodes. If only one cluster is added into an isolation group, the behavior will be the same as the behavior of the strict isolation.

d. Click **Next**.

15.4.1 Web server plug-in when using the ODR server

If you are using an ODR server and if your web servers are running on managed nodes (a node agent is present on the web server system), you must disable the automatic generation and propagation of the web server plug-in. This ensures that you use the ODR server's plug-in file rather than the web server plug-in file. The reason is that the web servers automatically update their `plugin-cfg.xml` file when plug-in-related configurations occur in the environment. This has to be disabled because the `plugin-cfg.xml` file that is generated on the web servers directs requests directly to the application server environment rather than to the ODRs.

To disable automatic generation and propagation of the Web server plug-in, follow these steps:

1. Go to **Servers** → **Web servers** → *Web_server_name* → **Plug-in properties**.
2. Clear the **Automatically generate the plug-in configuration file** check box.
3. Clear **Automatically propagate plug-in configuration file**.
4. Click **OK**, and save the configuration.

This process does not apply to topologies that have enabled the web server for Intelligent Management.

15.4.2 Configuring the ODR proxy plug-in configuration policy

The web server plug-in configuration file is automatically generated by the ODR when a change occurs. However, the file is not automatically propagated to the web servers. You must take one of the following actions:

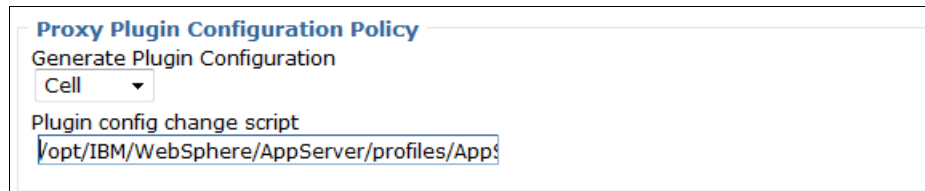
- ▶ Manually transfer the files to the appropriate locations on the web servers.
- ▶ Create a script that copies the configuration file to the appropriate locations on the web servers, and configure the ODR or ODR cluster to execute this script.

The ODR runs the configuration script every time it generates a new `plugin-cfg.xml`, therefore, allowing an automatic propagation of the proxy plug-in configuration.

The configuration for the proxy plug-in generation of the ODRs can be done at different scopes, depending on the ODR's topology. The configuration for an ODR dynamic cluster is outlined in the following steps:

1. From the administrative console, navigate to **Servers** → **Clusters** → **Dynamic Clusters** → *ODR_Cluster_Name* → **Server template**.
2. Expand the option **On Demand Router properties**.
3. Click **On Demand Router settings**.
4. Locate the Proxy Plugin Configuration Policy section, and set the appropriate value for Generate Plugin Configuration, as shown in Figure 15-21 on page 552. The following values are available for selection:
 - All: The generated `plugin-cfg.xml` file includes all ODRs in the environment.

- Cell: The plug-in file includes all ODRs in the same cell.
 - Node: The plug-in file includes all ODRs on the same node.
 - Server: The plug-in file includes only a single ODR (itself).
 - None: Disables generation of the `plugin-cfg.xml` file.
5. In the Plugin config change script field, enter the full path of the custom plug-in propagation script (see Figure 15-21).
 6. Click **Ok**.
 7. Save the configuration to the master repository.



Proxy Plugin Configuration Policy

Generate Plugin Configuration

Cell ▼

Plugin config change script

/opt/IBM/WebSphere/AppServer/profiles/AppS

Figure 15-21 Configuration for the ODR plug-in generation and propagation



Monitoring distributed systems

Having the ability to measure and monitor system interactions helps information technology (IT) to provide business continuity. Monitoring capabilities plays a key role in successfully managing enterprise systems. In WebSphere Application Server, there are a number of tools that can contribute to an organization's monitoring strategy and provide insights into the performance of the application server.

In this chapter, we provide an introduction to these toolsets.

We cover the following topics:

- ▶ Overview
- ▶ Enabling monitoring infrastructures
- ▶ Viewing the monitoring data
- ▶ Monitoring examples
- ▶ Monitoring operations
- ▶ IBM Tivoli Composite Application Manager for WebSphere Application Server
- ▶ Additional resources for monitoring

16.1 Overview

IT environments are complex, involving many different servers working together to deliver the electronic functions of business. In a single user interaction, it is typical that information can be retrieved from many systems. Consider the simple, distributed WebSphere Application Server environment in Figure 16-1.

The stars in Figure 16-1 show that even a simple web application request can pass through a whole series of dependent servers to successfully complete a request. JEE is a component-based architecture, requiring a request to interact and use 'n' number of these components to complete. Monitoring system components and their performance can become complex, yet are critical to understanding the overall performance of an application.

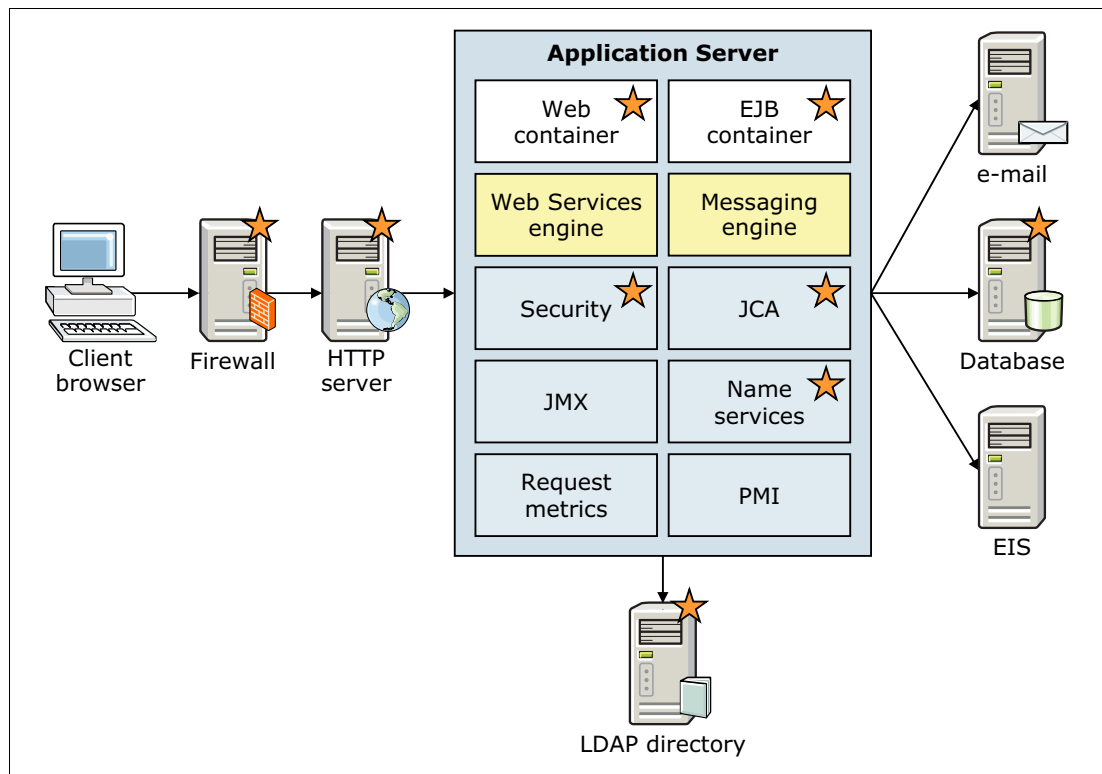


Figure 16-1 Simple web system topology

Monitoring the systems contributes to overall systems management by:

- ▶ Establishing an understanding of the performance baseline and of what runtime behaviors constitute “normal” operations
- ▶ Measuring performance and identifying poorly performing systems and components
- ▶ Identifying service failures and assisting in root cause identification

WebSphere Application Server monitoring tools rely primarily on information gathered from two core data infrastructures:

- ▶ Performance Monitoring Infrastructure (PMI), which is a collection of statistical agents scattered through out the application server that gather statistical data on the performance of the application server components. For more information about this topic, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/tprf_monitoringhealth.html

- ▶ Request metrics, which are primarily a set of timing agents that track a request as it navigates the components of the application server. A key differentiation of request metrics is that they are measured at the request level. The focus of a request metric is to record the time spent by individual requests in different components of the application and at the end of the request provide a record of where time was spent in the request. For more information about this topic, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/tprf_requestmetrics.html

16.2 Enabling monitoring infrastructures

This section shows you how to enable the PMI monitoring infrastructure and the request metrics that provide the performance data.

16.2.1 PMI defaults and monitoring settings

The enabling of PMI data is managed on a server-by-server basis. In the administrative console, complete the following steps:

1. Click **Monitoring and Tuning** → **Performance Monitoring Infrastructure (PMI)**.
2. Select the server for which you want to manage the PMI controls. Figure 16-2 on page 556 shows the PMI configuration window for the server.

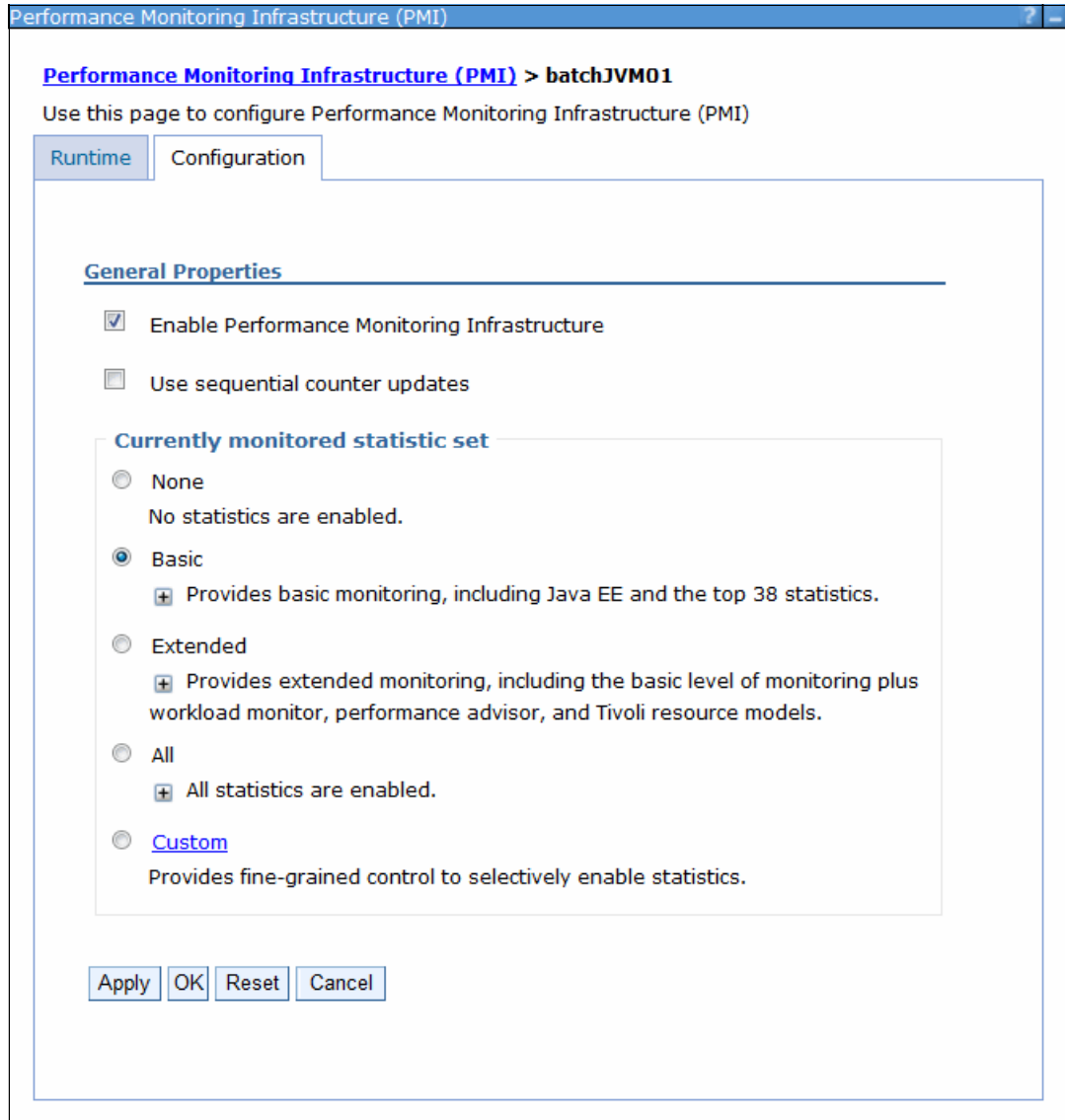


Figure 16-2 Default PMI settings

On this window, note that:

- PMI is enabled by default.
- The default statistical set is the Basic set.

The PMI data can be changed at run time using the settings on the Runtime tab.

Disabling and enabling: Disabling and enabling of PMI data requires a server restart.

The enabling and disabling of PMI is not available on the Runtime tab. However, the monitoring level can be set to None in a server with PMI enabled using the Runtime tab.

Understanding the sets of PMI statistics

The PMI statistic sets represent a group of individual statistical agents. The types of statistics that PMI can collect are classified. Information about these classes is in the WebSphere Information Center on the “PMI data classification” page at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/rprf_dataclass.html

Every action executed in an environment has a cost. Monitoring is no different, and for PMI, the cost of monitoring is impacted primarily by two factors:

- ▶ The amount of data that is monitored.
- ▶ The impact of individual performance metrics. Not all metrics have the same collection cost.

With PMI, there are multiple sets of statistics that can be enabled, as shown in Figure 16-2 on page 556. These sets of statistics are:

- ▶ None
- ▶ Basic
- ▶ Extended
- ▶ All
- ▶ Custom

None and All are self explanatory, so here we take a closer look at the options provided by Basic, Extended, and Custom.

Basic statistic set

The Basic statistic set is the default setting. The basic setting is configured with the intention of providing an overall understanding of application server health, including statistics as outlined in the JEE specification and other common performance hot spots and key monitoring points for JEE applications. Later, we discuss how to determine the impact and level of a statistic (see “Getting more information about statistics sets” on page 561).

Figure 16-3 on page 558 shows the list of PMI counters that are active for the basic PMI data level. For details about each counter, refer to “Getting more information about statistics sets” on page 561.

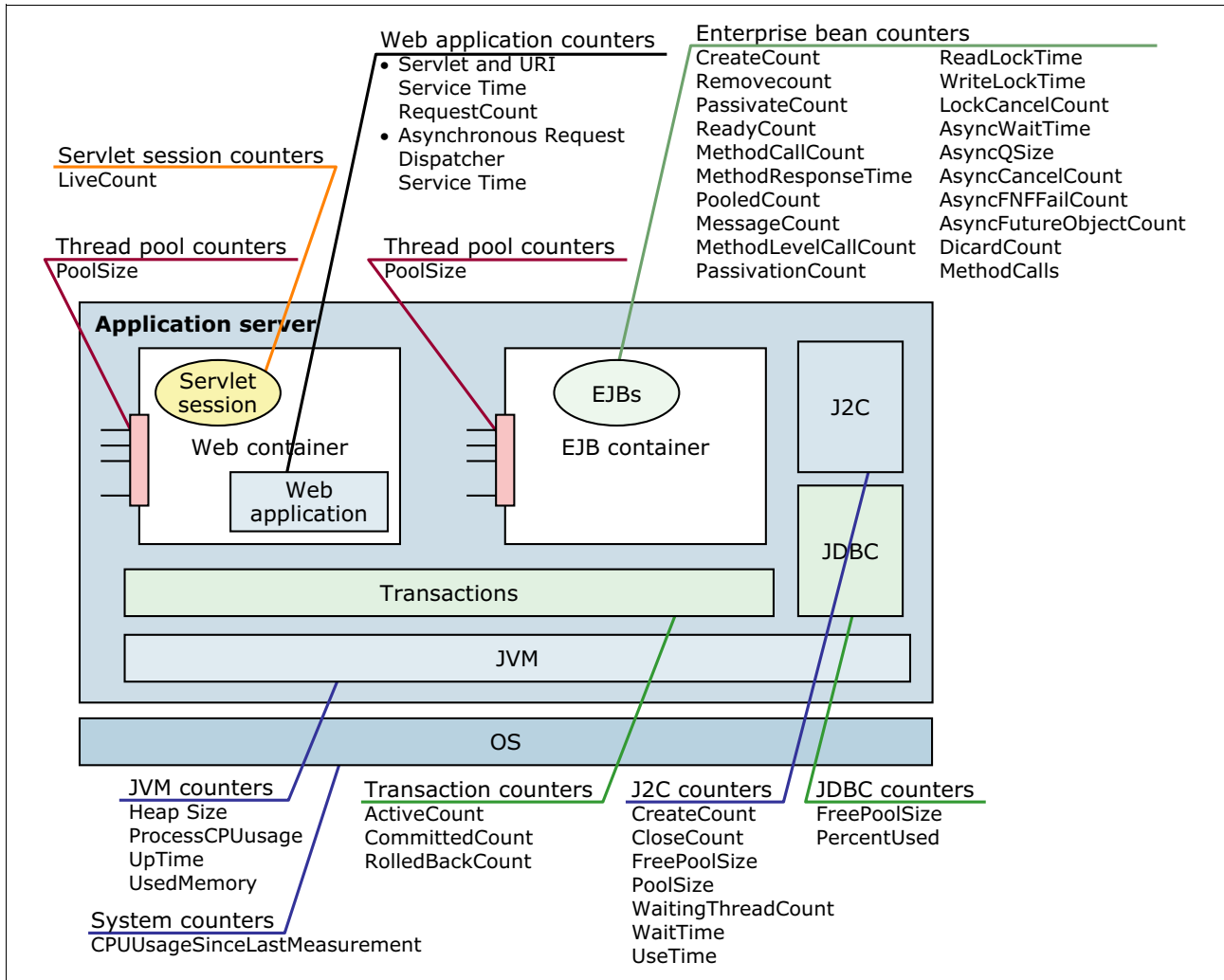


Figure 16-3 PMI basic counters

Extended statistic set

The extended PMI data set has the basic set and some additional statistics with a particular emphasis on statistics that look at the load on the server and the servers response to the load being applied. The statistical agents in the extended set might or might not apply to a JEE application depending on the individual application architecture and environment configurations.

Figure 16-4 on page 559 shows the extended metrics that are in addition to those of the basic configuration.

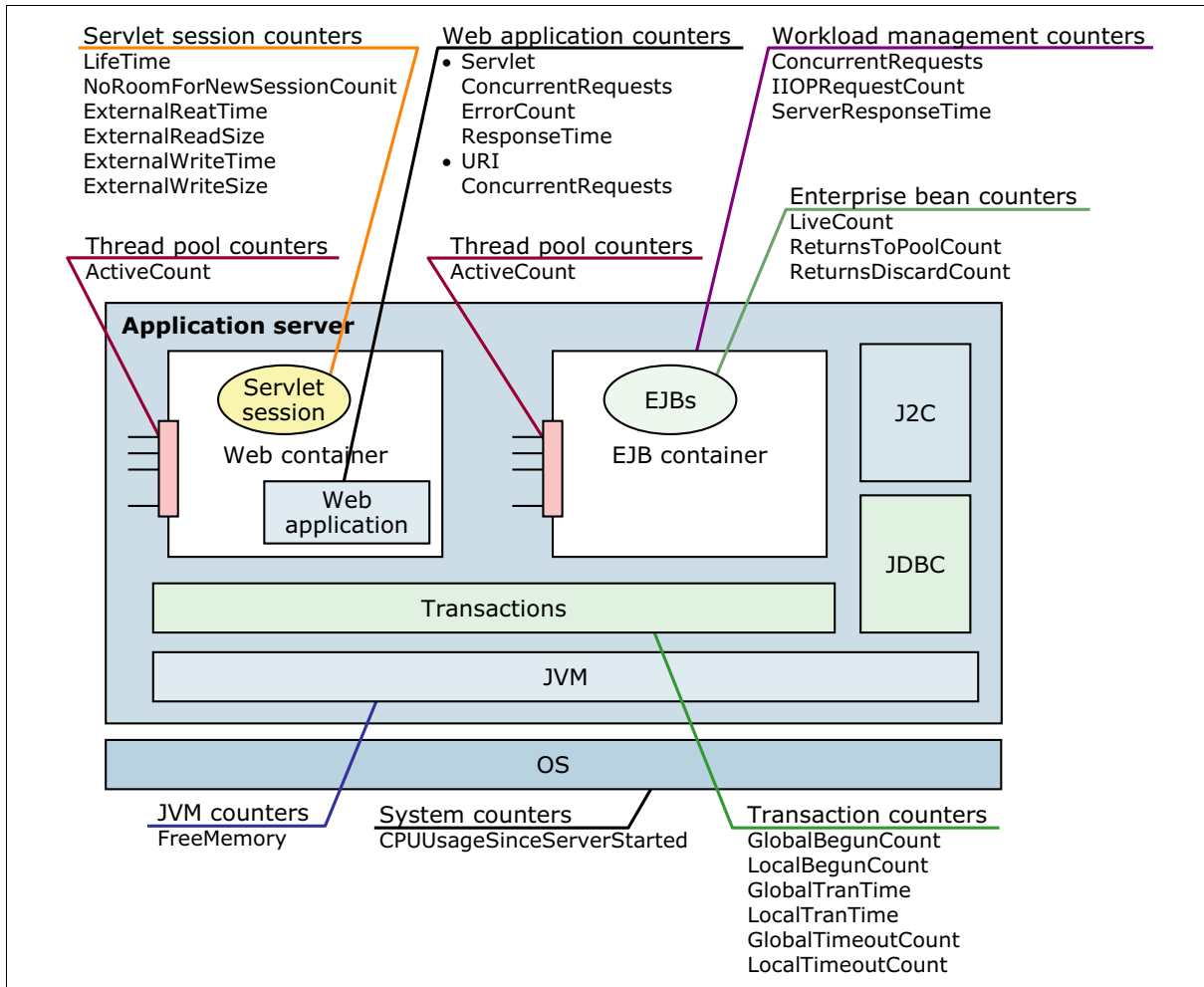


Figure 16-4 PMI extended counters

Custom statistic set

The Custom PMI data collection set allows the administrator to choose the counters that are most appropriate for the application(s) that are deployed on the server. Each counter is individually activated. This is the most powerful configuration but requires that the administrator spend some time reviewing the available statistical counters and also that the administrator understands the type of counter that is useful for the applications. For example, consider the counters activated for ServletSession if the extended data set is selected. The counters are:

- ▶ LiveCount
- ▶ LifeTime
- ▶ NoRoomForNewSessionCount
- ▶ ExternalReadTime
- ▶ ExternalReadSize
- ▶ ExternalWriteTime
- ▶ ExternalWriteSize

The NoRoomForNewSessionCount counter only applies if the Allow overflow from the web container session management was changed from its default value of true. This attribute is shown in Figure 16-5 on page 560.

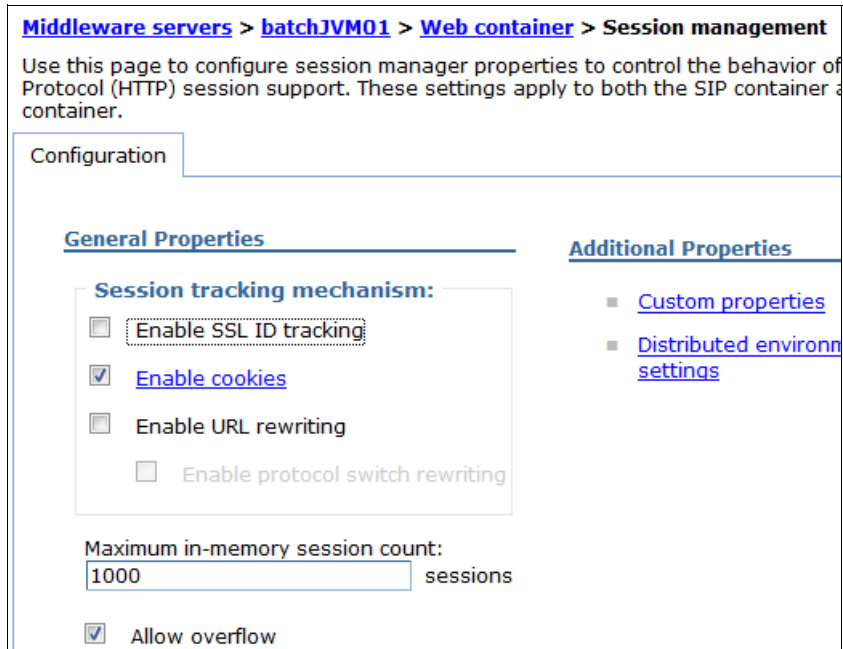


Figure 16-5 Allow overflow default is true

Similarly, the counters related to external session management only apply if session persistence is configured. Hence, the activation of the extended session information does little for an application where the overflow option is not modified and session persistence is not configured.

With a custom metric approach, the administrator can choose to simply add LiveCount and LifeTime counters and perhaps choosing other metrics of interest, such as the TimeoutInvalidationCount, to measure how many sessions are being timed out rather than logged off.

Tip for using custom PMI settings: The counters used for the basic set can be customized to form a baseline for the custom counter activation. Supplement them with additional counters that are relevant for the application types that are being deployed.

Impact of PMI

The actual impact of each statistic level varies depending on the particular applications on the server and load that is being executed by the server. In the WebSphere Information Center, each counter has a documented qualitative impact level to indicate the type of impact it will incur (see “Getting more information about statistics sets” on page 561). This is not intended to prevent administrators from using counters with high impact. It is important to remember that the impact is a relative measurement, and the administrator must balance the need for the data versus the impact incurred to enable a particular counter temporarily or for the long term.

The approximate impact of the PMI statistic sets are:

- ▶ Basic impact up to 2%.
- ▶ Extended impact up to 3%.
- ▶ All impact of up to 6%.

- ▶ Custom depends on the counters enabled, but it is reasonable to expect somewhere between 2% - 6%.

Getting more information about statistics sets

The WebSphere Information Center has extensive information to assist the administrator in understanding exactly which metrics are set for a particular level and to appreciate the potential impacts of using the statistics.

If we consider the number of components that make up an application server, as shown in Figure 16-6, there are many PMI counters available to help monitor the application server.

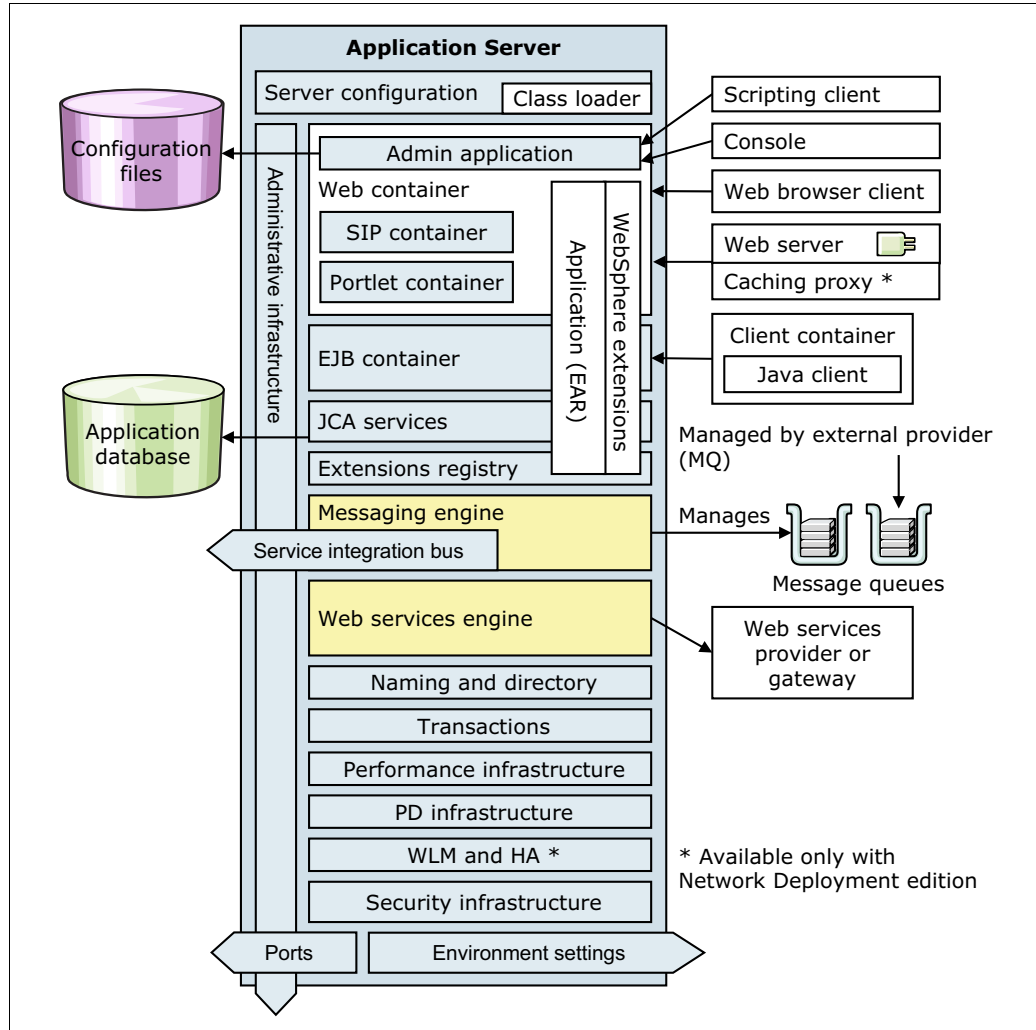


Figure 16-6 Application server components

The WebSphere Information Center provides a summary of PMI counters to help administrators understand the variety of counters that are available in each of the different counter classifications in the application server. A good place to start with gathering information is the article “Enabling PMI data collection” at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/tprf_pmi_encoll.html

From the links in the information center article above, to the counter classifications, it is possible to navigate and view the individual counters that each counter classification contains.

The following topics in the Information Center can provide more information:

- ▶ General PMI data organization, which is at the following website:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/rprf_dataorg.html
- ▶ WebSphere Application Server supports the Eclipse framework for extensible applications. A key part of this framework is the implementation of the Extension registry. These counters are only relevant when referring to extensible applications. For more information, go to the following website:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.base.iseries.doc/ae/cweb_extensions.html
- ▶ Service integration bus counters, which is at the following website:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.base.iseries.doc/ae/rprf_sibcounter.html
- ▶ Proxy counters, which is at the following website:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/rprf_proxycounter.html

16.2.2 Enabling request metrics

Enabling request metrics is a cell-wide configuration and when activated, it is activated for all servers in the cell. Complete the following steps to enable request metrics:

1. In the administrative console, click **Monitoring and Tuning** → **Request Metrics** (Figure 16-7 on page 563).

General Properties

Prepare Servers for Request metrics collection

Components to be instrumented

None
 All
 Custom

Select Components to instrument

AsyncBeans ▲
 EJB ▲
 JCA ▲
 JDBC ▲

* Trace level
 Hops ▼

Request Metrics Destination

Standard Logs
 Application Response Measurement(ARM) agent

Agent Type
 ARM40 ▼

ARM transaction factory implementation class name

Figure 16-7 Request metrics window

2. To enable request metrics:
 - a. Select **Prepare servers for request metrics collection**.
 - b. Choose a monitoring level from the Components to be instrumented section of the window.
 - c. Choose a trace level.
 - d. Choose a destination from the Request Metrics destination section of the Request Metrics window.

When configured, the servers must be restarted for request metrics to be enabled. The servers must also be stopped when disabling request metrics.

Understanding component instrumentation and trace levels

Trace levels and component instrumentation work together to determine if the request is instrumented. The component instrumentation levels are shown in Figure 16-8.

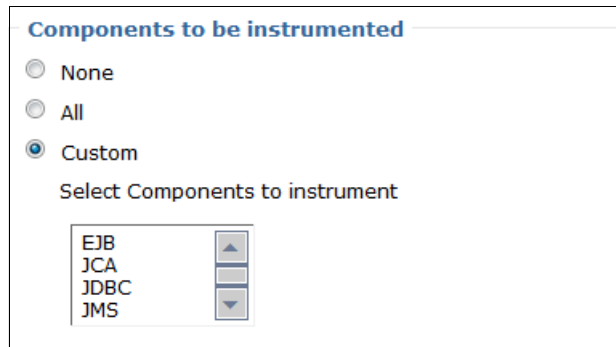


Figure 16-8 Components to be instrumented

If you select **All**, all components are monitored based on trace level settings.

If you select **Custom**, you can select the components to be monitored. Data is collected from the components if the trace level also calls for the capturing of data from this component.

Note: When a component is defined as an edge component, meaning the request enters or exits the application server through the component, then this component is instrumented even if it is not selected as part of the custom component listing.

Working in conjunction with the component instrumentation levels is the trace level (Figure 16-9).

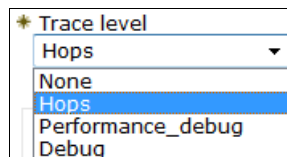


Figure 16-9 Request metric trace levels

The following trace levels are possible:

- ▶ **None:** No instrumentation is generated.
- ▶ **Hops:** Generates instrumentation information about process boundaries only. When this setting is selected, you see the data at the application server level, not the level of individual components, such as enterprise beans or servlets.
- ▶ **Performance_debug:** Generates the data at Hops level and the first level of the intra-process servlet and Enterprise JavaBeans (EJB) call (for example, when an inbound servlet forwards to a servlet and an inbound EJB calls another EJB). Other intra-process calls, such as naming and service integration bus (SIB), are not enabled at this level.
- ▶ **Debug:** Provides detailed instrumentation data, including response times for all intra-process calls. Note that requests to servlet filters are only instrumented at this level.

Note: Information about working with instrumentation and trace levels is provided in 16.4.4, “Request level details” on page 581.

Important: Request metrics are checked starting with the HTTP plug-in for some web-related settings. The HTTP-plug-in configuration must be regenerated and propagated after enabling request metrics.

Using request metric filters

One final way that can be used to control the request metric instrumentation is to use request metric filters. Filters provide a way to specifically target flows and components to reduce the impact of broad monitoring and to also make it easier to analyze the captured data by reducing the amount data that is captured.

It is important to understand, however, that filters are implemented as edge component filtering, not as intra-component processing, so an EJB filter is not effective if the EJB is always invoked from a servlet. In this case, it is the URI that needs filtering, not the EJB. Filters are applied on edge components.

Filters are configured by selecting the **Filters** link from the Additional properties section of the Request metrics window. This action opens the Request Metrics Filter window shown in Figure 16-10.

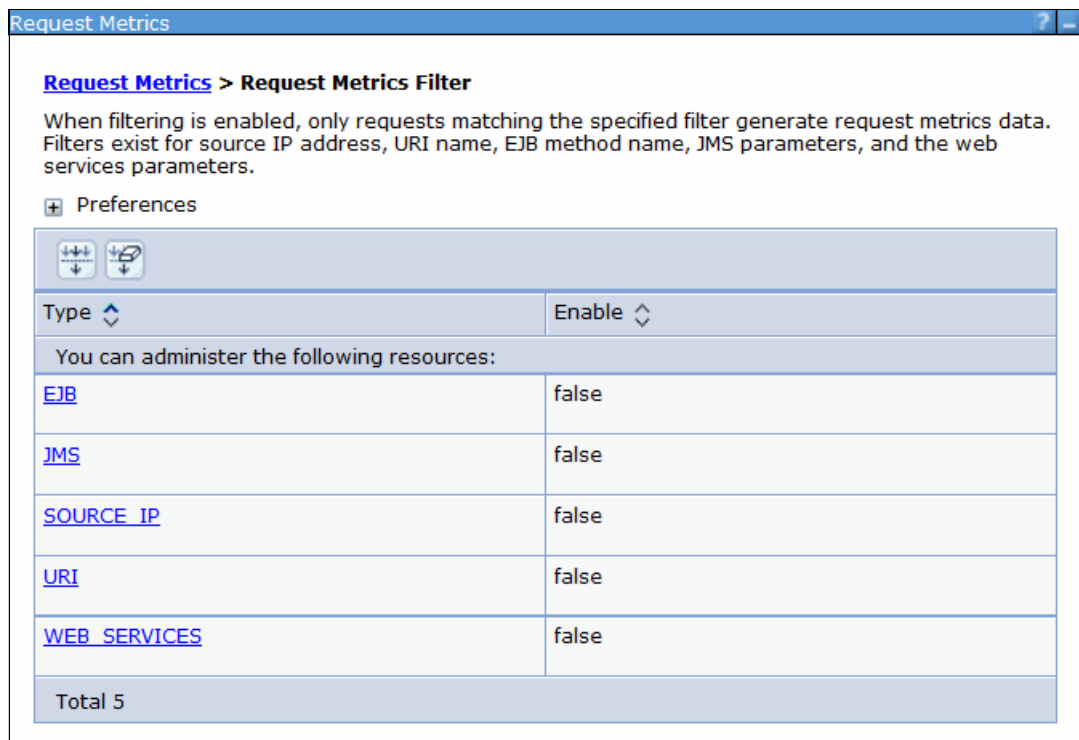


Figure 16-10 Request Metrics Filter window

Using filters, fine-grained controls can be applied to the different edge types of EJB, JMS, IP address, URI, and web services. The first step is to specify the filter. An example of each type can be seen by using the administration console to view that type of filter. For example, selecting the **URI** link in Figure 16-10 takes you to the URI window shown in Figure 16-11 on page 566.

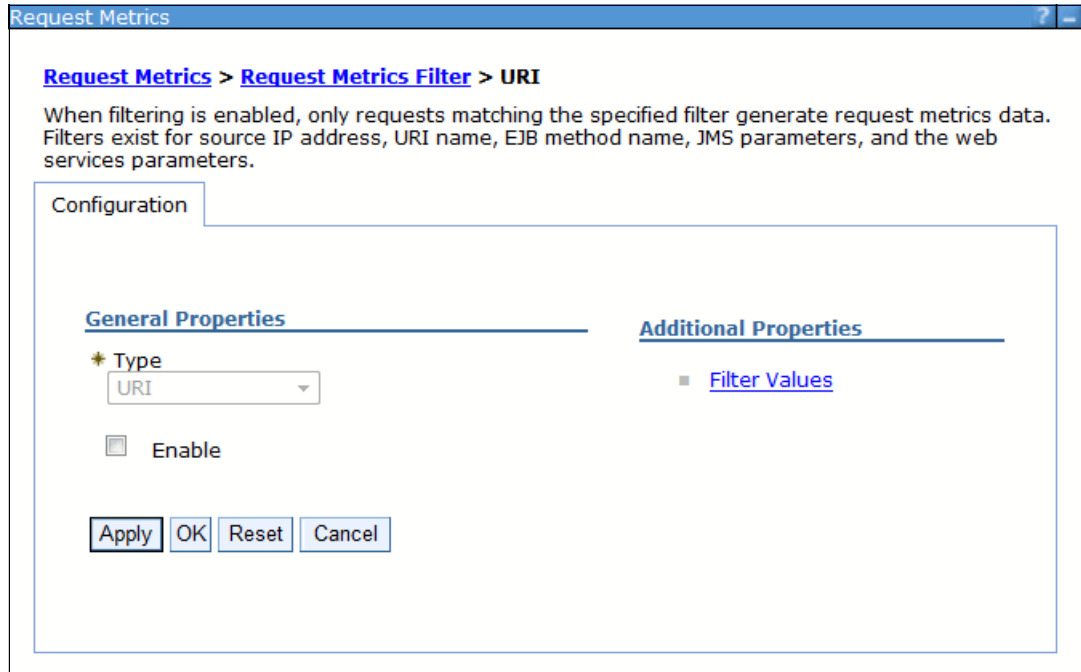


Figure 16-11 URI window

The Enable option must be selected for the filters to be enabled. Then the filters are used along with the component and trace level settings to determine which components are instrumented.

Note: Enabling filters requires an application server restart.

Select the **Filter Values** link in the Filter window to add or edit filters. This window is also where the default example filters are displayed (Figure 16-12).

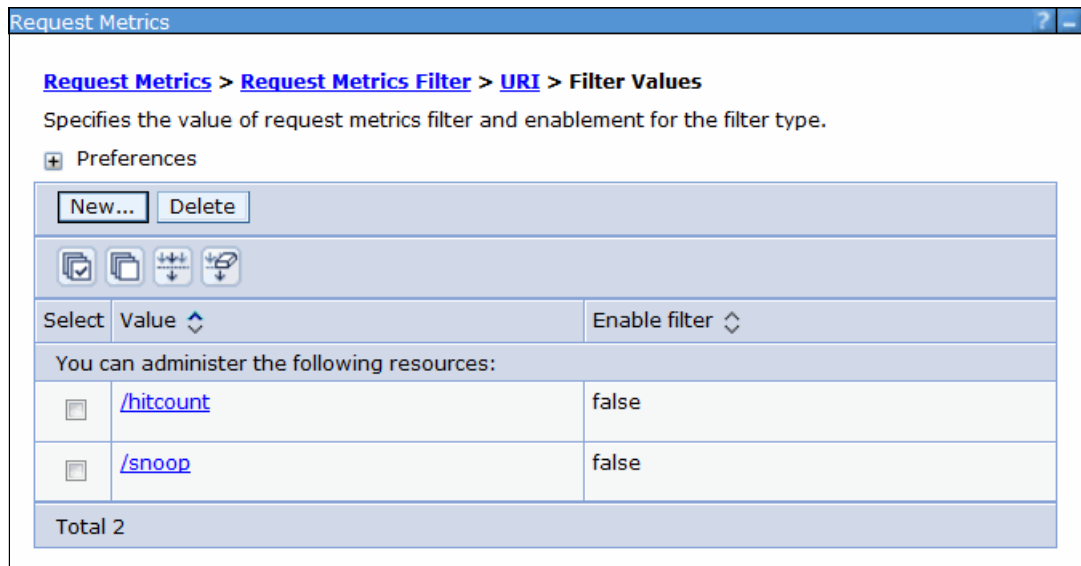


Figure 16-12 Default filter values displayed in filter value window

Each filter type has its own syntax that is appropriate for the type and allows the use of wildcard settings in the filters if desired. For example, the EJB filter specifies a method class or package that sets the scope of the filtering. The example URI filter value supplied for EJB filters are:

- ▶ `com.yourco.package.Class.method`
- ▶ `com.yourco.package.Class2.*`

Tip: Enabling / disabling a filter requires that you restart the server. It is important to plan the component levels and filters that an application might require to minimize the need to stop and restart servers.

Destination type considerations

The final consideration when configuring the request metrics is where the metrics are gathered. There are two types of supported destinations:

- ▶ Data can be logged with standard logs. In this configuration, the instrumented components are logged to the `SystemOut.log` file.
- ▶ Data can also be collated in an Application Response Measurement (ARM) data collector. In this case, the data is normally then moved to a monitoring system for analysis and display (for example, using IBM Tivoli Composite Application Manager for Transactions).

Tip: When configuring ARM agents for use with the application server, follow the installation instructions provided with the specific agent. For more information about ARM agents, see the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/cprf_arm.htmls

Both logging types can be activated at once. Writing to standard logs is not a best practice as a long term monitoring strategy because the impact can be higher than is desirable.

Impact of request metrics

The impact of request metrics can vary significantly based on the components being monitored and the complexity of the request execution within the monitored components. There are no specific metrics on what the impact is, but it is reasonable to assume that request metrics on every request and component might incur more impact than is desired. An organization must consider and plan carefully the interactions that it wants to monitor, then measure the specific impact associated with configuring request metrics for these components.

16.3 Viewing the monitoring data

WebSphere Application Server provides an interface for viewing the monitored data. The interface is the Tivoli Performance Viewer (TPV), which is in the administrative console.

16.3.1 Starting TPV monitoring and configuring settings

To work with the TPV from the administrative console:

1. Click **Monitoring and Tuning** → **Performance Viewer** → **Current Activity**.

- Select the server(s) that are to be monitored, and click **Start Monitoring**, as shown in Figure 16-13.

Tip: If you are only starting monitoring on a single server, monitoring can be started by simply clicking the server link and navigating to the TPV viewer.

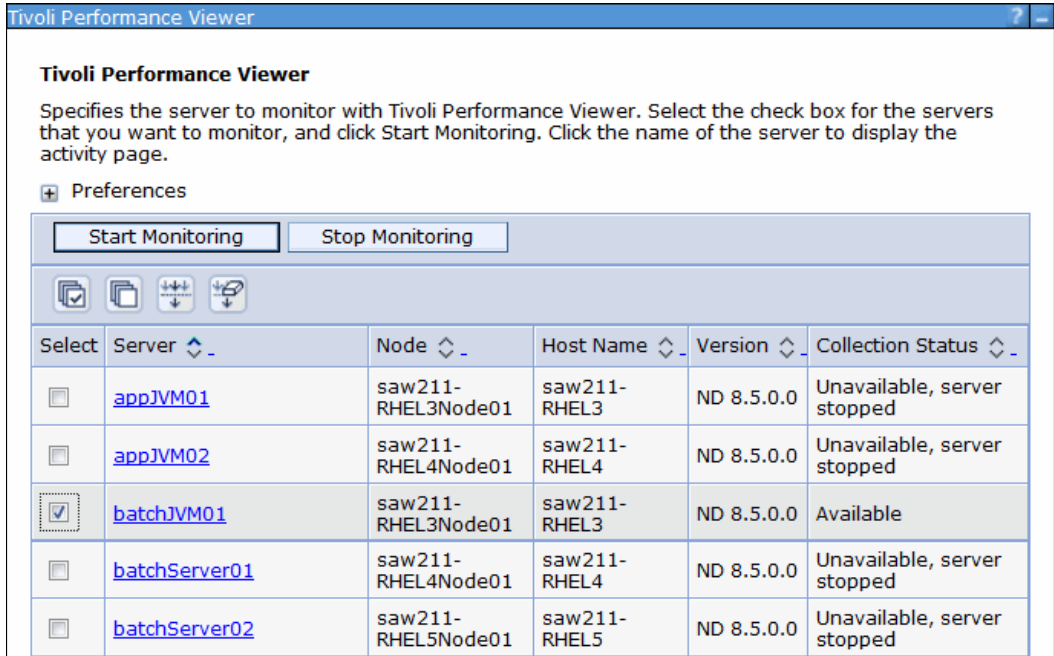


Figure 16-13 Start Server monitoring

After monitoring is started, a message is returned in the messages section of the window, and the Status column of the server is updated to *Monitored*.

- The PMI data can only be observed one server at a time when using a single user session. Select the server name link to navigate to the Tivoli Performance Viewer window (Figure 16-14).

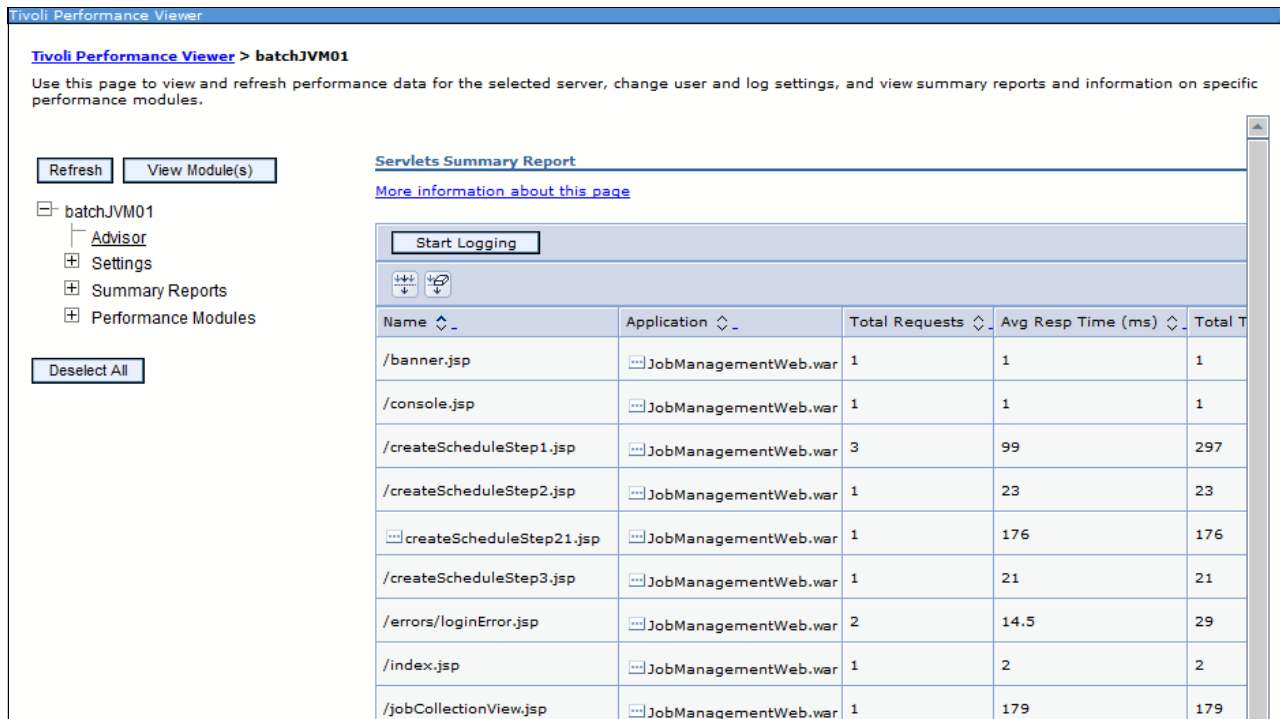


Figure 16-14 Tivoli Performance Viewer

The default view for this window shows the Servlet Summary Report pane, which indicates recent servlet activity and the TPV tree navigation pane.

Note: Information about the different ways that data can be viewed is provided in 16.3.2, “Exploring Tivoli Performance Viewer data views” on page 571, but before exploring the data, let us first take a look at the settings menu to examine the User and Logging settings.

- The user settings are reached by expanding the **Settings** category and selecting **User**. This window helps you control how much data is retained and how often data samples are taken in the live system (Figure 16-15 on page 570).

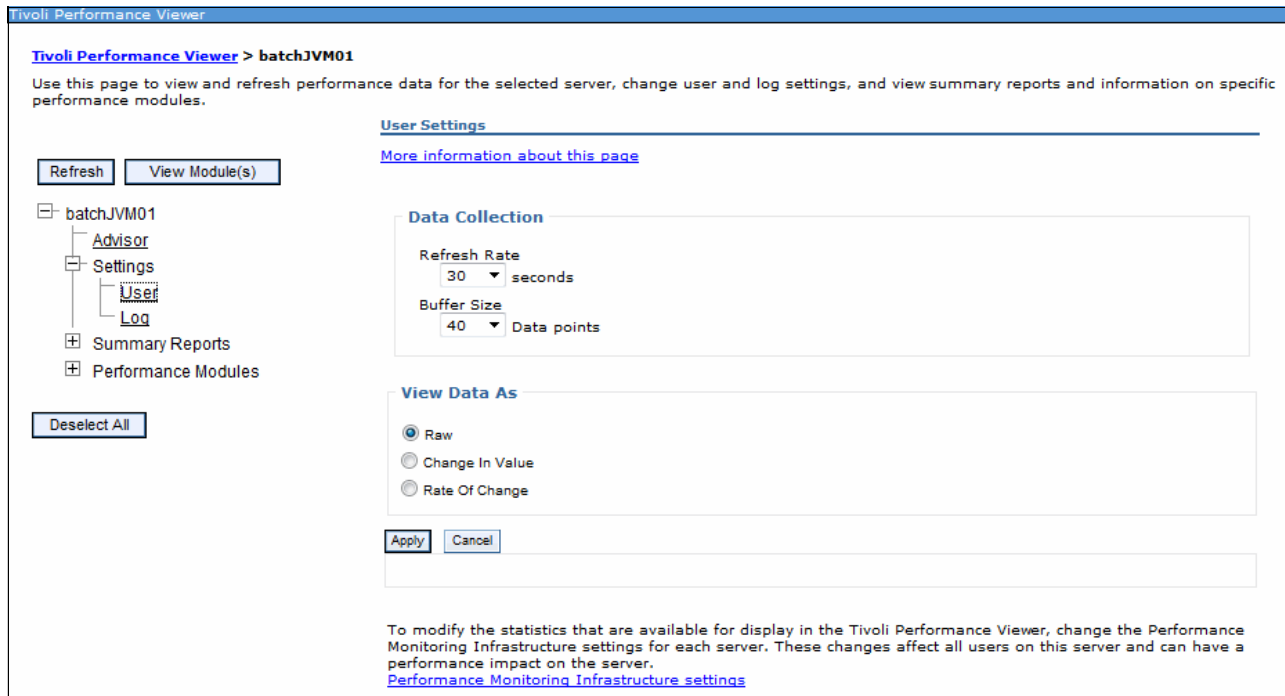


Figure 16-15 User settings

The user settings are significant and can have a direct impact on the performance of the server. The two key configuration settings are in the Data Collection section of the window:

- The refresh rate indicates the interval between data sampling. Higher frequency rates mean that the server will gather and report on statistics more frequently, adding load to the servers that are collecting data.
- The buffer size indicates the number of data points that are kept. More data points simply mean that the PMI data will require more memory.

For a stand-alone server, this means that PMI data at high frequency and high buffer re-initiation need more processing time and more memory.

In a distributed server model, the load is shared, but some settings might still need to be tuned. The data is collected at the node level and stored in memory on the node agent. Thus, if a node has many servers, the memory requirements of the node agent must be adjusted.

Also, in a distributed server configuration, the data is viewed from the deployment manager. Thus, to process the data, the deployment manager must have adequate memory and CPU resources. Consider that more than one administrator might be observing data at once.

5. A powerful feature of TPV is the ability to record PMI data and then replay the data later in a different deployment manager as though it were in real time or with options to fast forward and rewind.

Logging is started by clicking the **Start Logging** button shown in the TPV viewing window (Figure 16-14 on page 569). However, before clicking this button, the Log settings must be configured. Click **Settings** → **Log** to get the Log settings.

Examining the log settings that are available, the administrator is faced with several configuration choices:

- Duration: The logging of PMI data has with it a certain amount of impact (resulting from logging to a file, the buffering of data in memory, and disk usage for log storage). It is not intended to be used as a long term production monitoring strategy. Thus, when logging is enabled, it is configured to be disabled after a period of time.

PMI data logging used in short durations can be used to capture runtime characteristics that might need further investigation for sharing with development and troubleshooting specialists.
- Maximum file size and Maximum number of historical files: The settings for maximum file size and the number kept that are appropriate for your environment depend on two conditions:
 - How much PMI data is enabled.
 - The data sampling frequency (as configured in the user settings).
- File name: The server name and the time at which the log is started is appended to the file name specified to help users identify a log file.
- Log output format: The other configuration item of consequence is the format type. The default is XML, but the binary format requires a smaller footprint on the disk. If logging larger amounts of data, the binary logging format might be more suitable.

16.3.2 Exploring Tivoli Performance Viewer data views

Tivoli Performance Viewer has three primary types of data that can be viewed:

- ▶ Summary reports
- ▶ Performance modules
- ▶ Advisors

Summary reports

The summary reports provide a general overview in a tabulated format of the current system performance. The reports that are available include:

- ▶ Servlets
- ▶ EJBs
- ▶ EJB Methods
- ▶ Connection Pool
- ▶ Thread Pool

Servlet and EJB summary reports can be useful for identifying the application resources that are the most busy and to the extent that averages can be used to identify candidates that might warrant further investigation as to their performance. Together with request metrics, results from the summary reports can help identify possible candidates for request metric filters.

The information for Connection Pool and Thread Pool utilization, while useful, can also be easily determined by monitoring the metrics in the performance modules.

Note: For summary reports to be available, the PMI data must be reported at a sufficiently detailed level. For the basic PMI data level, only the Servlets and EJBs reports are available. Higher or custom PMI settings must be specified for the other reports.

Figure 16-16 shows an example of the Servlets report.

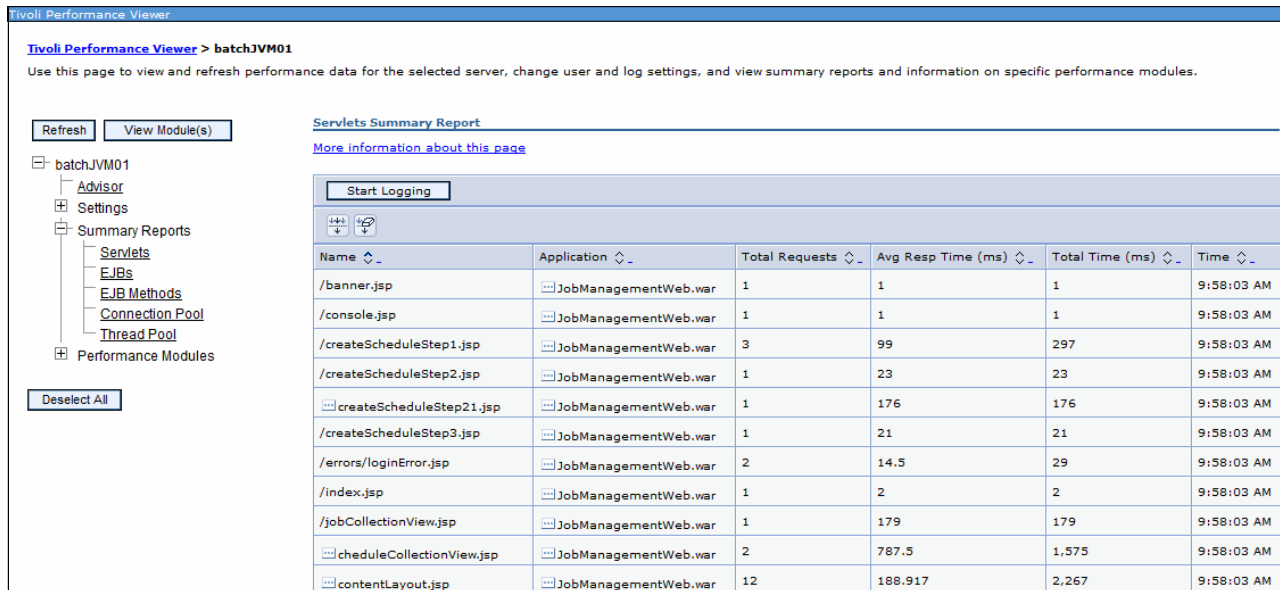


Figure 16-16 Servlets summary report

Figure 16-17 shows an example of the EJBs summary report.

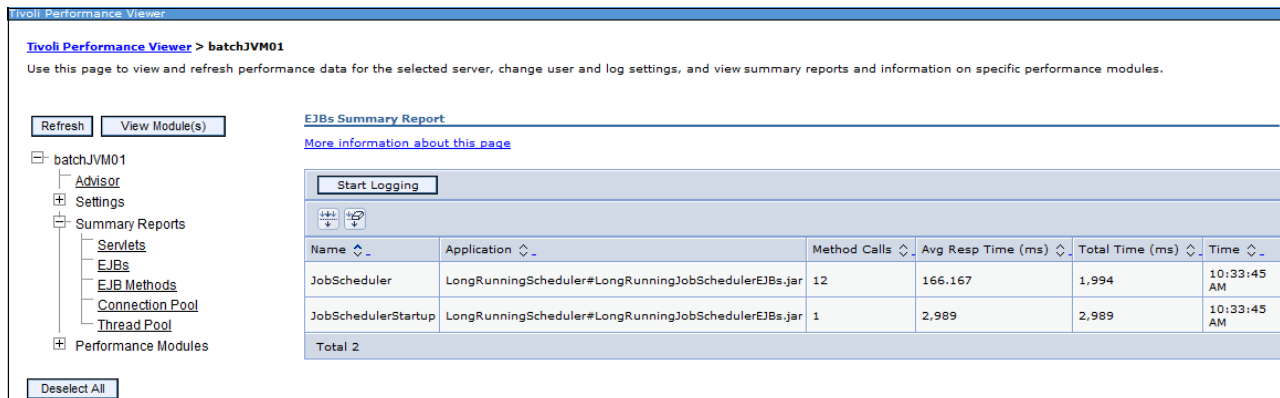


Figure 16-17 EJB summary

Tip: This summary data can be used to identify EJBs that are invoked often. If the system is working correctly, the slowest bean, on average, might be worth investigating if the time exceeds expected SLAs, and so on.

It can also be useful to identify what work is not happening. In Figure 16-17, there are two message-driven beans that have not been invoked. This situation, in itself, might be unusual and warrant investigation.

Performance modules

Performance modules provide a tracking mechanism for each of the PMI counters that are active. These counters are categorized under their different PMI data classifications. The data can be viewed as a table or graphically displayed. WebSphere uses dojo technology for plotting the data instead of Scalable Vector Graphics (SVG). The dojo technology provides a better user experience and is more memory and processor efficient.

The performance modules provide a powerful runtime view of the data as it is being recorded to allow the administrator to analyze the current system health.

The data that can be displayed is limited depending on the PMI level that is configured. The administrator can select one or more metrics for the current PMI level, as shown in Figure 16-18 and then click the **View Modules** button at the top of the window.

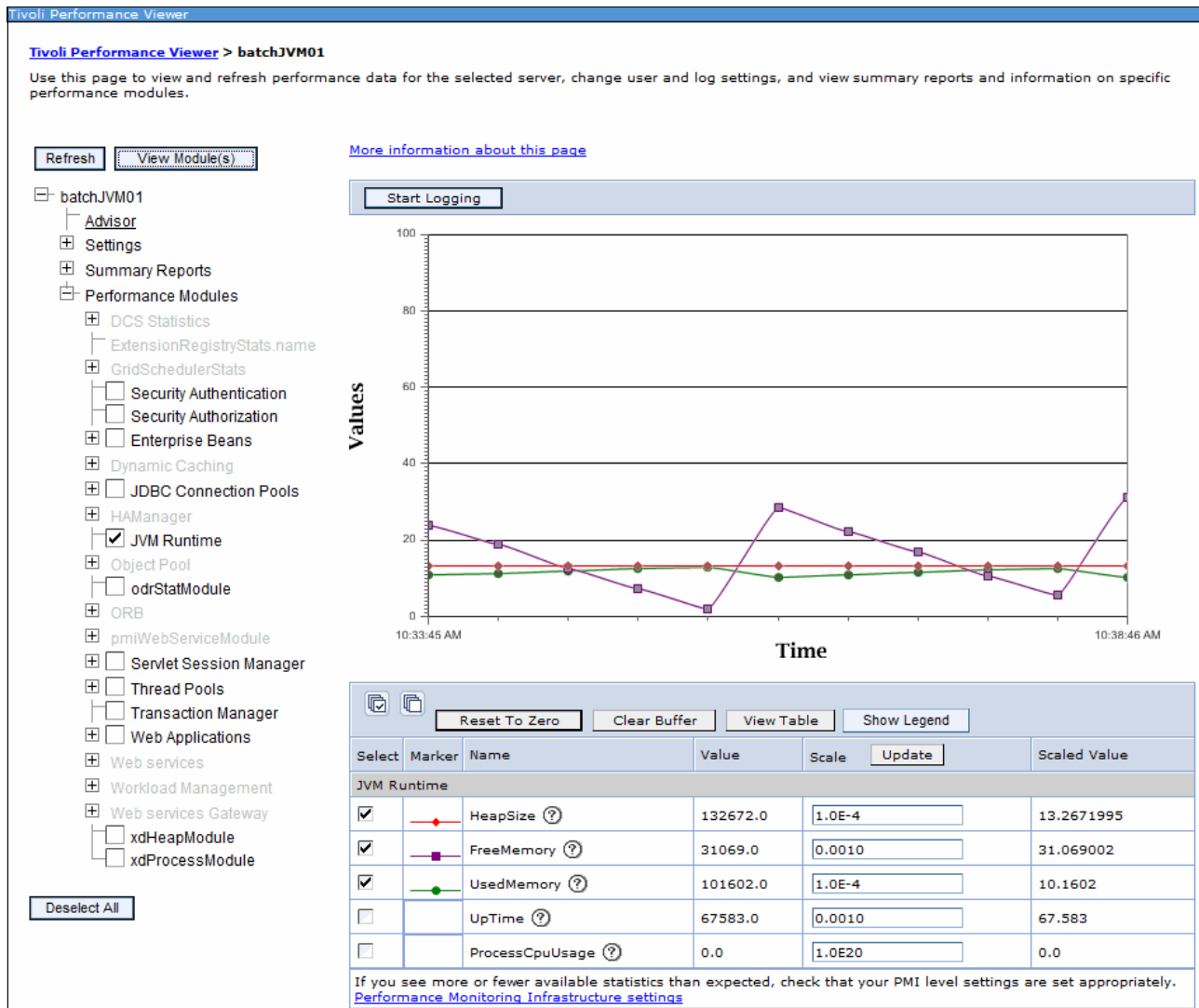


Figure 16-18 Performance modules

After the performance modules are selected, the data window provides a view of the data that is being collected. By default, the graphical view is used. The most recent data point and the graph key for the different counters can be seen under the graph. The graph can then be customized to include or exclude counters from the chosen set of PMI data. To view the data

in table format, as shown on Figure 16-19, click **View table**. Scales can also be adjusted by changing the scale and clicking **Update**.

Tivoli Performance Viewer
Tivoli Performance Viewer > batchJVM01
 Use this page to view and refresh performance data for the selected server, change user and log settings, and view summary reports and information on specific performance modules.

Refresh View Module(s) More information about this page

Start Logging

| Time | JVM Runtime HeapSize | JVM Runtime FreeMemory | JVM Runtime UsedMemory |
|-------------|----------------------|------------------------|------------------------|
| 10:41:46 AM | 133696.00 | 52679.00 | 81016.00 |
| 10:41:16 AM | 132672.00 | 3307.00 | 129364.00 |
| 10:40:46 AM | 132672.00 | 8250.00 | 124421.00 |
| 10:40:16 AM | 132672.00 | 14423.00 | 118248.00 |
| 10:39:46 AM | 132672.00 | 19445.00 | 113227.00 |
| 10:39:16 AM | 132672.00 | 26038.00 | 106633.00 |
| 10:38:46 AM | 132672.00 | 31069.00 | 101602.00 |
| 10:38:16 AM | 132672.00 | 5598.00 | 127073.00 |
| 10:37:46 AM | 132672.00 | 10515.00 | 122156.00 |
| 10:37:16 AM | 132672.00 | 16887.00 | 115784.00 |
| 10:36:46 AM | 132672.00 | 22123.00 | 110548.00 |
| 10:36:16 AM | 132672.00 | 28524.00 | 104147.00 |
| 10:35:45 AM | 132672.00 | 2153.00 | 130518.00 |
| 10:35:15 AM | 132672.00 | 7267.00 | 125404.00 |
| 10:34:45 AM | 132672.00 | 12461.00 | 120210.00 |
| 10:34:15 AM | 132672.00 | 18832.00 | 113839.00 |
| 10:33:45 AM | 132672.00 | 24038.00 | 108633.00 |
| Total 17 | | | |

Figure 16-19 Table view

Performance advisors

The last of the TPV data sets contains the TPV performance advisors. These advisors analyze the data using rules that are pre-configured by IBM based on best practice and performance observations. The advisors provide tuning best practices to help improve the performance.

The types of items that TPV provides advice about includes several well known performance hot spots:

- ▶ Object Request Broker service thread pools
- ▶ Web container thread pools
- ▶ Connection pool size
- ▶ Persisted session size and time
- ▶ Data source statement cache size
- ▶ Session cache size
- ▶ Dynamic cache size
- ▶ Java virtual machine heap size
- ▶ DB2 Performance Configuration wizard
- ▶ Connection use violations

Advisors are more of a tuning aid than a monitoring tool set and are not suitable for use in production environments. But advisors can be a useful aid in identifying well known performance hot spots in the current server configurations in testing.

Advisors are best used when:

- ▶ A reasonable load can be driven to the application server utilizing significant CPU (50+%).
- ▶ You want help in tuning a server while establishing initial performance benchmarks.

For more information about the advisors, see the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/cprf_whyuseperfadvisors.html

To view the Advisors window, click the **Advisor** link in the TPV menu window. The advisor provides advice to the administrator and from the advisors view, the different advice statements can be selected and further analyzed by the administrator, as shown in Figure 16-20.

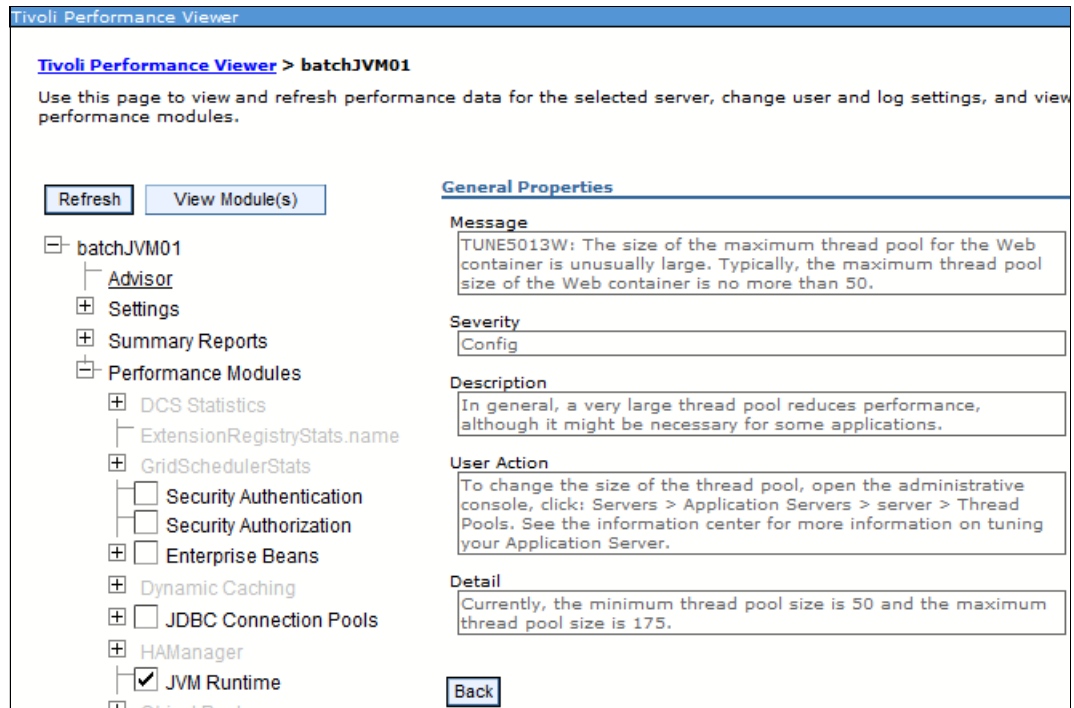


Figure 16-20 Advice window

16.4 Monitoring examples

In this section, a variety of monitoring data is used to provide examples and information about what that data tells an administrator.

Important: Although the statistical data that is observed in PMI and request metrics provides a powerful understanding of what is occurring in the environment, it is important to remember that the counters and statistics are mostly averages. You must always verify that the statistics make sense for your environment. For example, if an EJB seems to be having good response time, but it is throwing and catching an exception shortly after entering the bean, it still appears to have excellent response time. Response time alone is not enough to indicate a healthy system.

16.4.1 JVM memory and CPU usage

Environmentally, the way the applications and application servers use memory and CPU is important to the overall server performance. The application server PMI data provides some basic level monitoring capabilities.

The JVM Used memory and ProcessCpuUsage counters can be monitored and displayed in the TPV graph. Figure 16-21 shows an example of this data.

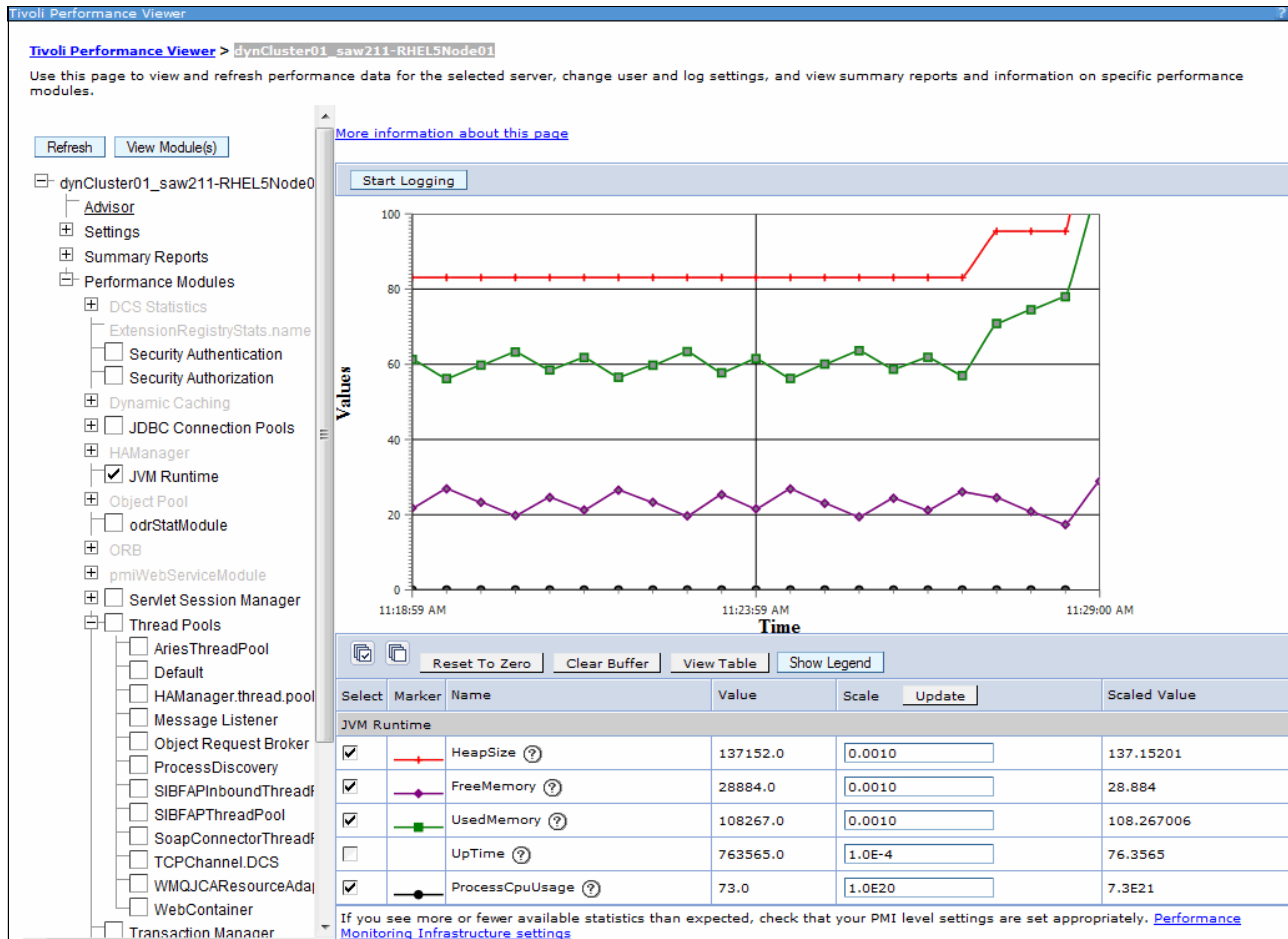


Figure 16-21 JVM memory and CPU usage

The report can be viewed by clicking **Monitoring and Tuning** → **Performance Viewer** → **Current Activity** → **Performance Modules** → **JVM Runtime**.

This graphical view is a good way to visually check that memory is not constantly growing and is stabilized over time by garbage collection. It also gives a good indication of how frequently garbage collection might occur.

However, when tuning memory sizes, use verbose garbage collection and a combination of the application server support tools available in IBM Support Assistant for the analysis of the memory usage.

Note: TPV provides a good representation of what occurs with the JVM memory, but the IBM Garbage Collection and Memory Visualizer tool and the IBM Pattern Modelling and Analysis for Java Garbage Collector tool provide more insightful help with memory-related configurations to assist tuning memory. For example, you can be provided with advice on heap sizes and garbage collection algorithms based on the patterns observed in the garbage collection log.

These tools are available as add-ons to IBM Support Assistant. Use IBM Support Assistant set up wizards to download these and other Support tools. For more information, refer to the following web sites:

- ▶ IBM Support Assistant:

<http://www.ibm.com/software/support/isa/>

- ▶ Complete list of ISA available add-ons:

<http://www.ibm.com/support/docview.wss?uid=swg27013116>

- ▶ IBM Education Assistant module for The IBM Garbage Collection and Memory Visualizer:

http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.was_v7/was/7.0/ProblemDetermination/WASv7_GCMV0verview/player.html

Verbose garbage collection

Assuming there is appropriate disk space, a monitoring strategy must include verbose garbage collection.

To enable verbose garbage collection, click **Servers** → **Server Types** → **WebSphere application servers** → **<your server>** → **Server Infrastructure** → **Java and process management** → **Process definition** → **Additional Properties** → **Java Virtual Machine** and select the **Verbose garbage collection** option.

When this field is enabled, a report is written to the output stream each time the garbage collector runs. The information can usually be found in native log files. In Version 7, WebSphere used the optimal throughput (optthruput) algorithm for garbage collection. Now generational concurrent garbage (gencon) collection is being used, which allows performance improvements. Because optthruput uses a large contiguous heap shared by all threads, when garbage collection is invoked, all of this area is scanned. This policy is beneficial for applications that demand optimal throughput, but it has long pause times as a side effect. Generational concurrent divides the heap memory in two pieces:

- ▶ Nursery, for new objects
- ▶ Tenured, for aged objects

So, in this policy, the time spent to scan one of the areas is smaller.

IBM Support Assistant provides tools that make the garbage collection analyses easier. For more information about the garbage collection on WebSphere, read the following articles:

http://www.ibm.com/developerworks/websphere/techjournal/1106_bailey/1106_bailey.html

http://www.ibm.com/developerworks/websphere/techjournal/1108_sciampacone/1108_sciampacone.html

16.4.2 Threading resources

Assuming that there is enough CPU and memory, it is reasonable to surmise that thread pools along with resource connections are a major factor in understanding the limits of throughput on the application server. The various thread pools in the application server control the entry points for requests into the system. If the pool is exhausted, requests to the system are queued and have to wait. From a monitoring perspective, it is preferable that thread pools are not constantly exhausted and running at their maximum.

Before monitoring the thread pools, the administrator needs to first be aware of what types of thread pools their application uses. The following application topologies provide some insight into what needs to be considered in understanding which thread pools might be important to application runtime throughput.

In the first example, consider a clustered application server environment where the deployed application has both web and EJB components deployed together in the JVM (Figure 16-22). For performance reasons, WebSphere Application Server will use process affinity when invoking the EJBs and sending requests from the web container to the EJB container in the same JVM.

In this scenario, threads are not swapped and the requests are executed on web container threads. Even though EJBs are extensively used in the application, the ORB thread pool is not used in this application topology. Web container threads in this topology control the concurrency of the application. This is the topology used by the sample application and is common to many JEE applications.

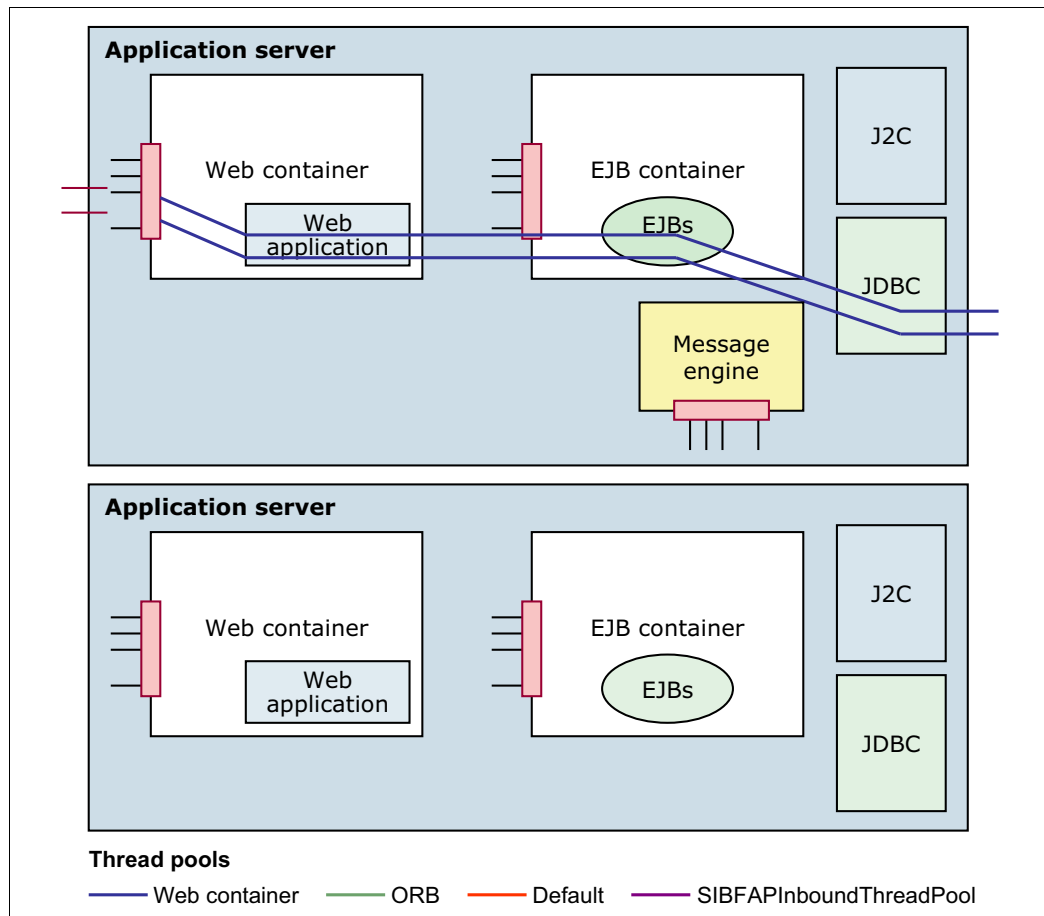


Figure 16-22 Web application

But what if the EJB components were deployed in a separate JVM, as shown in Figure 16-23?

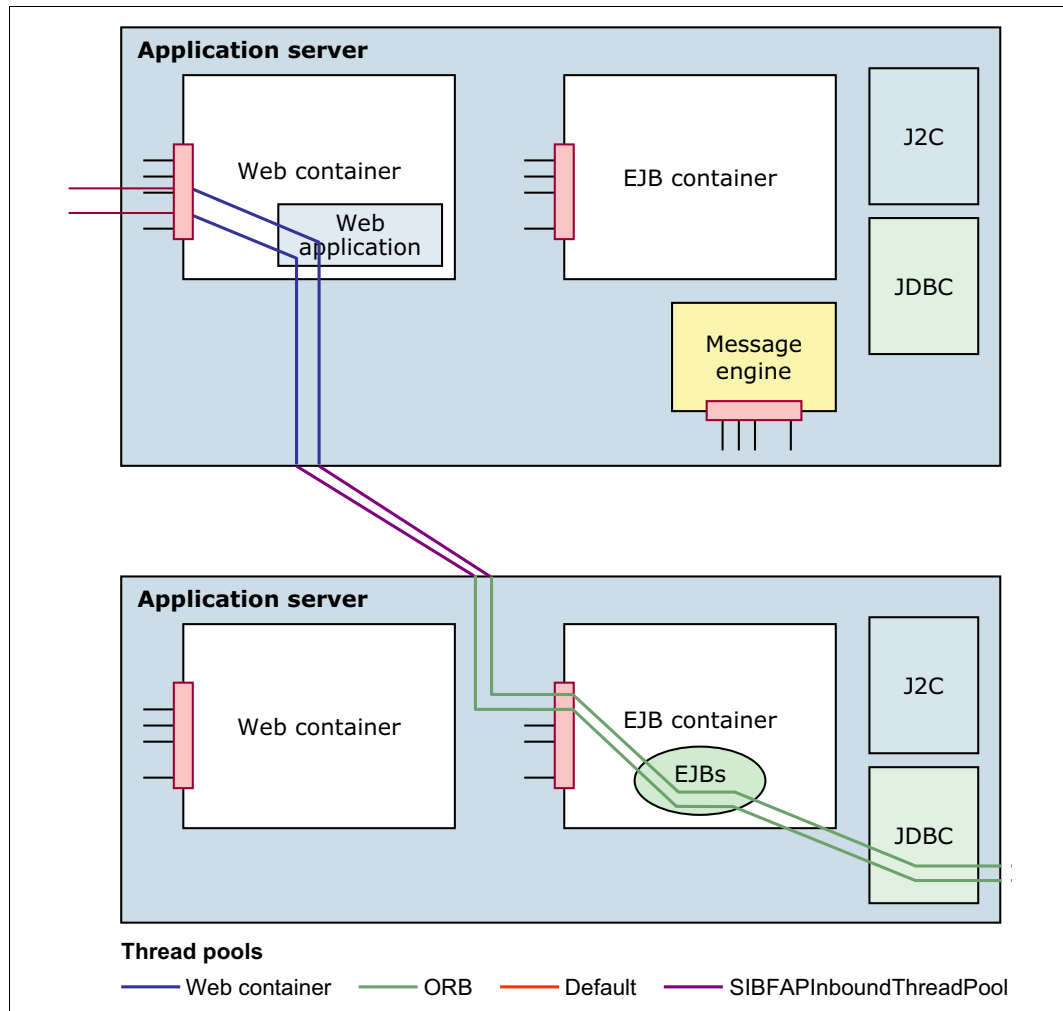


Figure 16-23 Distributed application model

In such a scenario, the ORB thread pool in each application server becomes equally important as the web container thread pool. Request throughput is now controlled by the web container, the ORB thread pool, and other parameters, such as memory and connections to external resources.

Different environment resources use different thread pools and a key consideration to understand as an administrator is what are the components with which the deployed applications will interact, and what is the likely environmental impact.

Tivoli Performance Viewer provides a number of resources that can assist you with understanding ThreadPool utilization. Consult the Thread Counters to learn more. Reports can be viewed by clicking **Monitoring and Tuning** → **Performance Viewer** → **Current Activity** → **Performance Modules** and then expand the **Thread Pools** section.

16.4.3 Database interactions

One of the hot spots for monitoring is interactions with external servers, especially interactions with databases. PMI provides a good set of metrics at even the basic level for these types of interactions.

Consider the data snapshot shown in Figure 16-24. This snapshot was taken in the example environment while the server was under load. You can view this report by clicking **Monitoring and Tuning** → **Performance Viewer** → **Current Activity** → **Performance Modules** → **JDBC Connection Pools**. The monitoring level in effect is the Basic level.







| Select | Marker | Name | Value | Scale | Update | Scaled Value |
|-------------------------------------|---|------------------------|----------|--------|-------------------------------------|--------------|
| jdbc/TradeDataSource | | | | | | |
| <input checked="" type="checkbox"/> |  | CreateCount (?) | 1.0 | 1.0 | <input type="text" value="1.0"/> | 1.0 |
| <input checked="" type="checkbox"/> |  | CloseCount (?) | 0.0 | 1.0E20 | <input type="text" value="1.0E20"/> | 0.0 |
| <input checked="" type="checkbox"/> |  | PoolSize (?) | 1.0 | 1.0 | <input type="text" value="1.0"/> | 1.0 |
| <input type="checkbox"/> | | FreePoolSize (?) | 1.0 | 1.0 | <input type="text" value="1.0"/> | 1.0 |
| <input type="checkbox"/> | | WaitingThreadCount (?) | 0.0 | 1.0E20 | <input type="text" value="1.0E20"/> | 0.0 |
| <input type="checkbox"/> | | PercentUsed (?) | 0.0 | 1.0E20 | <input type="text" value="1.0E20"/> | 0.0 |
| <input type="checkbox"/> | | UseTime (?) | 38.53416 | 1.0 | <input type="text" value="1.0"/> | 38.53416 |
| <input type="checkbox"/> | | WaitTime (?) | 0.0 | 1.0E20 | <input type="text" value="1.0E20"/> | 0.0 |
| JDBC Connection Pools | | | | | | |
| <input checked="" type="checkbox"/> |  | CreateCount (?) | 4.0 | 1.0 | <input type="text" value="1.0"/> | 4.0 |
| <input checked="" type="checkbox"/> |  | CloseCount (?) | 0.0 | 1.0E20 | <input type="text" value="1.0E20"/> | 0.0 |
| <input checked="" type="checkbox"/> |  | PoolSize (?) | 4.0 | 1.0 | <input type="text" value="1.0"/> | 4.0 |
| <input type="checkbox"/> | | FreePoolSize (?) | 3.0 | 1.0 | <input type="text" value="1.0"/> | 3.0 |
| <input type="checkbox"/> | | WaitingThreadCount (?) | 0.0 | 1.0E20 | <input type="text" value="1.0E20"/> | 0.0 |
| <input type="checkbox"/> | | PercentUsed (?) | 2.0 | 1.0 | <input type="text" value="1.0"/> | 2.0 |
| <input type="checkbox"/> | | UseTime (?) | 63.33555 | 1.0 | <input type="text" value="1.0"/> | 63.33555 |
| <input type="checkbox"/> | | WaitTime (?) | 0.0 | 1.0E20 | <input type="text" value="1.0E20"/> | 0.0 |

Figure 16-24 Basic PMI for database

From a monitoring perspective, this snapshot provides several useful statistics that can help the administrator understand if the database interaction is healthy. For measuring runtime health, the following counters are beneficial:

- ▶ PercentUsed indicates if the database connections are being overutilized or under utilized.

Depending on the application and capacity of the database, it is typically not a good sign if the database is 100% utilized. If a connection pool becomes 100% utilized all of the time after the system has been tuned, either the database might need more capacity to support more connections, or some type of error is occurring. For example, the application might have a connection leak. Utilization is a good indicator that database connections need some attention.

- ▶ Similarly, it is not unreasonable to have waiting threads on the data source because the resource is shared. From a monitoring perspective, it is a combination of the waiting thread count and the wait time that makes an interesting combination.

If the wait time and waiting thread count grow over time, this is an indication that the database might be responding more slowly than desired, or there are insufficient

connections available to support the load. How long is too long a wait time depends on application service level agreements and whether wait times occur all of the time or only on occasion under exceptional load.

- ▶ UseTime can help understand how much time is spent communicating with the database and can help to indicate if database response is degrading.

If the administrator believes there are response time errors with the database, request metrics can be useful in diagnosing with which components the application is spending its time.

JCA interactions

JCA is the more generic form of the JDBC and is used with standard adapters that comply with the JCA standard. The adapter can have connection pooling that is the same connection pooling used with JDBC. Probably the most common JCA adapter other than JDBC is the one used for JMS connections, whether connecting to the service integration bus or connecting to an external JMS provider, such as WebSphere MQ.

Hence, when working with JCA or JMS connections, the same indicators used for the JDBC are relevant with the exception of counters that are JDBC specific. When monitoring, it is beneficial in particular to monitor the waiting threads, their wait time, and the current pool size.

16.4.4 Request level details

When examining performance data, especially when tuning a server, it can be useful to have more detailed data. This situation is particularly true when first trying to pin down and explore bottlenecks. Request metrics can provide this lower-level information, showing where time is spent in a request.

Manually creating an application server perceived response time graph can help illustrate the response time of each component as the request is processed. The request metrics at the outer most or edge component are taken as well as all inner components because each one handles the request. Each new component represents a narrower view excluding the work done by the components that proceed it.

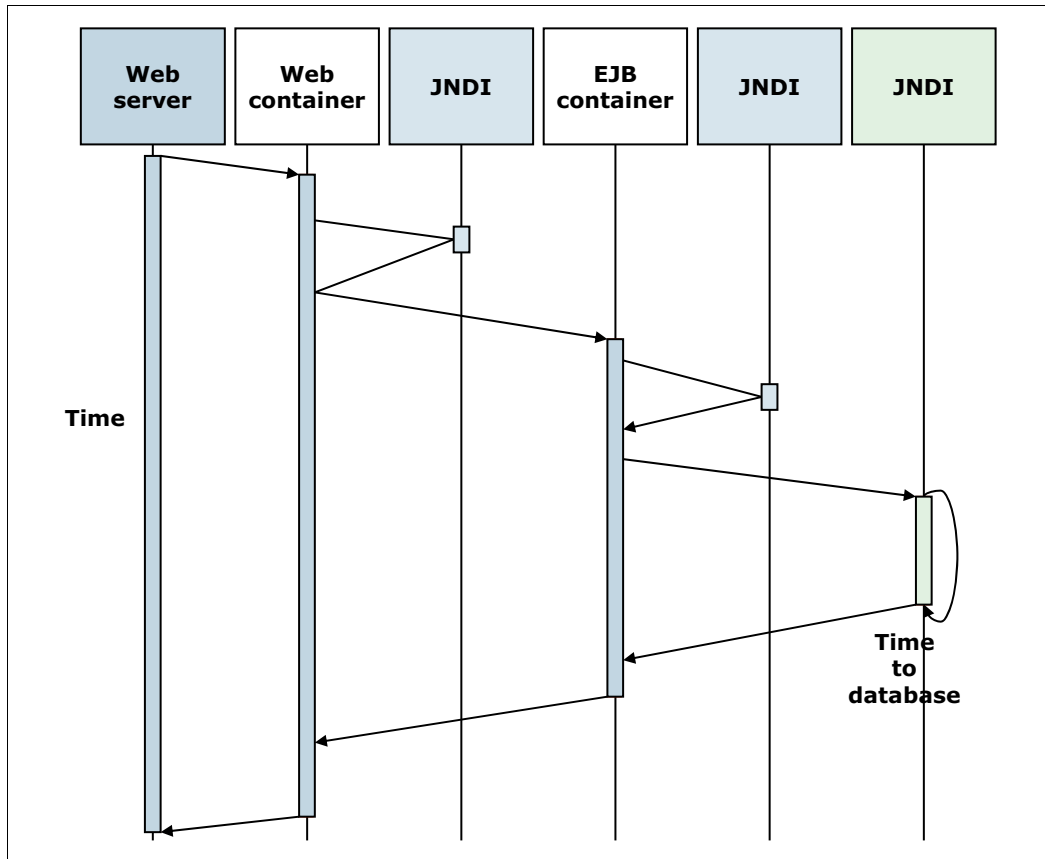


Figure 16-25 Time view of request metrics

Tip: The metrics, when printed to the standard logs, are printed from right to left with the innermost statistic reported first.

Request metrics are ideal for helping to identify the slowest and worst performing request types and can assist in identifying areas where performance can be improved. As a second phase, data from more detailed monitoring for the request type can be broken down to provide insight on where the bulk of time is being spent and provide insight into what areas of the code warrant investigation.

Unlike PMI, there is no built-in tool for request metric analysis, so for these examples, the data is extracted from the logs. IBM has tooling available in the Tivoli monitoring suites, but it is not part of the application server offering.

Example 16-1 shows the request metrics captured for a single request in the standard log file. In this example, request metrics are enabled with a trace level of hops and all components instrumented. As you can see, it is quite verbose even with a small amount of detail active.

Example 16-1 Simplified PMI data example

```
[7/15/11 11:49:56:343 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=6,event=1
type=JDBC
detail=javax.resource.spi.ManagedConnectionFactory.matchManagedConnections(Set,
Subject, ConnectionRequestInfo) elapsed=0
```

```

[7/15/11 11:49:56:390 BRT] 00000030 PmiRmArmWrapp I PMRM00031:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=7,event=1
type=JDBC detail=javax.resource.spi.ManagedConnection.getConnection(Subject,
ConnectionRequestInfo) elapsed=0
[7/15/11 11:49:56:421 BRT] 00000030 PmiRmArmWrapp I PMRM00031:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=8,event=1
type=JDBC detail=javax.resource.spi.XAResource.start(Xid, int) elapsed=0
[7/15/11 11:49:56:734 BRT] 00000030 PmiRmArmWrapp I PMRM00031:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=9,event=1
type=JDBC detail=java.sql.Statement.executeQuery(String) elapsed=297
...
Lots of lines deleted for simplification
...
[7/15/11 11:50:02:031 BRT] 00000030 PmiRmArmWrapp I PMRM00031:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=75,event=1
type=JDBC detail=java.sql.PreparedStatement.executeQuery() elapsed=516
[7/15/11 11:50:02:078 BRT] 00000030 PmiRmArmWrapp I PMRM00031:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=76,event=1
type=JDBC detail=javax.resource.spi.XAResource.end(Xid, int) elapsed=0
[7/15/11 11:50:02:078 BRT] 00000030 PmiRmArmWrapp I PMRM00031:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=77,event=1
type=JDBC detail=javax.resource.spi.XAResource.commit(Xid, boolean) elapsed=0
[7/15/11 11:50:02:093 BRT] 00000030 PmiRmArmWrapp I PMRM00031:
parent:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=78,event=1
type=JDBC detail=javax.resource.spi.ManagedConnection.cleanup() elapsed=0
[7/15/11 11:50:02:109 BRT] 00000030 PmiRmArmWrapp I PMRM00031:
parent:ver=1,ip=9.44.168.198,time=1310740003343,pid=5576,reqid=21,event=1 -
current:ver=1,ip=9.44.168.198,time=1310740042828,pid=9588,reqid=2,event=1 type=URI
detail=/daytrader/scenario elapsed=7656

```

The timing data can be plotted manually. Graphing complex applications might be an easy way to visualize where time is being spent. Plotting data manually can take some time. Interaction diagrams can be substituted and the timing plotted on the diagrams.

In this example, the request takes 7656 milliseconds. Looking at all of the details in the log file, you can see exactly where any piece of total time is being spent. Of note in this configuration is that only the edge metrics are captured. The edge being the JDBC data and the original servlet URI request.

Whatever representation is preferred, the key understanding that is required is that request metrics provide a useful mechanisms for the analysis of where time is spent in requests that are being executed.

As more detail is added, the picture becomes more complete. Instead of a trace level of hops, you can use trace level debug to gather more details. Example 16-2 on page 584 shows metrics captured at a trace level of debug.

Example 16-2 Additional component types now recorded

```
[7/15/11 15:27:52:609 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=11,event=1
type=JDBC
detail=javax.resource.spi.ManagedConnectionFactory.createManagedConnection(Subject
, ConnectionRequestInfo) elapsed=250
[7/15/11 15:27:52:625 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=12,event=1
type=JDBC
detail=javax.resource.spi.ManagedConnectionFactory.matchManagedConnections(Set,
Subject, ConnectionRequestInfo) elapsed=0
[7/15/11 15:27:52:625 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=13,event=1
type=JDBC detail=javax.resource.spi.ManagedConnectionFactory.getConnection(Subject,
ConnectionRequestInfo) elapsed=0
[7/15/11 15:27:52:640 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=14,event=1
type=JDBC detail=javax.resource.spi.XAResource.start(Xid, int) elapsed=0
[7/15/11 15:27:52:859 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=15,event=1
type=JDBC detail=java.sql.Statement.executeQuery(String) elapsed=203
...
Lines deleted
...
[7/15/11 15:27:54:843 BRT] 00000030 PmiRmArmWrapp I PMRM0003I:
parent:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=4,event=1 -
current:ver=1,ip=9.44.168.198,time=1310754399656,pid=7108,reqid=56,event=1
type=JDBC detail=java.sql.PreparedStatement.executeQuery() elapsed=31
```

For more information about these techniques, see the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/tprf_requestmetrics.html

16.5 Monitoring operations

You can easily monitor the status of your environment. With the operational summaries, you can identify where applications, servers, or autonomic managers are running in your environment, the health of your environment, and if your environment is performing according to your service-level agreement.

For more information about configuring your cell to autonomically manage your environment, refer to 13.2, “Configuring dynamic operations” on page 472.

Intelligent Management monitoring provides the following functionalities:

- ▶ **Runtime operations overview**

With Intelligent Management, you can manage complex system operations with real time, meaningful visualization tools. Controlled implementation of autonomic capabilities helps you reduce your resource management cost.
- ▶ **Creating and managing reports**

With reports, you can view the performance of your virtualized environment. You can view statistics on availability, response time, traffic, and throughput.
- ▶ **Configuring the visualization data service**

The visualization data service, logs historic data in text files for reuse with other charting programs. Historic data is logged in comma-separated values with time stamps in standard long value from the java.util.Date class. By using the visualization data service, you can log historical data, calculate charge back values, or perform capacity planning.
- ▶ **Task management**

You can be notified about decisions that are made by autonomic managers. Notifications can represent either planned or unplanned events.
- ▶ **Troubleshooting extended administration**

Occasionally, when you use the extended manageability features, you might encounter a behavior that is not expected. For example, you can correct Web browser configurations so that your Web browser can support the visualization function.

16.5.1 Runtime operations overview

Operational alerts

Operational alerts inform you of the current state of your environment, including any issues, so that you can take action if necessary. For example, you might see an operational alert when a service policy is breached. For notification about issues that are continuing to occur over multiple intervals, runtime tasks are generated.

Reports

You can use customized charting to show if goals are being met. To benefit from this virtualized environment, you must know how your applications are performing. Dynamic charting provides a visual perspective of application performance. Specifically, statistics, such as availability, response time, traffic, and throughput are supported. A wide range of options from which you can create various charts is also provided.

You can organize your charts into chart groups to easily access charts from any reports view. You can also move charts to a new window to continue viewing the chart while performing other tasks in the administrative console. You can also move the chart back to the original chart group.

You can also view reports in the Reports tab of the following settings panels:

- ▶ Applications
- ▶ Dynamic clusters
- ▶ Clusters
- ▶ Servers
- ▶ Service policies

Note: Runtime operations reports use Scalable Vector Graphics (SVG) to display the data. Microsoft Internet Explorer does not have a SVG viewer installed by default. You can install a SVG plug-in or configure runtime operations to display JPEG images. To update the reports preferences, click **Runtime operations** → **Reports** → **Reports preferences**. Edit the Default chart format field.

The runtime operations reports are blank until you begin running an active load against your cell.

Dashboard

The dashboard displays a high-level summary of your overall environment. You can use the dashboard preferences to configure the information that displays in the dashboard, such as the default reports or the data from a specified chart group.

Runtime summaries

You can view the runtime information for the on demand routers, nodes, core groups, applications, deployment targets, service policies, and core components in your environment. Core components include a variety of autonomic controllers and managers. The runtime summaries include a list of instances for the particular resource type, status, stability and other information.

The charts that are included in the applications, deployment targets, and service policies summary views do not automatically refresh. To update the data, you can click the refresh icon on the chart. These charts display the average response time at a certain point in time. The default number of data sets that are displayed is five. You can, however, configure custom properties to specify that a unique number of data sets be displayed instead. For more information about runtime operations custom properties refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.wve.doc%2Fae%2Frwve_visrunopscustprop.html

If you are not running any work load requests through the on demand router (ODR) to your applications, no data is displayed in the chart. Also, if you are charting service policy-related metrics, no goal lines for the data set are displayed in the key. After the ODR sends the work load requests, the chart is automatically updated and the statistics are displayed. However, the key table is not automatically updated. Refresh the panel to view the goal lines that are added to the data set.

To access the runtime summaries, click **Runtime operations** in the administrative console.

Operations tabs

The operations tabs show the overall status of the selected item in the following settings panels:

- ▶ Applications
- ▶ Dynamic clusters
- ▶ Clusters
- ▶ Servers
- ▶ Core groups

16.5.2 Creating and managing reports

With reports, you can view the performance of your virtualized environment. You can view statistics about availability, response time, traffic, throughput.

Reports are charts that show runtime data. You can use this data to monitor your environment, and take correctional actions when necessary.

You can view reports that display live data, or you can use charts displaying historic data logged over a period of days, weeks, months, years. To use historic charting, configure and enable the visualization data service. For more information about configuring the visualization data service refer to 16.5.3, “Configuring the visualization data service” on page 588.

For historic charting, use the sliding bar to adjust the percentage of time to view. You can focus on a specific time interval and include more data in the chart.

Note: To observe historic data over a long period of time, increase the maximum heap size of the deployment manager. The default maximum heap size is 256 MB. To increase the performance and to prevent heap dumps, go to **System administration** → **Deployment manager** → **Java and process management** → **Process definition** → **Java Virtual Machine**, and set the maximum heap size to a higher value, for example 2048 MB.

Creating and managing reports:

1. Open the Reports tab in the administrative console. You can access the Reports tab from any of the following paths:
 - **Runtime operations** → **Reports**
 - **Servers** → **All servers** → **<server_name>** → **Reports**

Note: The Reports tab is not available for on demand router (ODR) servers.

- **Servers** → **Clusters** → **Dynamic clusters** → **<your_dynamic_cluster_name>** → **Reports**
 - **Servers** → **WebSphere Application server clusters** → **<your_cluster_name>** → **Reports**
 - **Applications** → **All application** → **<your_application_name>** → **Reports**
 - **Operational policies** → **Service policies** → **<service_policy_name>** → **Reports**
2. Act on operational alerts. Operational alerts display the status of resources in your environment. The name of the specific resource is highlighted. Go to the configuration panel or view the chart for that resource to take corrective actions.
 3. Configure report and chart preferences. Expand Reports preferences. The preferences are global and apply to any new charts that you create. However, the preference change does not apply to existing charts. To change the preferences for an existing chart, click **Preferences** on the chart.
 4. Add a chart. Click the **Open a new chart** tab. A new tab opens with a blank chart. Click **Add data** to specify the data set and metrics that you want to monitor. If you accessed the Reports tab through the **Runtime operations** → **Reports** panel, you can specify the scope of the chart. Click **Change scope**.
 5. Create, access, and remove chart groups. Chart groups are global and can be accessed from any Reports tab:
 - **Create:** If you have a specific set of charts in the tabs that you want to save, type a name, and save the current group of chart tabs configuration as a chart group by clicking **Save**.
 - **Access:** To access the chart group from any reports panel later, go to Saved chart groups, and click the name of the chart group.

- You can also open a chart in a new window, so that you can continue doing other tasks in the administrative console while still monitoring your chart. Click **View chart** in the new window. To return the chart from the new window to the chart group, click **View chart** in chart group.
- Remove: To remove a chart group, select the chart group from Saved chart groups, and click **Remove chart group**.

16.5.3 Configuring the visualization data service

The visualization data service logs historic data in text files for reuse with other charting programs. Historic data is logged in comma-separated values with time stamps in standard long value from the `java.util.Date` class. By using the visualization data service, you can log historical data, calculate charge back values, or perform capacity planning.

Note: You must use the deployment manager to implement this feature. Ensure that if you are using multiple core groups, they are correctly bridged.

If you are a user with either a monitor or an operator administrative role, you can only view the visualization data service information. If you have a configurator administrative role, you can change the configuration. If you have an administrator role, you have all of the privileges for the visualization data service.

Configuring the visualization data service:

1. In the administrative console, click **System administration** → **Visualization Data Service**.
2. Enter a value in the Time stamp format field. The time stamp format specifies a time and date pattern that is used when logging the visualization data. Use the `SimpleDateFormat` Java class to format your time stamp.
3. In the Maximum file size field, type a whole integer for the maximum file size for logs.
4. In the Maximum number of historical files field, type a whole integer for the maximum number of logs to be generated per historic cache type.
5. In the File name field, type the path where the log files are generated. You can use a variable in the file name value, for example: `${LOG_ROOT}/visualization`.
6. In the Data log write interval field, type a whole number between 1 and 365 for the interval in which the logs are generated in seconds, minutes, hours, or days. If you plan to log data for several metrics over a period longer than 1 week, increase the Data log write interval for better performance.
7. From the Data transformer action list, select **AVERAGE** or **SKIP** to specify how to transform data when the interval reaches its maximum value. More data points are provided than you might want to use. The **AVERAGE** option averages the existing data points between the specified interval. The **SKIP** option skips the data points to only use the points specifically at the intervals.
8. Select **Enable log** to start logging historic data.
9. If logging was enabled before you configured the visualization data service, restart your deployment manager.

16.5.4 Task management

You can be notified about decisions that are made by autonomic managers. Notifications can represent either planned or unplanned events.

Planned events

Planned events are events for which the runtime environment has an action plan. For example, you might have a health policy defined that has an average response time that is breaching its configured limit, which can trigger an increased footprint of a dynamic cluster. If the product is operating in automatic mode, the action plan runs, and you can view a notification of the action that was taken. In supervised mode, you can view and approve the action plan. The interaction modes can vary in your configuration. A mix of automatic and supervised mode activities by dynamic clusters and health policies can exist.

Unplanned events

If an event occurs and it is not assigned to an action plan, you can be notified that something unexpected has happened. After you receive the notification, develop a plan to correct the situation, if it is a valid issue.

Runtime tasks

A runtime task is generated when an event occurs. Runtime tasks provide information from which you can accept, deny, or close the recommended action plan. Tasks that have taken action or expired tasks stay in the runtime task list for 24 hours by default. You can change this default by setting a cell level custom property. For more information about how to set the custom property, refer to 16.5.5, “Managing runtime tasks” on page 589.

Runtime tasks reside in the deployment manager memory and are logged in `<was_root>/profiles/<Dmgr_profile_name>/tmsStorage` as well. When the deployment manager is restarted, the deployment manager reads the runtime task entries in the `tmsStorage` log.

Event logging

You can enable logging for events. For more information about task management service event logger, refer to “Event logging” on page 589.

Notifications

You can have email notifications sent to specified users when runtime tasks occur. For more information about defining email notifications, refer to “Defining email notification” on page 590.

16.5.5 Managing runtime tasks

A runtime task is generated by a runtime component within the product. Runtime tasks provide information from which you can accept, deny, or close the recommended action plan.

Use runtime tasks to view and manage the recommendations that the autonomic controllers provide to improve the health and performance of your environment. If you are running in supervised mode, you must accept, deny, or close each runtime task. In automatic mode, the autonomic controllers automatically take these actions.

A task is lost when it is sent from a node when the deployment manager is not running. The same holds true in a high-availability deployment manager environment. Even though the down time is minimal between the time when an active highly-available deployment manager

shuts down and when the standby deployment manager becomes active, tasks are still lost. After a failover of a highly-available deployment manager occurs, the executing tasks at the original active deployment manager are changed to the unknown state.

Managing runtime tasks:

1. From the administrative console, click **System administration** → **Task management** → **Runtime tasks**.
2. Click the **task ID** link to view the task target objects and corresponding monitors of a specific task. A task target object is a server, cluster, service policy, node, health policy, or application. One of these objects might be linked to configuration, performance, or log monitor views in the administrative console.

With a configuration monitor, the placement controller might recommend that an instance of the TestCluster dynamic cluster start on the test4 node. The recommendation states that this action is required for the dynamic cluster to meet its configured minimum number of running instances. You can click the link to the configuration panel for the TestCluster dynamic cluster to view its settings and verify or change the minimum number of running cluster instances.

3. For a performance monitor, you can click a link to a chart with data that is specified in the monitor and specific to the target object. For a log monitor, click the Runtime tab of the target object Java virtual machine logs to investigate the displayed logs.
4. To view the action plan for the task, click the task ID. To act on a specific task, select the corresponding task, and from the action list select Accept, Deny, or Close. Accepting the task commits the previewed action plan. Closing the task means that the task is successfully handled by the task management service or is manually closed. Denying the task places the task in inactive status if the task is not submitted again in the next batch of task submissions by the originating component. You can also select multiple tasks with the same actions. After you act on a task, the action list is unavailable for that task.
5. Click **Submit**.

Defining email notification

In addition to actively monitoring tasks on the Runtime tasks panel, you can have email notifications sent to specified users when runtime tasks occur.

By defining email notification, you can specify a set of email addresses to be notified when runtime tasks are generated.

The sender user ID is preset to wasxd. Register the sender user ID in your Simple Mail Transfer Protocol (SMTP) registry before the cell can successfully send email notifications.

Defining an email notification:

1. Optional: Set the user ID from which the emails are sent. By default, the value is preset to wasxd. To change this default:
 - a. In the administrative console, click **System administration** → **Cell** → **Custom properties** → **New**.
 - b. Enter the name of the custom property as `task.email.global.sender.id`, and set the value to the specific email user ID that you want to use.
 - c. Click **Apply**. The change becomes effective when you save the configuration.
2. Configure email notification in the administrative console. Select **System administration** → **Task management** → **Notifications**. Select **Enable notifications**. When runtime tasks are generated, an email notification is sent to each of the specified email addresses.

3. Verify that email is sent for a particular task. When all of the changes are made, click **Test email** to verify that the email is sent for a particular task. If the email server uses spam-blocking software, this test can fail and prevent the email from being displayed in the test inbox. To save your changes, click **Apply** or **OK**.

When notification is enabled, an email is sent to each of the specified addresses when a task is generated. When notification is enabled and no email addresses are specified, no emails are sent.

16.6 IBM Tivoli Composite Application Manager for WebSphere Application Server

IBM Tivoli Composite Application Manager (ITCAM) for WebSphere is available for WebSphere Application V8.5. After being installed, it is embedded in the application server and can be configured to monitor application performance, providing real-time status information in WebSphere console. More information about ITCAM for WebSphere Application Server is at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/ecam.html

ITCAM for WebSphere can easily be integrated with ITCAM for Application Diagnostics to provide deep dive diagnostics data, real-time monitoring, and management. ITCAM for Application Diagnostics requires additional licensing. More information is at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/planning_an_installation/overview_of_itcam_for_application_diagnostics.html

16.6.1 Installing the data collector

The first step in enabling ITCAM is to install and configure a data collector for the monitored servers. ITCAM for WebSphere can be installed from WebSphere media through IBM Installation Manager. For information about the installation, refer to Chapter 2, “Getting started with ITCAM for WebSphere Application Server”, in the *ITCAM for WebSphere Application Server 7.2* manual, at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfad.doc_7101/itcam_ecam_installation_72_85.pdf

16.6.2 Configuring Tivoli Composite Application Manager for WebSphere metrics

The last step of the installation process is to run the configuration tool that configures the servers for data collection. The TPV settings for the servers can also be adjusted to include ITCAM data during the configuration.

To complete the configuration steps, refer to Chapters 3 and 4 in the *ITCAM for WebSphere Application Server 7.2* manual, at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfad.doc_7101/itcam_ecam_installation_72_85.pdf

After the data collector configuration is complete, access the deployment manager's administrative console, and navigate to the Performance Monitoring Infrastructure window for the configured server (Figure 16-26). A new ITCAM for WebSphere Application Server link is now in the Additional Properties section of the window.

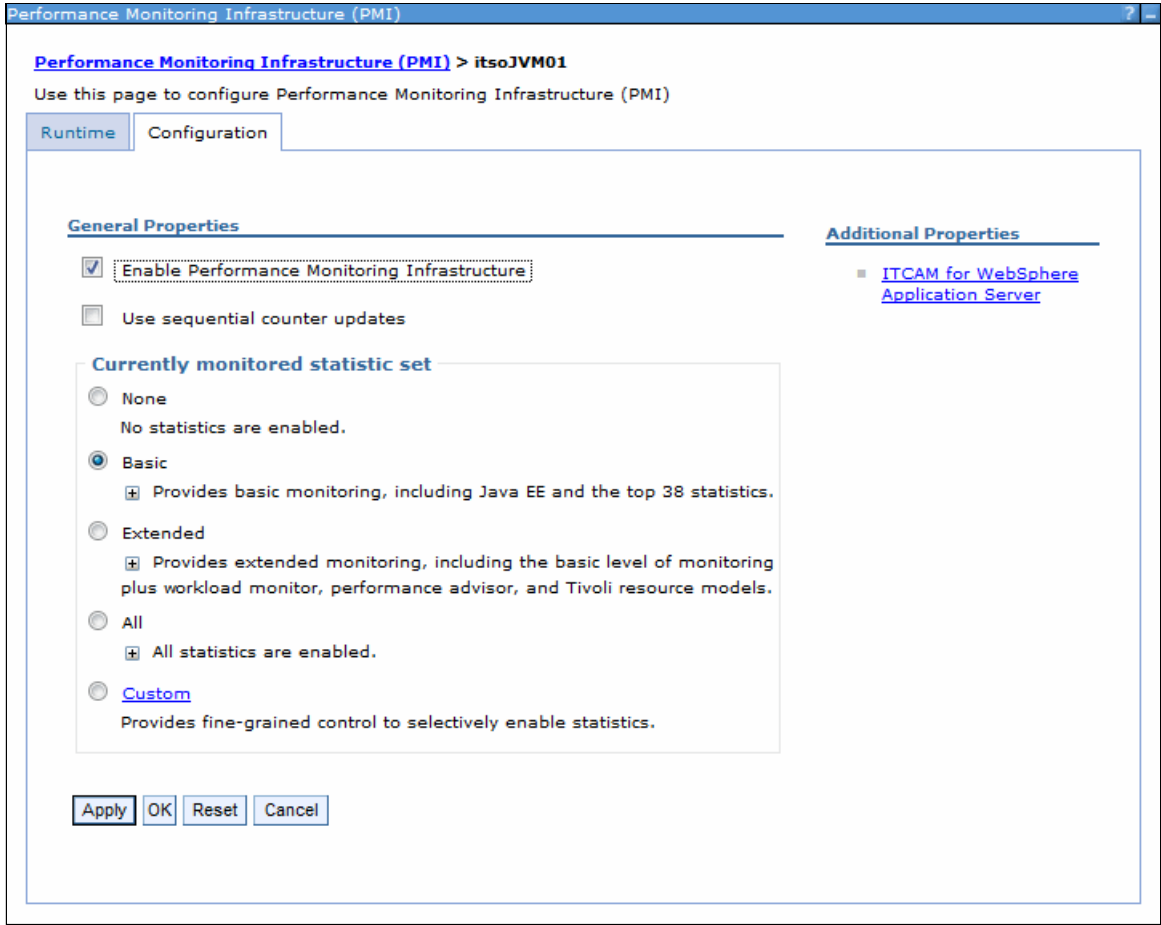


Figure 16-26 PMI configuration with ITCAM for WebSphere Application Server

Click the **ITCAM for WebSphere Application Server** link to open the ITCAM for WebSphere Application Server window (Figure 16-27 on page 593). This window contains the setting that enables the data collector.



Figure 16-27 ITCAM for WebSphere Application Server window

Note: This setting is only enabled if you choose to apply the required PMI settings, as mentioned in 16.2.1, “PMI defaults and monitoring settings” on page 555. Otherwise, enable the ITCAM for WebSphere Application Server data collector, and restart the server.

The PMI settings must also be configured at the custom level.

16.6.3 Viewing IBM Tivoli Composite Application Manager for WebSphere data

This section assumes that the data collector is installed and the ITCAM for WebSphere Application Server data collector is enabled:

1. Navigate to the Performance Monitoring Infrastructure window for the server, and select the **Custom** statistic set.
2. Click the **ITCAM for WebSphere Application Server** link, and navigate to open the configuration window.
3. Click the **Runtime** tab.
4. Click **Start Monitoring**.

Notes: You must start ITCAM for WebSphere Application Server monitoring on a server at each server restart. Because this action was taken in the **Runtime** tab, it is not preserved in the configuration.

After monitoring starts, the Stop Monitoring button is displayed instead. If you decide later to stop the ITCAM for WebSphere Application Server monitoring, return to this window, and click that button.

5. Click the server link (its0JVM01) in the breadcrumb at the top of the window to return to the PMI configuration window (Figure 16-26 on page 592).
6. Click the **Runtime** tab.
7. Click the **Custom** link to navigate into the custom monitoring window. Click **ITCAM Application Performance** to show the ITCAM counters (Figure 16-28 on page 594).

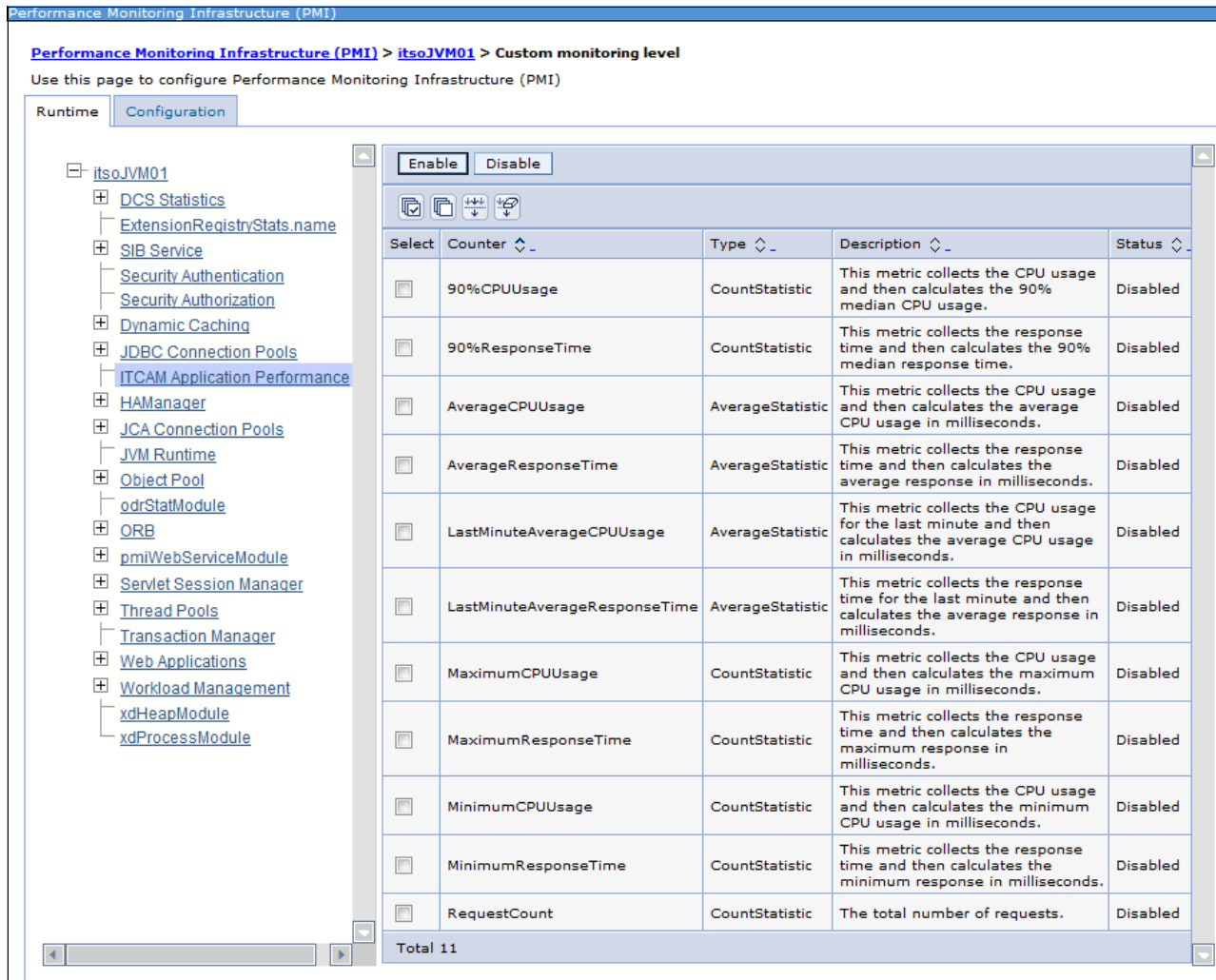


Figure 16-28 ITCAM Application performance counters

8. Choose which ITCAM for WebSphere Application Server counters are needed. The counter window provides additional description information and the current status of the counter.
9. Choose the counters that are desired, and click **Enable**.

Note: Some counters can only be activated by the system. This situation means that they will stay in disabled mode, even if you try to enable them.

10. After enabling the counters, navigate to the Current Activity Tivoli Performance Viewer window for the server that is being monitored.
11. Open the Performance Modules tree view and then select the **ITCAM Application Performance** menu item.
12. Click **View Module(s)** to view the data (Figure 16-29 on page 595).

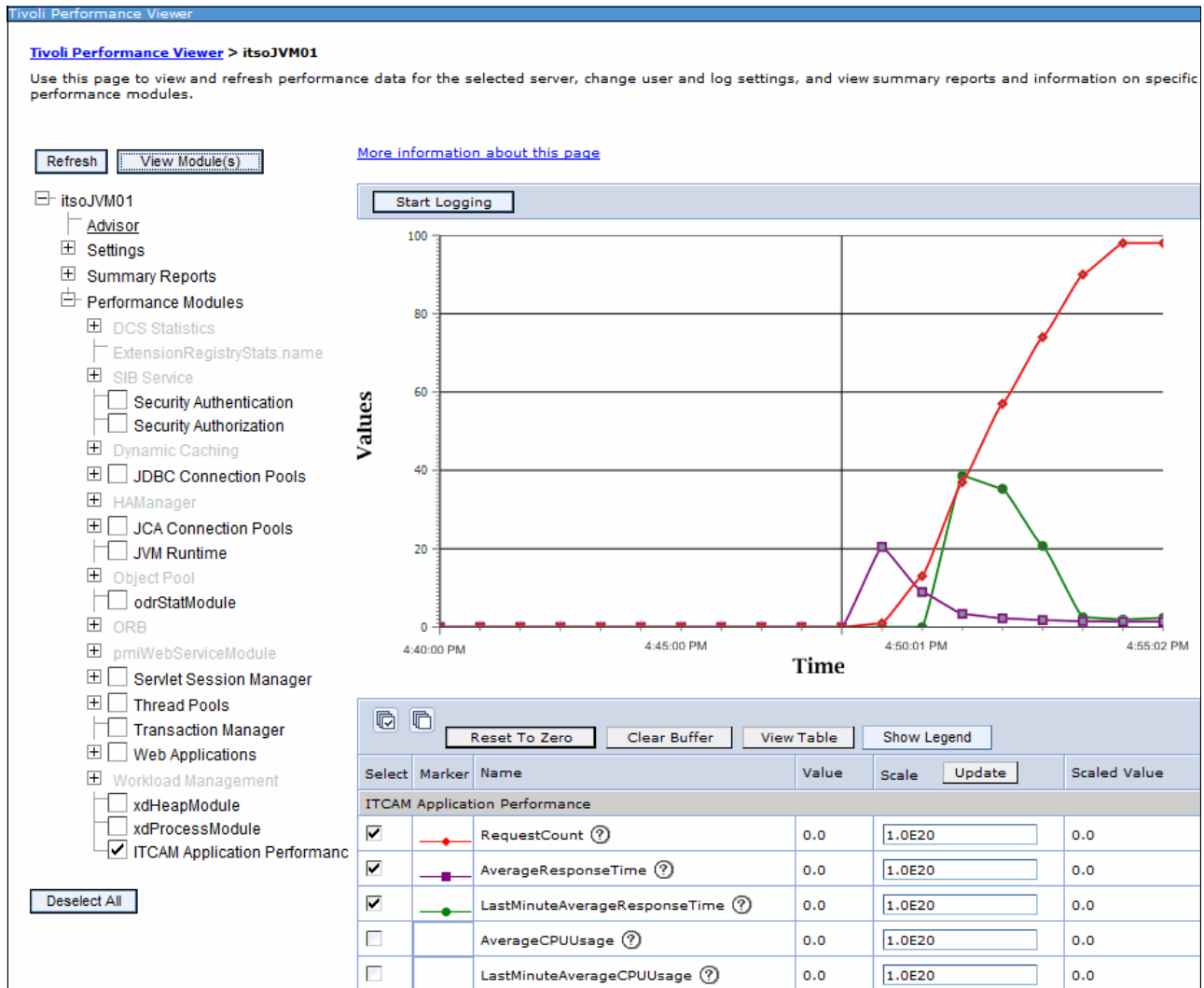


Figure 16-29 ITCAM counters graph

16.7 Additional resources for monitoring

In this section, we cover additional useful resources to be used while monitoring your WebSphere environment.

16.7.1 Java dump and core files

Java core files are like a picture of what occurs inside the Java virtual machine. When it is used, this kind of file is helpful to analyze scenarios where the CPU is reaching 100% percent of utilization, when threads are hanging or where the performance is slow.

Java dump and system dumps files are a picture of the objects that were in Java virtual machine memory and are helpful in diagnosing memory related problems such as memory leaks.

Both kinds of files can be generated by the WebSphere console by completing the following steps:

1. Click **Troubleshooting** → **Java dumps and cores**.
2. Select the desired server.
3. Click one of the available buttons:
 - **Heap dump**
 - **Java core**
 - **System dump**
4. Look for the files in the profile directory.

For more information about javadumps, refer to the Java Information Center at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fjavadump.html>

For more information about heapdumps, refer to the Java information Center at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fheapdump.html>

For more information about system dumps, refer to the Java information Center at the following website:

http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fdump_viewer.html

16.7.2 Basic logging

WebSphere Application Server provides a great many logging options, and the most significant environment incidents are logged automatically. The basic logging provides Java virtual machine logs, diagnostic trace, and service log files commonly named `SystemOut.log`, `SystemErr.log`, `trace.log`, and `activity.log`. To support the analysis of logging activity, the IBM Support Assistant can be used to launch the IBM Tivoli Log Analyzer. This tool supports the downloading of symptom catalogs of known issues that go into the logs. Furthermore, application developers can build symptom catalogs and write logs from their applications to add to the manageability of their applications.

16.7.3 Advanced logging

In WebSphere V8.5, an alternative to the basic log and trace facility called High Performance Extensible Logging (HPEL) is offered. It provides three repositories, as shown in Figure 16-30 on page 597:

- ▶ **Log data repository:** A storage facility for log records, typically information stored in `SystemOut.log`, `SystemErr.log`, or `java.util.logging` at level detail or higher.
- ▶ **Trace data repository:** A storage facility for trace records, typically information written to `java.util.logging` below level detail.
- ▶ **Text log:** A plain text file for log and trace records, which is provided for convenience.

Note: All the data that is written to log and trace repositories are parsed and formatted to be stored in a text log file. For this reason, consider disabling the log file as soon as possible to enhance server performance.

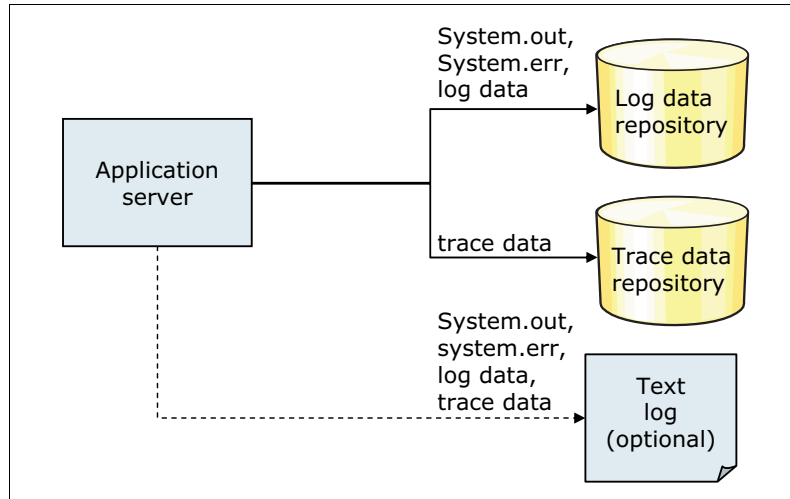


Figure 16-30 HPEL repositories

Now the data is stored in proprietary binary format instead of text as in basic logging (the only exception is for the text log repository). In this way, the following benefits are realized:

- ▶ There is no more text parsing.
- ▶ More data is available because truncation is not necessary.
- ▶ Data is not formatted unless it is needed.
- ▶ No need to clear log files before server start, for example, to diagnosis a problem.
- ▶ Trace speed is improved and more data can be available, and it has half of the impact that basic tracing has.
- ▶ Provides a common solution between z/OS and distributed platform.
- ▶ Applications running with HPEL run faster than running the same application with basic logging.

To activate HPEL logging and tracing, in the administrative console, click **Troubleshooting** → **Logs and trace** → **<your_server>** → **Switch to HPEL Mode**.

Note: A server restart is needed after enabling HPEL logging and tracing.

To read the log and trace records in this new format, WebSphere V8.5 provides tools to analyze log and trace data in HPEL format. They are:

- ▶ The **logViewer** command-line tool

More information about the logViewer command line tool is at:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.base.doc%2Fae%2Ftrb_logviewer.html

- ▶ The administrative console log viewer
To access the administrative console log viewer:
 - a. You must first enable HPEL login for your application server.
 - b. On the administrative console, navigate to **Troubleshooting** → **Logs and Trace** → **your_application_server**.
 - c. Click **View HPEL logs and trace**.

More information about the administrative console log viewer is at:

http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/topic/com.ibm.iea.was_v8/was/8.0/ProblemDetermination/LogViewerWebUI/logviewerwebui_viewlet_swf.html?dmuid=20100720095532322577

More information about HPEL is in the Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/c_trb_HPELOverview.html

16.7.4 Operating system monitoring

Most operating systems have tools for monitoring memory, CPU utilization, and disk I/O. They must be monitored and controlled.

16.7.5 Summary of monitoring tips

The following summary lists a simple set of best practices about how to establish a useful monitoring environment:

- ▶ Take time to understand the applications that are being deployed into the environment. Use this knowledge to plan for the kinds of metrics that will be beneficial in understanding application performance.
- ▶ Activate monitoring at the planned level in all testing environments, especially when benchmarking. Although monitoring is vital to your ability to understand an environment, it has a processing impact, and uses CPU, memory, and other resources. Monitoring can impact capacity planning.
- ▶ Use monitoring to understand normal operations and gain an appreciation of what is normal for your systems.
- ▶ Check for differences in your systems after all release changes, especially if full performance testing and benchmarks have not been re-established.
- ▶ Use the basic metrics of PMI as a good starting set of metrics, and then customize them to your needs.
- ▶ Monitoring WebSphere Application Server alone is not enough. Application server monitoring needs to be part of an overall monitoring strategy.



Part 4

Managing z/OS systems



Performance tuning

Performance tuning is a complex task that spans multiple components and areas with the goal of improving system performance. It is heavily dependent on the hardware, software, and external factors that are employed in your enterprise. Performance numbers advertised in this chapter are based on measurements and projections using standard IBM benchmarks in a controlled environment. Therefore, the performance gains for your system can differ. Using a standard performance methodology and extensive profiling is advisable to understand an application's bottlenecks in production load scenarios.

WebSphere Application Server V8.5 delivers a highly scalable and highly-available platform for applications. It provides multiple tuning options to optimize runtime performance.

This chapter focuses on the z/OS system and suggests methods by which you can improve performance through a combination of product features and application development considerations. It includes the following topics:

- ▶ Introduction to WebSphere Application Server for z/OS V8.5 performance
- ▶ External factors and z/OS specifics
- ▶ Performance tuning templates
- ▶ 64-bit considerations
- ▶ JVM tuning
- ▶ Connection pool tuning
- ▶ Runtime provisioning
- ▶ Pass by reference
- ▶ Logging and tracing
- ▶ Tuning workload management on z/OS systems
- ▶ Fast response cache accelerator and caching
- ▶ Using WebSphere for z/OS Optimized Local Adapters
- ▶ IBM HTTP Server Status monitoring page
- ▶ Tools

17.1 Introduction to WebSphere Application Server for z/OS V8.5 performance

WebSphere Application Server for z/OS V8.5 has version to version improvements of around 20% compared to WebSphere Application server V7. Some of these improvements are described in this chapter.

End-to-end performance enhancements include:

- ▶ Up to 15% for DayTrader 2.0 running WebSphere V8.5 in comparison to the V7
- ▶ An additional 43% for DayTrader 2.0 running on a z196 machine in comparison to an IBM System z10® machine
- ▶ Up to 24% for SOA Benchmark running WebSphere V8.5 in comparison to V7
- ▶ An additional 40% for SOA Benchmark running on an IBM System z196 machine in comparison to a z10 machine

These numbers are informational only.

WebSphere Application Server for z/OS V8.5 also includes Java Virtual Machine J9 2.6 and the following sets of native DLL libraries inside the delivered Java run time:

- ▶ 31-bit version
- ▶ 64-bit version
- ▶ 64-bit version optimized for the z196 architecture

The Java specification is still V6 because there is no change in the application layer interfaces, but it is enhanced to V6.0.1. WebSphere Application Server for z/OS V8.5 also includes a new garbage collection mechanism and an optimized just-in-time (JIT) compiler.

WebSphere Application Server for z/OS V8.5 is hardware-platform aware and introduces z196 hardware usage into the J9 JVM to use new instructions and paradigms, such as out-of-order-execution pipeline and InfiniBand infrastructure for I/O bus or Parallel Sysplex coupling.

This release focuses on the following areas:

- ▶ Optimized runtime performance for programming models
- ▶ Memory optimization and improvement of server startup time
- ▶ Improved installation time, application deployment, and configuration functions
- ▶ Improved runtime performance as a base for stack products
- ▶ Small, simple, and fast development environment

For more detailed information about performance improvements with WebSphere Application Server for z/OS V8.5, go to the following website:

[http://www-03.ibm.com/support/techdocs/atmastr.nsf/5cb5ed706d254a8186256c71006d2e0a/981aad32cbab471886257604001e1ec8/\\$FILE/WP101532%20-%20Why%20WebSphere%20Application%20Server%20for%20zOS%20-%20Notes.pdf](http://www-03.ibm.com/support/techdocs/atmastr.nsf/5cb5ed706d254a8186256c71006d2e0a/981aad32cbab471886257604001e1ec8/$FILE/WP101532%20-%20Why%20WebSphere%20Application%20Server%20for%20zOS%20-%20Notes.pdf)

17.2 External factors and z/OS specifics

External factors, such as collocation and hardware configuration, can play an important role in performance-tuning scenarios. In this section, we discuss the more common external

factors that can affect performance and provide suggestions for ways to obtain the best performance possible.

17.2.1 Getting the most benefit from collocation

Collocation of the data layer and application layer can be beneficial during application processing on the same system or in the same sysplex. Collocation provides the following benefits:

- ▶ Takes advantage of cross-memory data transfer, reduces overall request latency, improves overall throughput, and reduces overall CPU utilization through the elimination of network traffic handling
- ▶ Allows for the assertion of security identity, maintains the same thread of execution, and manages within a single Workload Manager (WLM) environment

17.2.2 Addressing hardware configuration

Hardware is also influential to overall system performance. Sources of performance boosts include CPU, disks with parallel access volumes support, network cards and switch speed, processor cache, system memory, and machine model. In this section, we list important configuration settings to consider.

Additional resource: For information about hardware configuration settings, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.installation.zseries.doc%2Fae%2Ftins_preparez.html

17.2.3 z/OS tuning tips

In this section, we provide information about z/OS tuning tips.

ZFS

Using cache-enabled and sysplex-aware z/OS Distributed File Service zSeries file system (zFS) is preferred over the hierarchical file system (HFS). HFS generally provides better response times for all types of operations.

You can verify the cache and average response time per operation type using the following command:

```
F zfsproc,QUERY,ALL
```

Additional resource: For more information about ZFS tuning, refer to *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580.

Shared library and UNIX System Services

Use the SHRLIBRGNSIZE OMVS parameter carefully with 31-bit servers because it limits the usable region size system-wide. Review OMVS options to understand the limits that your environment imposes and current usage that it represents. WebSphere Application Server for z/OS provides a way to disable the shared libraries by implementing the standard WebSphere environment variable `_BPXK_DISABLE_SHLIB`. The application server prints out the SHRLIBRGNSIZE value as part of the message BB000341I during startup.

Example 17-1 BBOO0341I message printout with shared library size of 16 MB

```
BB000341I VARIOUS RESOURCE MONITORING DATA:  
(64):(16777216):():():():():():():():():()
```

Try to eliminate any unnecessary STEPLIBs, and review your PATH statement to prioritize frequently referenced programs. Consider caching your high activity read-only files.

Additional tips: For more UNIX System Services tips, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Ftprf_tuningonzos.html

Region size

Verify that there are no limiting factors to region size, such as IEFUSI exit or insufficient MEMLIMIT on your system, when preparing to use large heaps. Also verify that sufficient memory is available so that you do not incur a paging penalty.

SMF

Consider turning off collection of SMF type 92 file system related records, which can represent a significant impact for any UNIX-based process operating with a multitude of files. WebSphere Application Server for z/OS creates SMF record type 120. Consider enabling server interval SMF records and container interval SMF records rather than server activity records and container activity records. Also, using only SMF type 120 subtype 9 records that provide a unified request-based view can prove beneficial because this configuration represents less than a 1% impact. Verify that SMF data sets are allocated optimally.

RACF

Use RACLIST for the appropriate Resource Access Control Facility (RACF) classes, and use crypto cards where advantageous. Do not activate classes you do not use. Use the virtual lookaside facility (VLF) to cache user identifiers (UIDs) and group identifiers (GIDs) in the COFVLFxx parmlib member.

RACF tuning: For more specific information about RACF tuning, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fw1c6topsecuring_tun.html

TCP/IP

Consider setting the NODELAYACKS parameter either on WebSphere Application Server ports or globally on the TCPCONFIG statement. This option specifies that an acknowledgment is returned immediately when data is received that has the PUSH bit set in the TCP header. This setting can improve throughput by as much as 50%. Increase the TCPSENBFRSIZE and TCPRCVBFRSIZE parameters to at least 64 KB. Optimize your DNS configuration so that lookups for frequently-used servers and clients are being cached.

TCP/IP tuning tips: For more TCP/IP tuning tips, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftprf_tunetcpip.html

RRS

Resource Recovery Services (RRS) are crucial to sync point processing. Use coupling facility (CF) log streams or high performing DASD for DASD-only log streams that are allocated with large control interval (CI) sizes for RRS log streams.

Sample RRS log stream definition jobs are provided in the `/WAS_product_image_path/util/zos/JCL/` directory. To off load the entire directory to a partitioned data set, use the `zCopyFilesToPds.sh` script provided in the `/WAS_product_image_path/bin/` directory, as shown in Example 17-2.

Example 17-2 Using the zCopyFilesToPds.sh script to off load directory content

```
/usr/lpp/zWebSphere/V85R0/bin/zCopyFilesToPds.sh -sourceDirectory  
/WAS_product_image_path/util/zos/JCL/ -targetPDS 'YOUR.DATA.SET' -debug
```

For more information about RRS specific tuning, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Ftprf_tunezrrs.html

Language Environment

For best performance, ensure that the CEELPA IBM Language Environment® load library is in your link pack area (LPA) concatenation by specifying it in your active LPALSTxx member.

Do not enable the Language Environment HEAPCHK option unless absolutely necessary because the performance cost associated with heap diagnostic testing is high.

Language Environment tuning tips: For Language Environment for z/OS-specific tuning tips, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftprf_tunezle.html

For more information about tuning your Language Environment heap, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftprf_tunezleheap.html

17.3 Performance tuning templates

WebSphere Application Server V7 introduced three tuning templates as suggested starting points to improve application server performance. WebSphere Application Server for z/OS V8.5 contains the latest version of these tuning templates:

- ▶ Peak applies tuning parameters that are well suited for performance test and proof of concept (POC) environments, where peak runtime performance is desired.
- ▶ Development tunes the application server for a development environment where frequent application updates are performed and system resources are typically constrained.

- Default (Standard) returns the server configuration to the standard defaults.

These templates provide a multitude of the latest preferred practices and tuning recommendations. Review and test the templates for application impact before using them on any option environments.

Note: Some parameters that are tuned by these scripts (most notably the ORB Pass-By-Reference setting) can impact the functionality of applications.

You can apply these templates by selecting the **Server runtime performance tuning setting** option when you create a WebSphere Application Server V8.5 profile (using the profile management tool for z/OS), as shown in Figure 17-1.

Figure 17-1 Applying profile template during WebSphere Application Server creation in zPMT

You can also apply a template manually by running the ApplyPerfTuning Jython script, as shown in Example 17-3.

Example 17-3 Manually invoking the ApplyPerfTuning Jython script

```

${WAS_INSTALL_ROOT}/bin/wsadmin.sh -lang jython -f
/WAS_product_image_path/scriptLibraries/perfTuning/V70/ApplyPerfTuning.py
-nodeName nodename -serverName servername -templateFile template.name

```

Applying your own templates: If you develop your own performance templates based on the provided templates, be aware that customized templates might not be portable with other levels or versions of WebSphere Application Server. The underlying structure of the configuration XML files is prone to change.

For more information about tuning profiles and the values that they introduce, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Ftprf_tuneappserv_script.html

17.4 64-bit considerations

This section provides a comprehensive overview of 64-bit mode and its impact on the application server's performance. The 64-bit mode was available in WebSphere Application Server V6. Since WebSphere Application Server V7, the installation defaulted to 64-bit mode because 31-bit mode was deprecated. Consider reconfiguring application servers that were migrated from previous releases to 64-bit mode.

17.4.1 Enabling 64-bit mode

Ensure that z/OS page set allocations are sufficient and that JCL and Automation procedures do not specify the AMODE=31 parameter on the **start** command before you switch to 64-bit mode. If the detected mode differs from the AMODE setting, an error message of BB000336E is produced, and the server fails to start.

Refer to the following website for further information:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Fcrun_z64bit.html

To change to 64-bit mode using the administrative console:

1. Click **Servers** → **Server types** → **WebSphere application servers** → **servername**.
2. On the Configuration tab, select **Run in 64 bit JVM mode**, as shown in Figure 17-2.

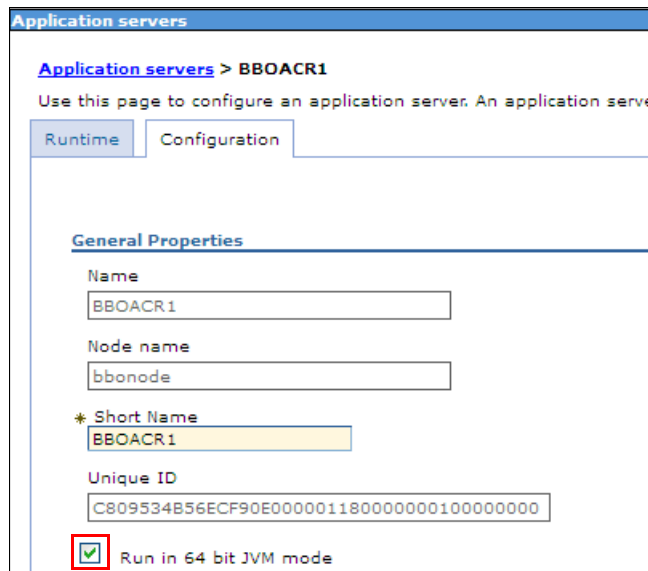


Figure 17-2 Switch to 64-bit mode

3. Click **Apply**.
4. For a Network Deployment configuration, click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or select the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
5. Click **Restart** for the target server in the **Servers** → **Server types** → **WebSphere Application servers** view.

Tip: The administrative console uses the following command to facilitate the bit mode change to 64-bit addressing inside the XML configuration files:

```
AdminTask.setJVMMode('[-nodeName nodename -serverName servername -mode bitmode]')
```

For application development purposes, JVM provides the `com.ibm.vm.bitmode` programmatic API, to determine the bit mode setting in which an application server is running.

17.4.2 Effects of switching to 64-bit mode

Running WebSphere Application Server in 64-bit mode allows for heap relief. It grants WebSphere Application Server the ability to run heaps larger than 1 GB and provides access for up to 16 EB of virtual memory. Alternatively, the cost of using 64-bit object references can enlarge your heap by as much as 40%. The inherently bigger objects also affect data locality, and thus contribute to higher translation look-aside buffer (TLB) and data cache miss rates. These higher rates slow dynamic address translation (DAT) and affect application performance.

The compressed references and large page support features can provide relief for reduced throughput and memory footprint growth incurred when migrating from 31-bit JVM to 64-bit JVM. With these two JVM properties turned on, 64-bit environments can match and outperform previous 31-bit environments. These features are not turned on by default because they have software and hardware requirements.

Compressed references

Compressed references is a method for managing object pointers with the JVM. Some workloads in 64-bit environments have shown objects increased by up to 45% because the object header and object references doubles in width. Compressed references mode is usable for JVM heaps of up to 30 GB on z/OS. It reduces the size of the 64-bit object pointer to 4 bytes (word).

In compressed references mode, object header referenced data (such as class-related data and thread data) are allocated below the 2 GB bar. This allocation enables all the references to be 32-bit and no padding to occur in the object header, as illustrated in Figure 17-3.

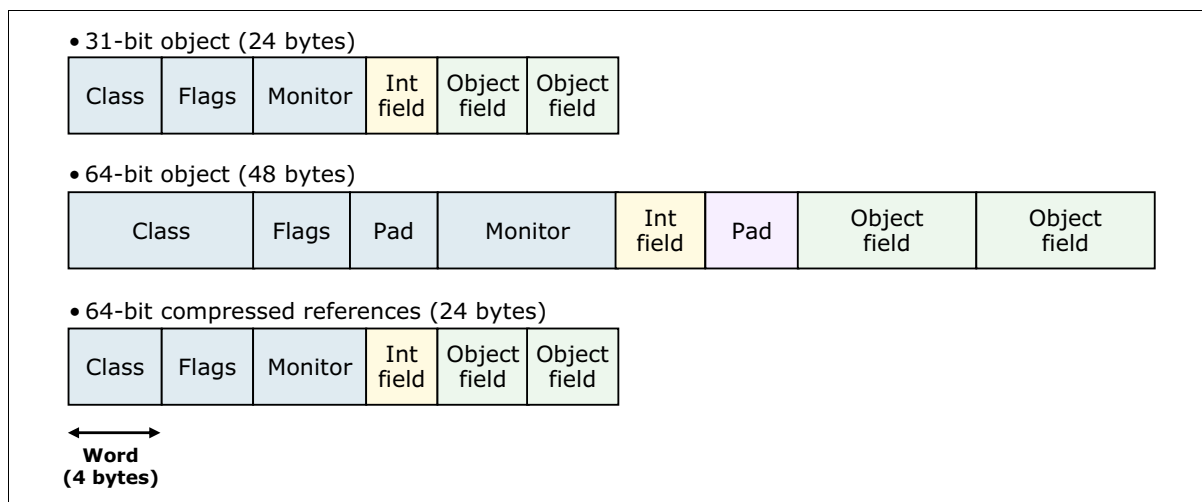


Figure 17-3 Header and object references in 31-bit and 64-bit mode

For object data references, we use a paradigm where 64-bit heap allocations are aligned on double-word boundaries, leaving the lowest three bits at zero. This allows for a right-shift compression for object references. Adequate compression is chosen based on the top of Java heap specified by the `-Xmx` maximum Java heap option value, as listed in Table 17-1.

Table 17-1 Compressed references algorithm choice based on maximum heap size

| Max heap size specified by the <code>-Xmx</code> option | Top of the heap located | Shift amount used for object reference compression |
|---|-------------------------|--|
| 2 GB or less | below 2^{32} | 0 |
| 6 GB or less | below 2^{33} | 1 |
| 14 GB or less | below 2^{34} | 2 |
| 30 GB or less | below 2^{35} | 3 |
| Greater than 30 GB | above 2^{35} | Only supported without compressed references |

For further information about compressed references mode, go to the following website:

<http://public.dhe.ibm.com/partnerworld/pub/whitepaper/1d71a.pdf>

To change to compressed references mode for the JVM:

1. Click **Environment** → **WebSphere variables**.
 You can also enable this variable on a process basis by clicking **Servers** → **Server Types** → **WebSphere application servers** → *server_name* → **Java and process management** → **ProcessDefinition** → *servant* → **Environment entries**.
2. Select a scope based on your desired affected environment.
3. Click **New**, and specify `IBM_JAVA_OPTIONS` in the name field.
4. Add or append the `-Xcompressedrefs` value, as shown in Figure 17-4 on page 610.
5. Click **Apply**.

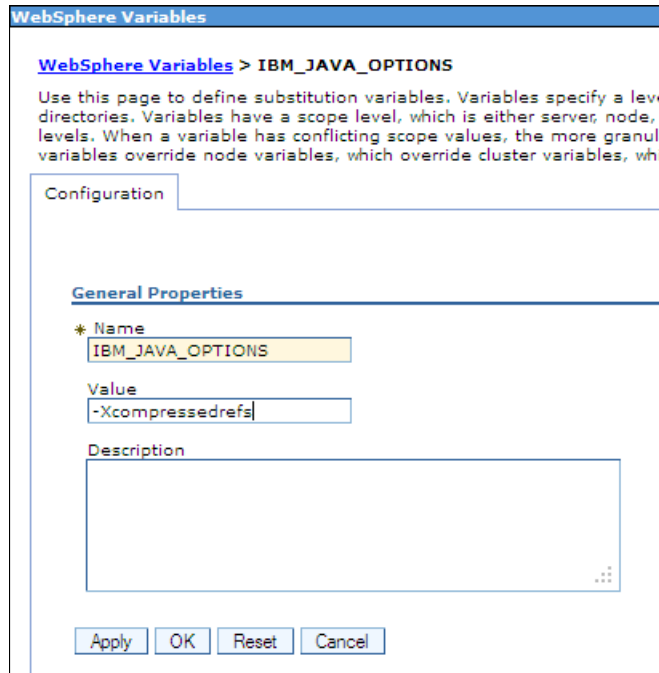


Figure 17-4 Enabling compressed references

6. For a Network Deployment configuration, click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or select the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
7. Click **Restart** for the affected server in the **Servers** → **Server types** → **WebSphere Application servers** view.

Note: Be aware that specifying `-Xcompressedrefs` as a generic JVM argument (on IBM System z9® or earlier) or for heaps over 30 GB produces an error and JVM fails to start.

Use the `F servername, JAVACORE` command or click **Troubleshooting** → **Java dumps and cores** to produce the dump file with a list of JVM arguments or VerboseGC output, as shown in Example 17-4, to confirm the usage of the feature.

Example 17-4 VerboseGC information for compressed references

```
<initialized id="1" timestamp="2011-07-09T12:56:32.600">
  <attribute name="gcPolicy" value="-Xgcpolicy:gencon" />
  <attribute name="maxHeapSize" value="0xc0000000" />
  <attribute name="initialHeapSize" value="0xc0000000" />
  <attribute name="compressedRefs" value="true" />
  <attribute name="compressedRefsDisplacement" value="0x0" />
  <attribute name="compressedRefsShift" value="0x1" />
<vmargs>

  <vmarg name="-Xcompressedrefs" />
```

Note: When running compressed references, a different JVM initializes; therefore, when analyzing problems, you need to enable compressed references with the dump extractor to analyze dumps that are produced by the JVM.

For more information about enabling the feature, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcontainer_tune_jvm.html

Requirements: The compressed references feature has the following requirements:

- ▶ IBM z/OS V1R8 or later
- ▶ IBM Developer Kit for Java 6, 64-bit edition, SR4 or later (APAR PK82091)
- ▶ IBM z/OS support APAR OA26294

Large page support

With the evolution of 64-bit environments, applications produce a higher virtual footprint and use freely the pointers to 2⁶⁴ virtual storage map. The mapping of real or absolute real storage ranges to virtual one is done by using hardware dynamic address translation (DAT) structures.

To avoid DAT hardware access to main storage tables for every request, the translation look-aside buffer (TLB) was introduced. TLBs are fast array memory within each processing unit. In cases where TLB hits, translation processing is much faster. If a TLB miss occurs, calculating the virtual-to-real mapping of a page can take several dozen cycles. Even with 512 TLB entries per processing unit in z10, EC model it is not enough to generate high TLB hit ratio in today's virtual storage intensive applications. Therefore, z10 mainframes now support large pages with enhanced-DAT architecture. One large page can hold 256 times more virtual memory than a 4 KB page, guaranteeing fewer TLB misses and TLB entries to represent the data footprint of the application.

Large pages are allocated above the 2 GB virtual bar. They are 1 MB in size fixed (non-swappable) pages and backed up by real storage. As such, they are not available to 31-bit applications, and application executable code cannot reside in them. The amount of storage for large pages is defined by the LFAREA IEASYSxx system parameter and can be up to 80% of real memory. You must perform an IPL to introduce this parameter.

If the system is constrained for 4 KB pages and the large pages are still available, it will convert the available large pages into 4 KB pages. The system can also swap 4 KB pages if more large pages are needed. When allocated, however, large pages cannot be converted into 4 KB pages because they are fixed.

Java applications that allocate lots of Java objects and cause frequent garbage collection will typically benefit from large pages. The data access pattern can also affect the benefits of large pages.

For more information, refer to the following website:

<http://public.dhe.ibm.com/partnerworld/pub/whitepaper/1d71a.pdf>

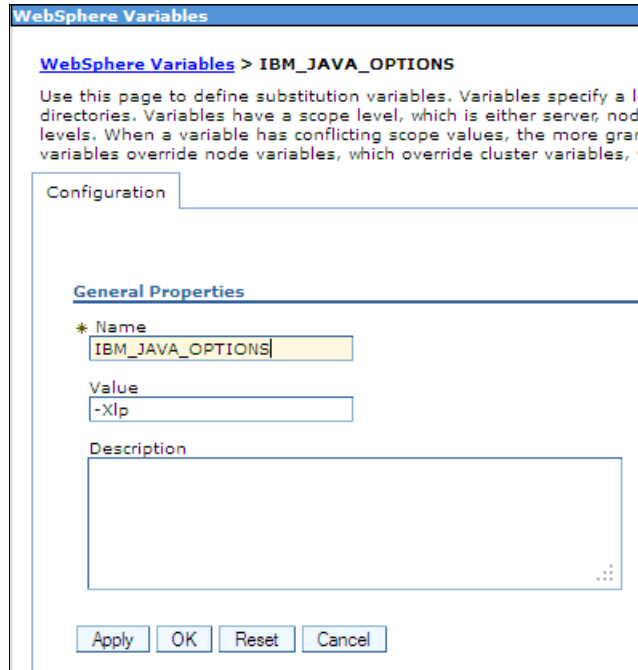
To enable this feature, complete the following steps:

1. Click **Environment** → **WebSphere variables**.

You can also enable this variable on a by process basis by clicking **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Java and process**

management → **ProcessDefinition** → **servant** → **Environment entries** or by specifying the **-Xlp** option on the generic JVM argument for the process.

2. Select a scope based on your desired affected environment.
3. Click **New**, and specify **IBM_JAVA_OPTIONS** in the name field.
4. Add or append the **-Xlp** option on the Value field, as shown in Figure 17-5.
5. Click **Apply**.



The screenshot shows the 'WebSphere Variables' configuration page for the variable 'IBM_JAVA_OPTIONS'. The 'Name' field is set to 'IBM_JAVA_OPTIONS' and the 'Value' field is set to '-Xlp'. The 'Description' field is empty. The 'Apply' button is highlighted.

Figure 17-5 Enabling large pages support

6. For a Network Deployment configuration, click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or select the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
7. Click **Restart** for the target server in the **Servers** → **Server types** → **WebSphere Application servers** view.

Note: If large page support is not enabled or if there are not enough large pages to satisfy the allocation request during JVM initialization, it falls back to the default usage of 4 KB pages.

Use the **F *servername*, JAVACORE** command, or click **Troubleshooting** → **Java dumps and cores** to produce the dump file with a list of JVM arguments. Alternatively, use **VerboseGC** output, as shown in Example 17-5, to confirm the usage of the feature.

Example 17-5 VerboseGC information for large pages

```
<initialized id="1" timestamp="2011-07-09T12:56:32.600">
  <attribute name="gcPolicy" value="-Xgcpolicy:gencon" />
  <attribute name="pageSize" value="0x1000" />
  <attribute name="requestedPageSize" value="0x100000" />
  <attribute name="gcthreads" value="2" />
</vmargs>
```

<vmarg name="-X1p" />

Requirements: This feature requires the following components:

- ▶ IBM z/OS V1R9 or later
- ▶ IBM Developer Kit for Java6, 64-bit edition, SR2 or later (APAR PK82091)
- ▶ IBM System z® hardware using the IBM System z10 processors or later
- ▶ IBM z/OS support APAR OA20902 (only needed for IBM z/OS V1R9)
- ▶ IBM z/OS support APAR OA25485

17.5 JVM tuning

This section provides basic tuning tips for JVM performance.

17.5.1 Default garbage collection

WebSphere Application Server for z/OS V8.5 changes the default policy from using the **optthruput** option to using the **gencon** option, which was introduced with IBM Java 5.

With the **optthruput** option, all objects were allocated from one large contiguous heap that was shared by all threads. When a garbage collection was invoked, the entire heap was scanned, and unreferenced objects were marked and swapped. This option was best for applications that required optimal throughput but introduced longer garbage collection pause times.

The **gencon** option is a Generational Concurrent Garbage Collector and reduces these longer pause times by dividing the heap into two sections, the *nursery* and *tenured* areas. New objects are always allocated in the nursery part of the heap. When objects age (have been scanned multiple times by garbage collection), they are moved to the tenured area. The JVM constantly monitors the size of both the nursery and tenured areas and adjusts the size of each based on garbage collection frequency and pause times.

This philosophy is based on the observation that most objects are short term. Thus, by allocating them in separate heaps, scanning becomes more efficient because none of the longer-lived objects are scanned. This type of garbage collection policy is ideal for transactional workloads or heavily cached workloads because it reduces the impact of garbage collection by referencing these long-lived objects infrequently.

17.5.2 General JVM suggestions

This section provides general JVM suggestions.

JVM heap

Verify your current and actively used heap size using the following command:

```
F servername,DISPLAY,JVMHEAP
```

Example 17-6 shows the output of this command.

Example 17-6 Output of the DISPLAY,JVMHEAP command

```
BB000201I JVM HEAP INFORMATION FOR SERVER BB0C003/BB0ACR1/STC05742
BB000202I (STC05742) HEAP(NURSERY), COUNT(00000026), FREE 000
```

```
STORAGE(00000000044457E0), TOTAL STORAGE(0000000005410000)
BBO00202I (STC05742) HEAP(MATURE), COUNT(00000000), FREE 001
STORAGE(00000000090987D8), TOTAL STORAGE(000000000C000000)
BBO00204I JVM HEAP INFORMATION FOR SERVER BBOC003/BBOACR1/STC05742
COMPLETE
```

Do not increase the heap size if paging is occurring for your application server. You can use the administrative console or generic JVM arguments to set the minimum and maximum heap sizes:

- ▶ The following JVM argument sets the initial Java heap size:
-Xms<size>
- ▶ The following JVM argument sets the maximum Java heap size:
-Xmx<size>

To set the minimum and maximum heap sizes using the Administrative console:

1. Click **Servers** → **Server Types**, and click **WebSphere application servers**, as shown in Figure 17-6.

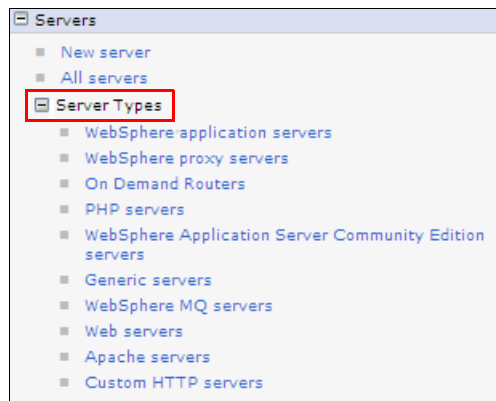


Figure 17-6 WebSphere application servers in Servers expandable

2. In the Application servers view, select your desired application server, as shown in Figure 17-7.

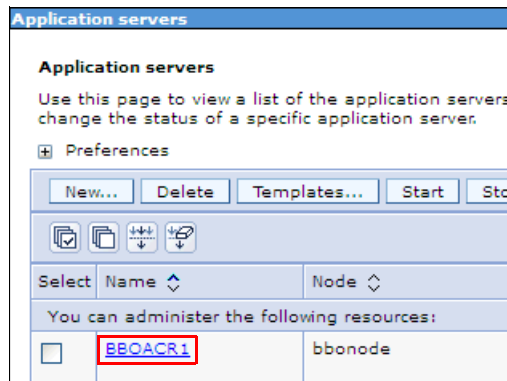


Figure 17-7 WebSphere Application Server view server selection

3. Navigate to Server Infrastructure, and expand the Java and Process Management option. Select **Process definition**, as shown in Figure 17-8 on page 615.



Figure 17-8 Specific application servers configuration window

4. Select the desired process type. In this demonstration, we chose the Servant process, as shown in Figure 17-9.

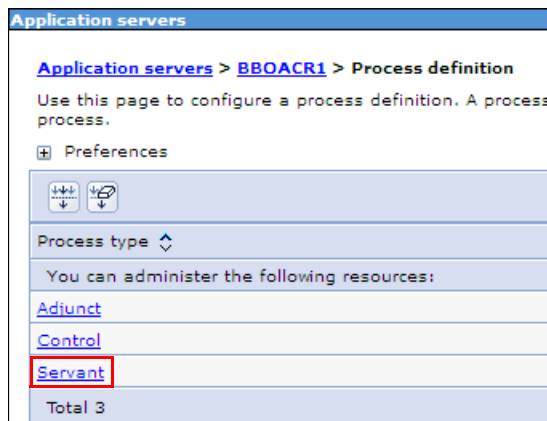


Figure 17-9 Servant process inside the Process definition view

5. On the Additional Properties window, click **Java Virtual Machine**, as shown in Figure 17-10.

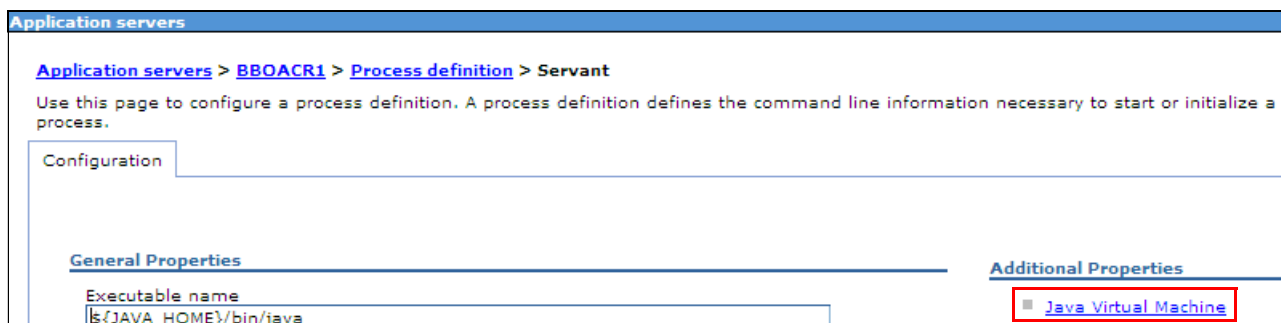


Figure 17-10 Java Virtual Machine option on the Servant process view

- Specify an initial heap size and a maximum heap size for the servant process JVM, as shown in Figure 17-11.

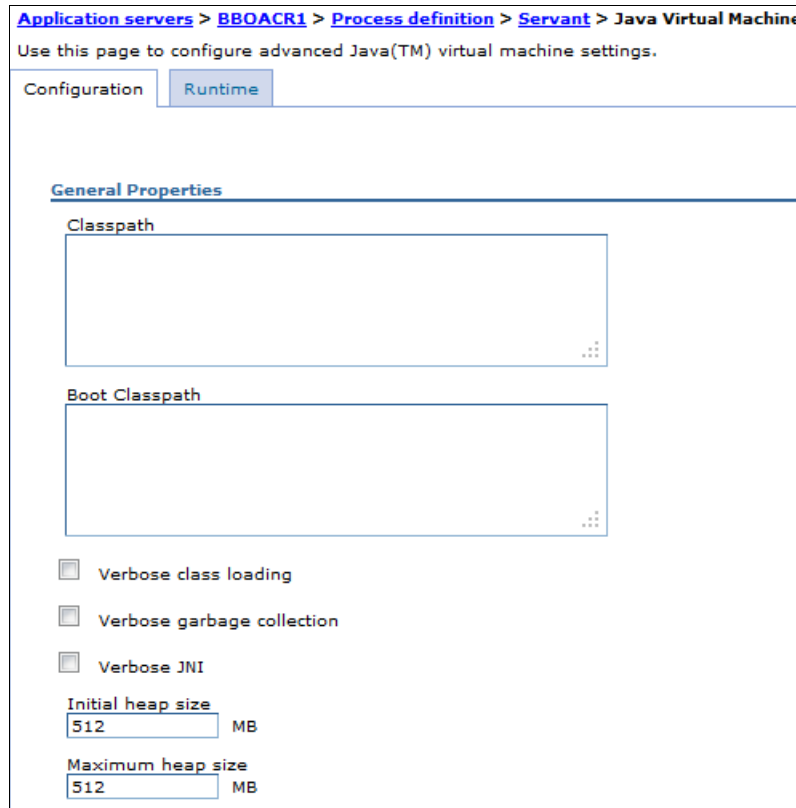


Figure 17-11 Java Virtual Machine view of the servant process

Specifying the same initial and maximum heap size can avoid compaction and expansion phase of the garbage collection and, in turn, benefit performance. However, using the same size can also have a negative impact on the heap fragmentation and can increase garbage collection cycles compared to using a lower initial heap size. Test your application storage behavior and analyze garbage collection traces to find the optimum settings for your application server environment. Garbage collection can run no more than 5% of the total time.

Preferred practice: For a **gencon** garbage collection policy with JVM heap over 1 GB, consider changing the default values on the **IBM_JAVA_OPTIONS** environment variable, as follows:

- ▶ **-XX:NewSize=640m**
- ▶ **-XX:MaxNewSize=640m**
- ▶ **-XX:SurvivorRatio=16**

The balanced policy is designed for large heap sizes and can be useful when you use the **gencon** policy with a heap size greater than 4 GB or if you use large arrays.

For more information about balanced garbage collection policy, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.java.doc.60_26%2Fvm626%2FJ9%2FGC%2Fmm_gc_balanced.html

Application behavior

Verify the behavior of your application in relation to the Java memory usage to ensure that your application meets the following suggestions:

- ▶ Adheres to preferred programming practices
- ▶ Does not overutilize objects
- ▶ Does not create memory leaks

JVM arguments

During Java configuration, verify that no unnecessary classes are present in the class path. At the same time, make sure that no class is missing because the class search can be I/O intensive. The classes that are referenced most frequently are located near the front of the path.

You can use several general JVM properties to speed up the server start time at a cost of lower runtime performance or disabling functions. You can use this approach for development environments where start speed is preferred over runtime optimization. However, carefully review the following options and their consequences:

- ▶ **-Xquickstart**
JVM uses a lower optimization level for class method compiles.
- ▶ **-Xverify:none**
JVM skips the class verification stage during class loading. Corrupted or invalid class data is not detected.
- ▶ **-Xgcthreads<number of processors>**
Garbage collection uses several threads during collection.
- ▶ **-Xnocompactgc**
Disables the heap compaction garbage collection operation.

For more information about JVM arguments, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Frun_rconfproc_jvm.html

Dump processing

To limit dump processing, export these environment variables on the WebSphere Application Server profile:

```
JAVA_DUMP_OPTS=ONANYSIGNAL(JAVADUMP[3],SYSDUMP[1]),ONINTERRUPT(NONE)
JAVA_DUMP_TDUMP_PATTERN=DUMP.D%y%m%d.T%H%M%S.%job
```

Additional tips: For more information about JVM-specific tuning, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcontainer_tune_jvm.html

17.6 Connection pool tuning

For more information about connection pool tuning, refer to 14.2.2, “Data source tuning” on page 497.

17.7 Runtime provisioning

Runtime provisioning is a feature introduced in WebSphere Application Server V7. It focuses on improving application server startup time by providing intelligent analysis of application set and server configuration to determine the needed subset of components to be loaded. This feature does not load an entire runtime library and thus decreases the memory footprint and shortens the start time.

There is no need for administrators and application developers to modify any processes to take advantage of the runtime provisioning. To turn this feature on, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**, as shown in Figure 17-12.



Figure 17-12 WebSphere application servers window

2. In the **WebSphere application servers** view, select your desired application server, as shown in Figure 17-13.

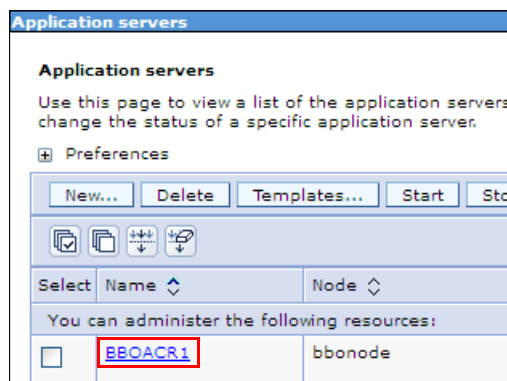


Figure 17-13 WebSphere application server view server selection

3. Select **Start component as needed** to enable runtime provisioning in this server, as shown in Figure 17-14.

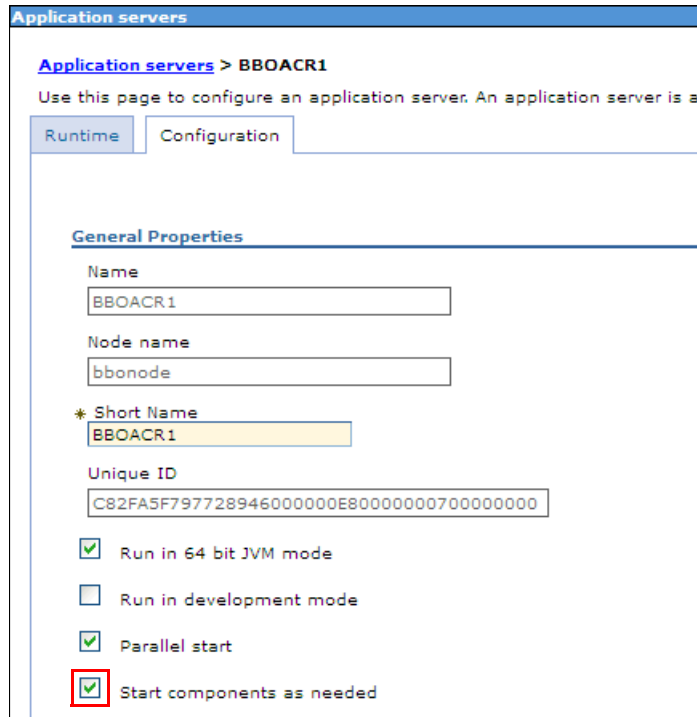


Figure 17-14 Application servers configuration window

WebSphere Application Server V8.5 provides a way to disable WebSphere MQ functionality inside the application server.

For more information about this feature, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftmm_ep.html

17.8 Pass by reference

Refer to 14.4.2, “The pass by reference parameter” on page 512 for information about this ORB service option.

To learn how to enable Pass message payload by reference for JMS, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Fcjn_passbyref_steps.html

To learn how to use PassByReference optimization in SCA applications, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftsca_passby_ref.html

17.9 Logging and tracing

This section focuses on the logging and tracing options that are specific to the z/OS system and the new standardized model of logging that is introduced in WebSphere Application Server V8.5.

17.9.1 High Performance Extensible Logging overview

High Performance Extensible Logging (HPEL) is a complete solution for standardized logging and tracing in WebSphere Application Server V8.5. It is an alternative to the existing log and trace facilities that are offered on the z/OS platform, which use LogStreams, Component Trace, Job Entry Subsystem, Hierarchical File System, or other facilities. It is a new strategic logging solution delivered across platforms. It does not include native traces.

HPEL uses mechanisms with data repositories where logging and tracing data is stored all in one place in binary format. To improve performance, data is not shared across instances and is not converted by LogViewer to plain text unless needed. HPEL provides an optional text log function for debugging convenience, where log or trace data can be written to text log file in plain text format immediately. You can use the administrative console or the LogViewer commands to view the HPEL collected logs and traces.

HPEL out performs existing log and trace facilities and can have performance benefits for log-intensive applications; however, HPEL is not enabled by default. An HPEL API is provided in the `com.ibm.websphere.logging.hpel` package to simplify development of tools for log and trace processing.

For more information, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrb_usinghpel.html

17.9.2 Enabling HPEL mode

To enable HPEL logging, refer to 19.4.5, “Advanced logging” on page 725. You must disable text logging after enabling HPEL to improve performance.

17.9.3 z/OS logging and tracing tips

For WebSphere Application Server for z/OS in basic logging mode, the SystemOut.log, trace.log, and STDOUT streams are redirected to the SYSPRINT ddname by default. The System.err, and STDERR streams are redirected to the SYSOUT ddname. WebSphere Application Server for z/OS cataloged procedures associate these ddnames with print (SYSOUT=*) data sets, causing message logs to go into WebSphere Application Server job output.

In basic mode, ensure that your logging and tracing level is not unnecessarily high by completing the following steps:

1. Navigate to **Troubleshooting** → **Logs and trace**, as shown in Figure 17-15.

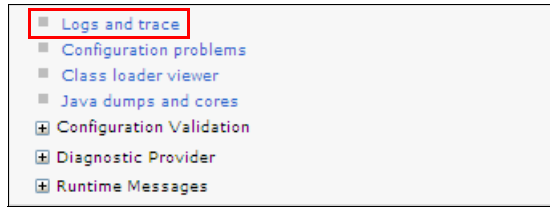


Figure 17-15 WebSphere Application Server Logs and trace view

2. Click the target application server name, as shown in Figure 17-16.

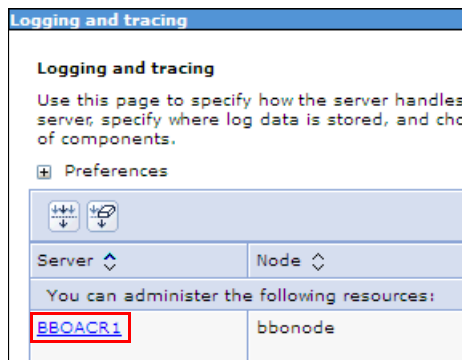


Figure 17-16 Application server choice on Logging and tracing view

3. Click **Change log detail levels**, as shown in Figure 17-5.

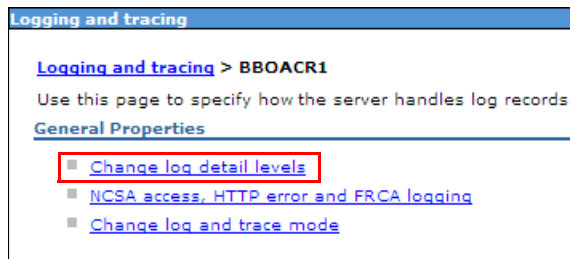


Figure 17-17 Change log detail levels on Application server Logging and tracing view

4. Verify your logging and trace detail level. The default logging level is *=info, as shown in Figure 17-18 on page 622.

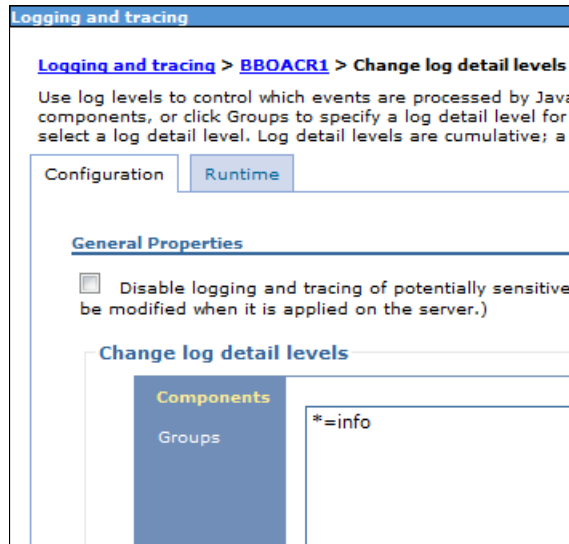


Figure 17-18 Configuration view of the change log detail level view

Preferred practice: Start the trace string from the most broad component groups and then select more specific traces. The advantage to this approach is that the trace settings for classes or packages that are contained in a larger group are specified correctly by including them later in the trace string.

The logging string is processed from left to right. During the save processing optimization, part of the logging string might be modified or removed if the levels they configure are overridden by another part of the logging string.

For more information about trace controls:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fwe1c_ref_trb_file.html

In addition to the administrative console, you can also use WebSphere Application Server V8.5 modify commands to change the tracing levels dynamically:

- ▶ **F *servername*, TRACEALL={0,1,2,3}**
Establishes a general trace level for the server.
- ▶ **F *servername*, TRACEINIT**
Resets the trace settings to the initial trace settings.
- ▶ **F *servername*, TRACENONE**
Turns off all trace settings.
- ▶ **F *servername*, TRACETOSYSPRINT={YES|NO}**
Selects whether to send the trace to SYSPRINT.
- ▶ **F *servername*, TRACEBASIC=*x* and F *servername*, TRACEDETAIL=*x***
Sets the trace level, where *x* is a value from Table 17-2 on page 623 that represents a component.

Note: Subcomponents, specified by numbers, receive detailed traces. Other parts of the product receive tracing as specified on the TRACEALL parameter.

Table 17-2 TRACEBASIC and TRACEDetail components values

| Value | Product component |
|-------|-------------------|
| 0 | RAS |
| 1 | Common utilities |
| 3 | COMM |
| 4 | ORB |
| 6 | OTS |
| 7 | Shasta |
| 9 | z/OS Wrappers |
| A | Daemon |
| E | Security |
| F | Externalization |
| J | JRAs |
| L | Java EE |

On z/OS, you can use CTRACE as your tracing option. You can use the CTRACE facilities in WebSphere Application Server for z/OS to manage the collection and storage of trace data. CTRACE data is written to address space buffers in private (pageable) storage, which can be formatted using an interactive problem control system (IPCS) if a dump file of the address space is created.

CTRACE data can also be written to trace data sets on disk or tape using an external writer procedure. This procedure uses the BBOCTIxx parmlib member and BBOWTR procedure with samples provided in the `/WAS_product_image_path/util/zos/JCL/` directory. Because CTRACE uses system resources efficiently, you can collect valuable trace data with minimal impact on performance. Ensure that CTRACE for all components is configured to MIN or OFF when not used.

For information about how to set up CTRACE support, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fttrb_trcover.html

You can display the CTRACE status using the following command. Example 17-7 shows the output of the command:

```
D TRACE,COMP=cell_short_name
```

Example 17-7 D TRACE,COMP=cell_short_name command output

```
-D TRACE,COMP=BBOCELL
 IEE843I 00.50.47 TRACE DISPLAY 792
      SYSTEM STATUS INFORMATION
ST=(ON,0001M,00002M) AS=ON BR=OFF EX=ON MO=OFF MT=(ON,024K)
  COMPONENT      MODE BUFFER HEAD SUBS
-----
  BBOCELL        ON          HEAD  11
  ASIDS          *NOT SUPPORTED*
```

| | |
|----------|-----------------|
| JOBNAMES | *NOT SUPPORTED* |
| OPTIONS | MINIMUM |
| WRITER | *NONE* |

As in previous releases, you can set up an error log using the coupling facility or using a DASD-only log stream for a single WebSphere Application Server or shared error log for several servers. If you decide to use the feature, use a coupling facility log stream that is shareable across the sysplex.

You can view the error log records using the log browse utility (BBORBLOG) located in the */WAS_product_image_path/util/zos/EXEC/* directory.

For more information about using BBORBLOG EXEC, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ffrprf_tuneztrace.html

If you are using Transaction XA partner logs or SIP recovery log streams, use coupling facility log streams. For more information, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.installation.zseries.doc%2Fae%2Ffcins_logstrm.html

For information about JDBC tracing, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Ffrtrb_jdbccomp.html

For internal tracing tips, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Ffrprf_tuneztrace.html

17.10 Tuning workload management on z/OS systems

This section discusses the tailoring needs and benefits of workload management on z/OS systems in conjunction with WebSphere Application Server.

17.10.1 The concept of workload management on z/OS systems

Workload management consists of categorizing, prioritizing, routing, and reporting on requests. On z/OS systems, the Workload Manager (WLM), which is an operating system component, allows the system programmer to configure the rules that are used to differentiate and organize units of work. When classified, WLM can determine the service level agreement (SLA) for this type of work. WLM then assigns system resources to units of work from that workload, making a best attempt to allow all work to meet the specified goals. Less important work is sacrificed to meet the goals of more important work if necessary.

WebSphere Application Server for z/OS uses this facility for workload classification by default. It uses a controller-servant region mechanism and can run several instances of servant regions for a single WebSphere Application Server, called a *dynamic servant region expansion*. WLM manages these servant instances in a dynamic application environment. The instances are started as dictated by workload within the guidelines of the WLM MIN/MAX SERVANT parameters. WLM then routes work to the appropriate servant based on the classification rules. It also uses sophisticated algorithms to choose the best candidate if multiple servants are bound for the same service class.

All application work that runs inside WebSphere Application Server runs under WLM-managed enclaves. Proper WLM goals can significantly affect application throughput. To allow servants to start in parallel, use the `wlm_servant_start_parallel` custom property.

For more complex scenarios and to understand the overall workload goals in your organization, consult with your WLM systems programmer.

For more information about WebSphere Application Server and the z/OS WLM, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Ftrun_wlm.html

17.10.2 Classification rules

The work priority of location service daemons and controllers must be higher than the priority for servants. Work inside controllers and daemons is categorized based on started task control (STC) classification rules, where a servant's classification is more complex.

The WebSphere Application Server's servant lifecycle with WLM includes the following phases:

- ▶ The start phase before the servant is bound to a service class queue
- ▶ The initialized phase after the servant is bound to a service class queue

During the first phase, the servant is guided by the STC classification rules, which are most likely of the velocity type. During the second phase, application work runs in enclaves that are guided by CBtype classification rules. Specify achievable response time goals with a percentile here. You can also classify work using the collection name of the cluster. WLM performs better with less service classes.

Velocity™ goals for application work are not meaningful and must be avoided.

If running in multi-instanced servant environment, use WLM classification and define unique service classes for different priority work that is running in the same server. Also be sure to allow for at least one servant per unique service class.

Using non-enclave threads: z/OS V1.12 introduced a parameter to bind non-enclave threads to the goals of the STC service class for the life of the servant. This IEA0PTxx parm1ib member parameter is called MANAGENONENCLAVEWORK. It applies to supporting tasks (such as garbage collection threads) inside the servant. It can be set dynamically, and WLM policy is refreshed as part of the SET OPT=xx command.

For more information about controlling WLM dynamic application environment, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Fcontainer_data_concepts.html

For WebSphere Application Server work request management, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Fcrun_wlmzos.html

For more information about WLM tuning tips for z/OS, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fwelc_ref_adm_consider.html

You must also add a classification rule for OMVS type of work for BPXBATCH to prevent elongation of the start process and execution of the applyPTF.sh script. For more information, go to the following website:

<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD102730>

17.10.3 Classification XML

To help WLM identify the work that is running inside the servant and to provide finer classification rules for applications, WebSphere Application Server uses workload classification XML. It is introduced by specifying a path to the XML for the `wlm_classification_file` WebSphere variable, as shown in Figure 17-19.

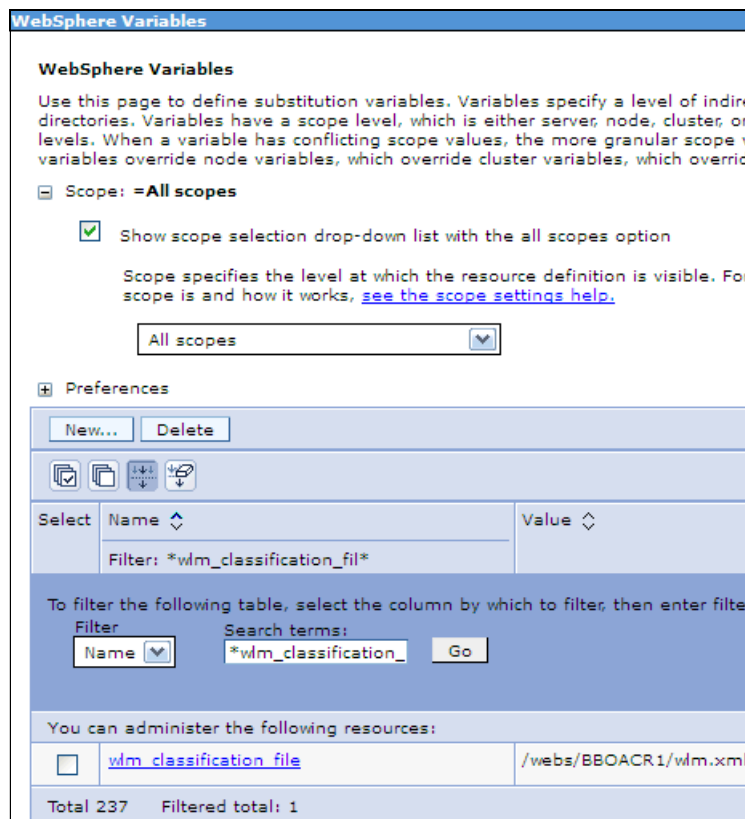


Figure 17-19 The `wlm_classification_file` environment variable

Example 17-8 shows a simple `classify.xml` file for use with sample applications.

Example 17-8 Simple `classify.xml` example

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Classification SYSTEM "Classification.dtd" >
```

```

<Classification schema_version="1.0">

  <InboundClassification type="http"
    schema_version="1.0"
    default_transaction_class="WSHTTP">
    <http_classification_info transaction_class="HTTP"
      host="wtsc60.itso.ibm.com"
      description="ITSO host">
    <http_classification_info transaction_class="WS_XML"
      uri="/xmlsamples/*"
      description = "XML" />
    <http_classification_info transaction_class="WS_PLANT"
      uri="/PlantsByWebSphere/*"
      description = "PLANTS" />
  </http_classification_info>
</InboundClassification>
</Classification>

```

For information about the full syntax of classification XML, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Frrun_wlm_tclass_dtd.html

WebSphere Application Server for z/OS V8.5 provides increased reliability, availability, and serviceability (RAS) granularity with request level classification. It allows you to specify RAS attribute values for HTTP, IIOp, optimized local adapter, and certain MDB requests.

For more information about RAS granularity, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Fae%2Ftrun_ras_granularity.html

17.10.4 Commands and tools

To analyze and monitor your WLM environment, you can use the WLM Work Queue Viewer (WQUEUE) tool. This ISPF-based tool can help you in displaying the application environments that are used on your system, as shown in the Example 17-9. For more information, refer to the link in 17.14, “Tools” on page 643.

Example 17-9 WQUEUE main panel

TIME: 12.Jul.2011 10:22:04 REL: HBB7770 SYS: SC60 PLEX: PLEX60 ENV: TSO

Select: ALL

Application Environment Monitor

| AppEnv | Type | SubName | WMAS | Del | Dyn | NQ | QLen | Str | Hav | Unb | Trm | Min | Max | ICnt |
|---------|------|---------|------|-----|-----|----|------|-----|-----|-----|-----|-----|-----|------|
| BBOC003 | CB | BBOACR1 | 004A | Off | On | 1 | 0 | 0 | 1 | 0 | 0 | 6 | 18 | 0 |
| BBOC004 | CB | BBOACR2 | 0049 | Off | On | 1 | 0 | 0 | 1 | 0 | 0 | 6 | 6 | 0 |
| BBODMGR | CB | BBODMGR | 0045 | Off | On | 1 | 0 | 0 | 1 | 0 | 0 | 6 | 6 | 0 |

You can use the SMF 120.9 browser tool to build a sample classification XML for your existing environment from the provided SMF 120.9 records, as shown in Example 17-10. For more information, refer to 17.14, “Tools” on page 643.

Example 17-10 Invoke command for SMF Browser classify function

```
java com.ibm.ws390.sm.smfview.SMF "INFILE(YOUR.SMF.DATA)"
"PLUGIN(com.ibm.ws390.sm.smfview.ClassificationXMLFilter,/path/to/put/classify.xml
)"
```

The following modify command is available to verify classification and processing cost.

F *servername*,DISPLAY,WORK,CLINFO

Example 17-11 shows the output of the command.

Example 17-11 Output of the DISPLAY,WORK,CLINFO

```
BB000281I CLASSIFICATION COUNTERS FOR HTTP WORK
BB000282I CHECKED 147, MATCHED 147, USED 0, COST 2, DESC: HTTP root
BB000282I CHECKED 147, MATCHED 147, USED 5, COST 4, DESC: ITS0 host
BB000282I CHECKED 147, MATCHED 28, USED 28, COST 3, DESC: XML
BB000282I CHECKED 119, MATCHED 114, USED 114, COST 4, DESC: PLANTS
BB000283I FOR HTTP WORK: TOTAL CLASSIFIED 147, WEIGHTED TOTAL COST 560
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,WORK,CLINFO
```

The following modify command is available to reclassify work dynamically.

F *servername*,RECLASSIFY,FILE='/path/to/new_classify.xml'

Example 17-12 shows the output of the command.

Example 17-12 Output of the RECLASSIFY,FILE

```
BB0J0129I: The /webs/BB0ACR1/wlm2.xml workload classification file was
loaded at 2011/07/10 23:41:19.655 (EDT)
BB000211I MODIFY COMMAND RECLASSIFY,FILE='/webs/BB0ACR1/wlm2.xml'
COMPLETED SUCCESSFULLY
```

Classification file note: If the new workload classification file cannot be loaded, the application server discards the reloaded classification settings. The application server continues to run with the classification settings in effect before the modify command was issued.

Use the IBM z/OS Resource Measurement Facility™ (RMF™) Postprocessor workload activity reportor WebSphere Application server Runtime Performance Advisors to periodically review WebSphere Application Server performance indicators.

For more information about the RMF workload activity report, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftprf_monitoringhealth.html

17.11 Fast response cache accelerator and caching

This section discusses fast response cache accelerator (FRCA) support and caching enhancements in WebSphere Application Server V8.5.

17.11.1 FRCA overview

FRCA is a facility of the z/OS Communication Server TCP/IP component that IBM HTTP Server for z/OS has used for years. It provides a high-speed caching mechanism where cached responses are served with high performance using a minimum amount of CPU cycles. It provides caching capabilities for both static and dynamic content, such as pictures, HTML files, servlets, and JavaServer Pages (JSP) files.

Support note: Currently, FRCA cache is supported only for non-SSL connections.

For static content requests served from FRCA cache, the performance is comparable to the same scheme on a web server. Caching static content with FRCA is 14 times faster for 1 KB files and four-to-11 times faster for 5 KB to 100 KB files than when using DynaCache. When used in dynamic caching, FRCA is using 2.7 times less CPU cycles than DynaCache. For the dynamic content requests, FRCA is taken as an extension to the DynaCache capability of each application server. There FRCA is defined as an “External Cache group” that is accessed by an adapter bean and that is owned by WebSphere Application Controller. Since z/OS 1.11, FRCA also allows web traffic to be carried on an IPv6 network.

Note: FRCA functionality needs z/OS V1.9 or higher to be used. For FRCA to work properly, the fix for APAR PK72551 (UK42691) must be applied to the Communications Server TCP/IP on z/OS Version 1.9. If this fix is not applied, the server issues the BB000347E or BB000348E error message. TCP/IP uses CSM storage to maintain this cache. So, verify and plan for your CSM usage with your system programmer.

17.11.2 Enabling FRCA in WebSphere Application Server

The FRCA function is enabled transparently by a signal from WebSphere Application Server for z/OS. As such, TCP/IP configuration statements are not required to enable this support.

To enable FRCA support within WebSphere Application Server:

1. Click **Servers** → **Server Types** → **WebSphere application servers** → *server_name* → **Container Services**.
2. Click **Dynamic cache service**, as shown in Figure 17-20 on page 630.

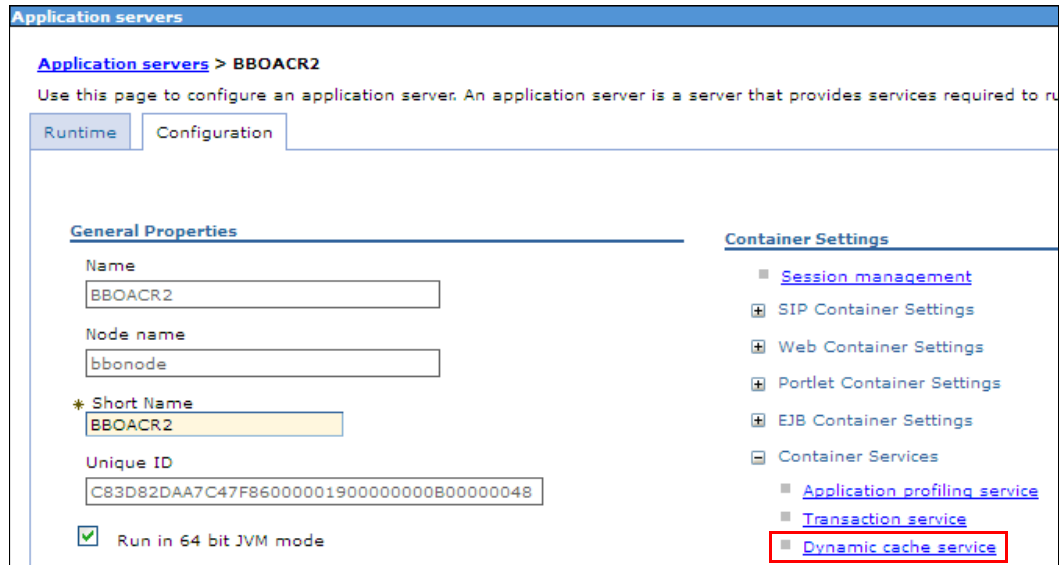


Figure 17-20 Application servers configuration window

3. Verify that the **Enable service at server startup** General Properties option is selected. Also, select **External cache groups** from the additional properties, as shown in Figure 17-21.

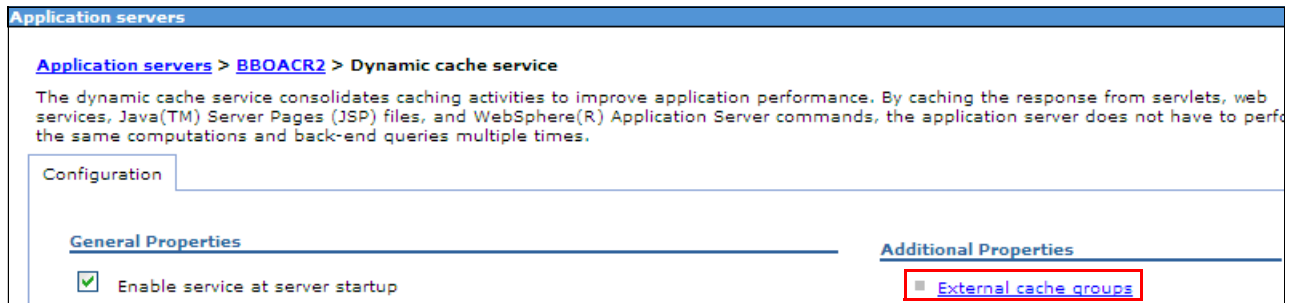


Figure 17-21 Dynamic cache service window

4. Click **New** to create a new external cache group, as shown in Figure 17-22.

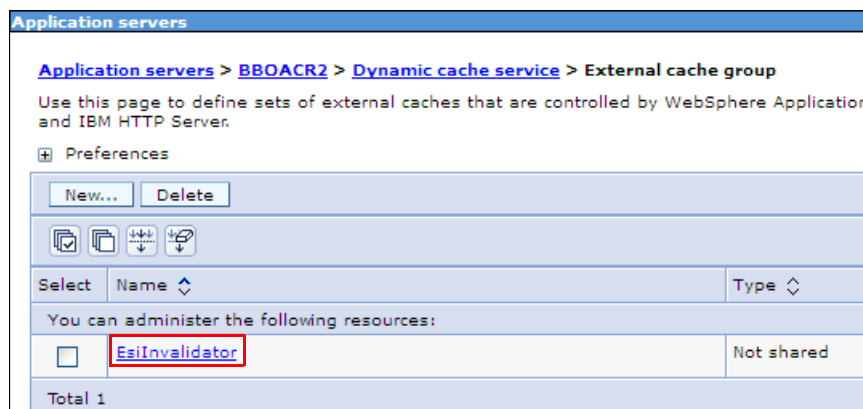


Figure 17-22 External cache group view

- Specify a name for the external cache group in the Name field, as shown in Figure 17-23 and then click **Apply**.

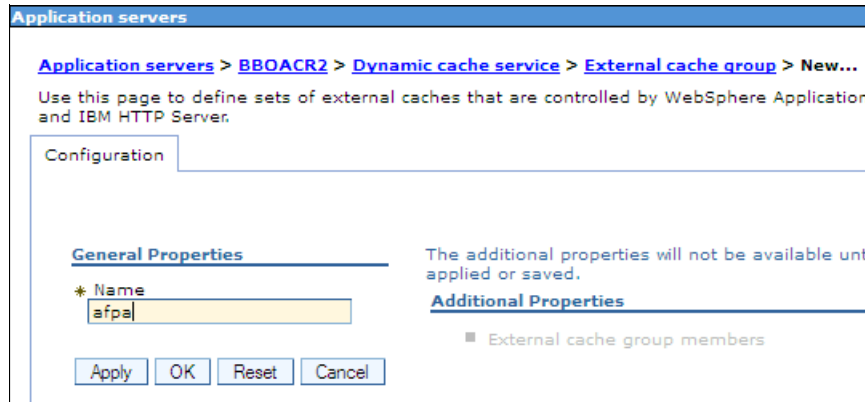


Figure 17-23 New External cache group window

- Select your newly created group, as shown in Figure 17-24.

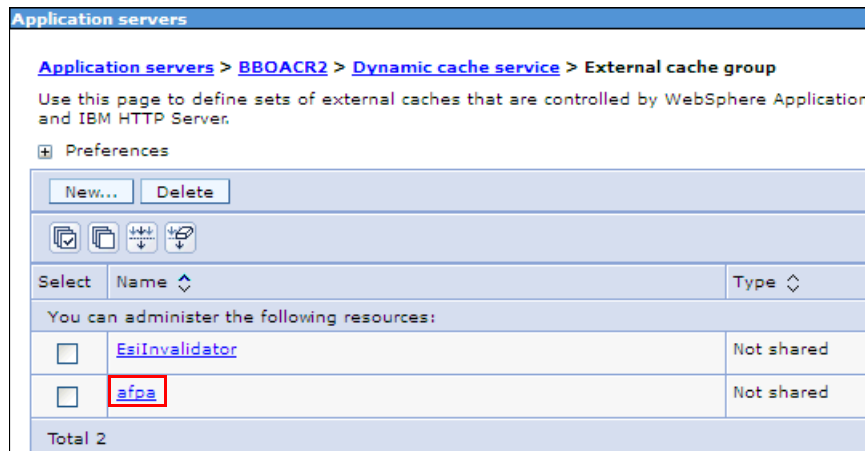


Figure 17-24 External cache group view

7. Select **External cache group members**, as shown in Figure 17-25.

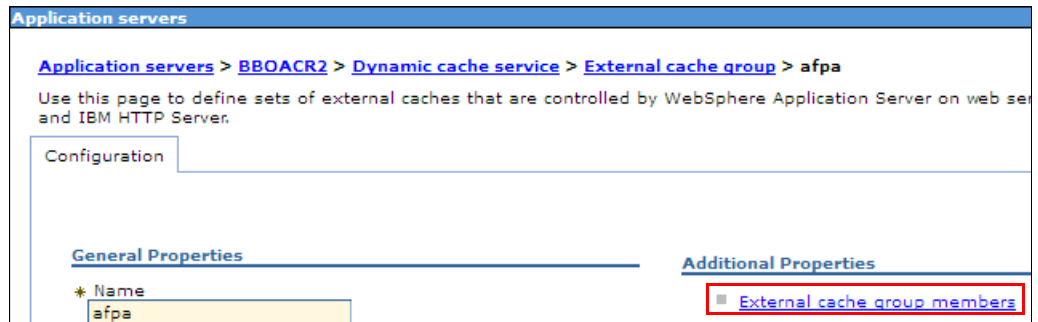


Figure 17-25 External cache group window

8. Click **New** to create the external cache group member, as shown in Figure 17-26.

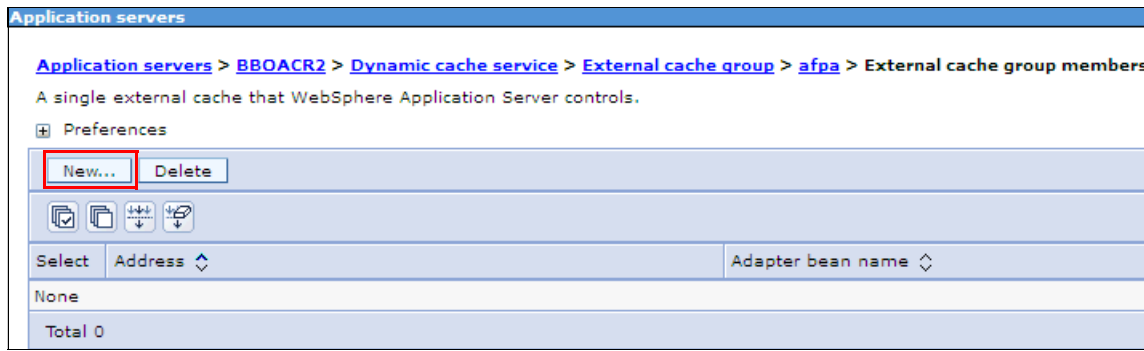


Figure 17-26 New External cache group member window

9. Select **Advanced Fast Path Architecture** with an adapter bean name of `com.ibm.ws.cache.servlet.Afpa`, and indicate a port for AFPA to listen on internally (0 for net-new usage). Select **Enable Fast response cache accelerator**, as shown in Figure 17-27. Apply your configuration by clicking **Apply**.

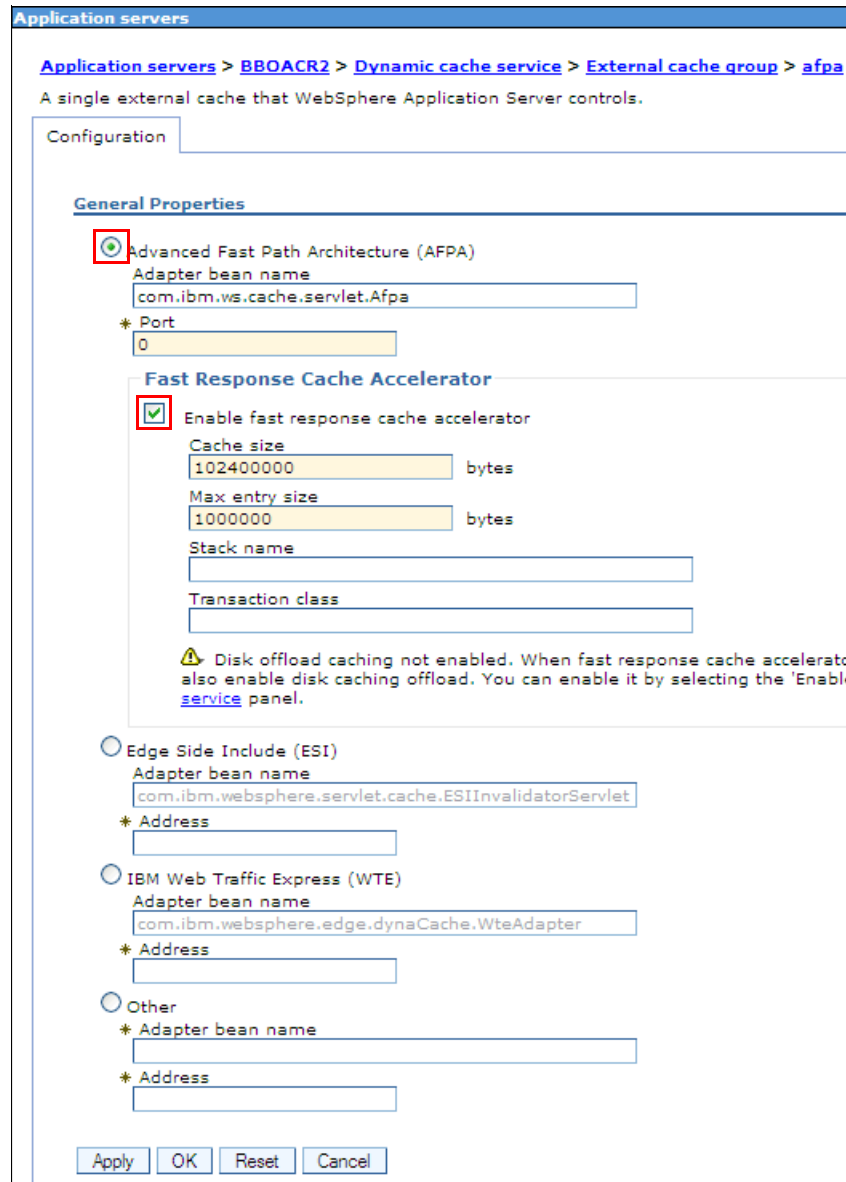


Figure 17-27 General properties page of the external cache group member

10. For a Network Deployment configuration, click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with appropriate node or nodes selected, or select the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
11. We suggest that you also enable disk caching offload by clicking **Servers** → **Server Types** → **WebSphere application servers** → `server_name` → **Container Services** → **Dyna cache service**.
12. Restart your application server by using the **Restart** button for the target server in the **Servers** → **Server types** → **WebSphere Application servers** view.

For more information, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftdyn_httpserverz.html

Disabling FRCA caching: By default, the FRCA cache is active on all channel chains that contain a web container channel and that do not contain an SSL channel. You can disable FRCA for specific channel chains and listener ports using the configuration tab for transport channels. Click **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Web Container** → **Settings** → **Web container transport chains** → **transport_chain**, and select the **Disable FRCA caching** option.

To turn on memory-to-memory replication for dynamic cache service, refer to 18.3.5, “Cache replication” on page 692. Also consider turning on portlet and servlet caching.

To use FRCA caching, add a cache policy in the cachespec.xml file for your FRCA-targeted object together with a property name that specifies the FRCA external cache group name. Include the cache specification file with the deployment module.

Verification: Review the servant’s log file to verify that the cachespec.xml file was loaded successfully. A successful load is indicated by a DYNA0047I message. In case of syntax or other errors, the cache file is not used.

In our scenario, we used the snoop servlet from the DefaultApplication.ear file, which is located in the `/WAS_product_image_path/installableApps` directory. You can invoke this file at `http://host:port/snoop` to demonstrate the FRCA enablement.

We updated a portion of the default cachespec.sample.xml file, which is located in the `/WAS_product_image_path/properties/` directory, to cache the snoop servlet into the FRCA cache, as shown in Example 17-13.

Example 17-13 FRCA enablement inside the portion of cachespec.sample.xml

```
<?xml version="1.0" ?>
<!DOCTYPE cache SYSTEM "cachespec.dtd">
<cache>
  <cache-entry>
    <class>servlet</class>
    <name>/snoop</name>
    <cache-id>
      <component id="*" type="parameter">
        <required>>false</required>
      </component>
      <component id="" type="pathinfo">
        <required>>false</required>
      </component>
      <component id="host" type="header">
        <required>>false</required>
      </component>
      <timeout>180</timeout>
    </cache-id>
  <property name="ExternalCache">afpa</property>
</cache-entry>
</cache>
```

For more information and the full syntax for the cache specification XML file, refer to 17.11.3, “Cache specification XML” on page 636.

You can use the following commands to verify that FRCA is working inside WebSphere Application Server:

- ▶ **F servername,DISPLAY,FRCA**
- ▶ **F servername,DISPLAY,FRCA,CONTENT**
- ▶ **F servername,DISPLAY,FRCA,STATS**

Example 17-14, Example 17-15, and Example 17-16 show the output from these commands.

Example 17-14 Output of the DISPLAY,FRCA command

```
BB000351I TIME OF LAST FRCA DISPLAY 2011/07/13 17:28:18.768656
BB000352I FRCA CURRENT CACHED OBJECTS 1
BB000353I FRCA TOTAL CACHED OBJECTS 1 (DELTA 0)
BB000354I FRCA CURRENT CACHED SIZE 14K
BB000355I FRCA OBJECTS PUSHED 0 (DELTA 0)
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,FRCA
```

Example 17-15 Output of the DISPLAY,FRCA,CONTENT command

```
BB000352I FRCA CURRENT CACHED OBJECTS 1
BB000358I LIST OF CACHED OBJECTS WRITTEN TO JOBLOG
BB000357I SIZE: 13351 KEY: wtsc80.itso.ibm.com:32674/snoop
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,FRCA,CONTENT
```

Example 17-16 Output of the DISPLAY,FRCA,STATS command

```
BB000351I TIME OF LAST FRCA DISPLAY 2011/07/13 17:25:41.210033
BB000352I FRCA CURRENT CACHED OBJECTS 1
BB000353I FRCA TOTAL CACHED OBJECTS 1 (DELTA 0)
BB000354I FRCA CURRENT CACHED SIZE 14K
BB000355I FRCA OBJECTS PUSHED 0 (DELTA 0)
BB000356I FRCA OBJECTS OK - 16K 1
BB000356I FRCA OBJECTS 16K - 32K 0
BB000356I FRCA OBJECTS 32K - 64K 0
BB000356I FRCA OBJECTS 64K - 256K 0
BB000356I FRCA OBJECTS 256K - 1M 0
BB000356I FRCA OBJECTS >1M 0
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,FRCA,STATS
```

Use the following command to verify TCP/IP FRCA cache, as shown in Example 17-17:

```
D TCPIP,,NET,CACHEINFO
```

You can also use the equivalent **netstat -C** command.

Example 17-17 Output of the D TCPIP,N,CACH command

```
D TCPIP,TCPIP,N,CACH
EZZ2500I NETSTAT CS V1R12 TCPIP 221
CLIENT: BBOACR1 LISTENING SOCKET: 0.0.0.0..32674
 CACHETYPE: SHARED ASID: 00BB
MAXCACHE SIZE: 0000025000 CURRCACHE SIZE: 0000000004
MAXNUM OBJECTS: 0999999999 CURRNUM OBJECTS: 0000000001
NUMCONNS: 0000000008 CONNS PROCESSED: 0000000001
```

```

CONNSDEFERRED:      000000007  CONNSTIMEDOUT:      000000000
  REQUESTSPROCESSED: 000000001  INCOMPLETEREQUESTS: 000000000
  NUMCACHEHITS:      000000001  NUMCACHEMISSES:     000000007
  NUMUNPRODCACHEHITS: 000000000
CLIENT: BBOACR1      LISTENING SOCKET: 0.0.0.0..32672
  CACHETYPE:         SHARED      ASID:                 00BB
  MAXCACHE SIZE:     0000025000  CURRCACHE SIZE:      0000000004
  MAXNUMOBJECTS:    0999999999  CURRNUMOBJECTS:     0000000001
  NUMCONNS:         0000000000  CONNSPROCESSED:     0000000000
  CONNSDEFERRED:    0000000000  CONNSTIMEDOUT:      0000000000
  REQUESTSPROCESSED: 0000000000  INCOMPLETEREQUESTS: 0000000000
  NUMCACHEHITS:     0000000000  NUMCACHEMISSES:     0000000000
  NUMUNPRODCACHEHITS: 0000000000
2 OF 2 RECORDS DISPLAYED
END OF THE REPORT

```

Attention: If you need to cache objects larger than 10 MB, set the `protocol_http_large_data_response_buffer` custom property by clicking **Environment** → **WebSphere variables** for the desired server. The value for this property must be higher than the maximum size of the object to be cached.

For information about how to optionally turn on FRCA logging, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Futrb_httperrlogs.html

For information about monitoring the cache contents and overall settings, refer to 17.11.6, “Using IBM Extended Dynamic Cache Monitor to supervise caching” on page 637.

17.11.3 Cache specification XML

With caching enabled, rules are needed to define cacheable objects and the policies that guide them. You can define rules using the elements inside the cache specification XML file, as opposed to using an API-based type of caching. You can define multiple caching instances to cache the same servlet or object inside the single server.

The cache specification file supports the following content types:

- ▶ Static
- ▶ Servlet
- ▶ Portlet
- ▶ Webservice
- ▶ JAXRPCClient, for a web service client
- ▶ Command, for a WebSphere Application Server command programming model

You can save a global `cachespec.xml` file in the application server properties directory, but the best practice is to place the cache configuration file with the deployment module inside the application’s web module `WEB-INF` or enterprise bean `META-INF` directory.

When the server starts, the cache parses the `cachespec.xml` file and extracts a set of configuration parameters from each cache-entry element. Every time a new servlet or other cacheable object initializes, the cache attempts to match each of the cache-entry elements to find the configuration information for that object. Grouping, sharing, and invalidation mechanisms are available for distributing the cache and keeping it current. Disk off load is available for overflow and persistence.

For more information and the full syntax of the cache specification XML file, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/r_dyn_cachespec.html

17.11.4 FRCA and RACF integration

FRCA services can be restricted. If access is restricted (the SERVAUTH class is active and the FRCA resource are defined) in your environment, WebSphere Application Server must be granted access.

If the access is restricted, the message shown in Example 17-18 is issued.

Example 17-18 FRCA access denied message

```
BB00nnnnE FRCA INITIALIZATION FAILED. SERVER NOT AUTHORIZED TO USE FRCA SERVICES.
IOCTL RV=%d, RC=%d, RSN=%08X
```

To use FRCA, the following RACF definitions are required:

1. Define a new SERVAUTH profile with the corresponding system name and TCP/IP procedure name:
RDEFINE SERVAUTH EZB.FRCAACCESS.<system_name>.<TCPIP_procname> UACC(NONE)
2. Give the application server control region READ access to this SERVAUTH profile:
PERMIT EZB.FRCAACCESS.<system_name>.<TCPIP_procname> CLASS (SERVAUTH) ID (UID_CR) ACCESS (READ)
3. Refresh the SERVAUTH class to activate the changes:
SETRPOTS RACLIST (SERVAUTH) REFRESH

17.11.5 Caching enhancements in WebSphere Application Server V8.5

WebSphere Application Server V8.5 introduces integration of dynamic cache with the Java Persistence API (JPA) second level (L2). Dynamic cache service plugs in as a cache provider for JPA L2 cache and shares entity states across various persistence contexts, transactions, and users for both DataCache and QueryCache. As such, it can use all the monitoring, synchronization, and replication capabilities provided with the dynamic cache. For information about enabling JPA L2 cache for your applications, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Fae%2Fr_dyn_openjpa.html

Dynamic cache provides servlet caching support for the Servlet 3.0 specification. For more information, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Fae%2Fcdyn_servlet3caching.html

17.11.6 Using IBM Extended Dynamic Cache Monitor to supervise caching

IBM Extended Dynamic Cache Monitor is a tool delivered by IBM for dynamic object cache monitoring. It provides a real-time view of the dynamic cache state and is the only way to manipulate the cache resident data. It allows for simple cache statistics, cache entries, and cache policy information for servlet cache instances.

Extended Dynamic Cache Monitor brings the following enhancements to the monitoring capabilities:

- ▶ Display the contents of object cache instances
- ▶ Display the Dynamic Cache Mbean statistics for cache instances across all members of a cluster
- ▶ Look at the push-pull table associated with a cache instance
- ▶ Search memory contents, disk contents, and the push-pull table for cache IDs using a regular expression
- ▶ Compare cache instances

For more information about installing the Dynamic Cache Monitor, go to the following website:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.express.doc/info/exp/ae/tdyn_servletmonitor.html

For more information about installing Extended Dynamic Cache Monitor, go to the following website:

http://www.ibm.com/developerworks/websphere/downloads/cache_monitor.html

17.12 Using WebSphere for z/OS Optimized Local Adapters

This section describes how to optimize local adapters. This feature is a high-performance connectivity feature that is specific to a z/OS system.

17.12.1 Introduction to Optimized Local Adapters

WebSphere for z/OS Optimized Local Adapters (WOLA) is a function introduced in WebSphere Application Server V7.0.0.4 with IBM Information Management Support (IMS™) support included in V7.0.0.12. It evolved from a local communications mechanism of the daemon server that was present for local IIOP calls in WebSphere Application Server for z/OS. Local communications routines are now externalized and programmatic APIs are available together with a standard JCA adapter.

WOLA is an inbound and outbound method of local cross-memory communication between WebSphere Application Server for z/OS and external address spaces, as shown in Figure 17-28.

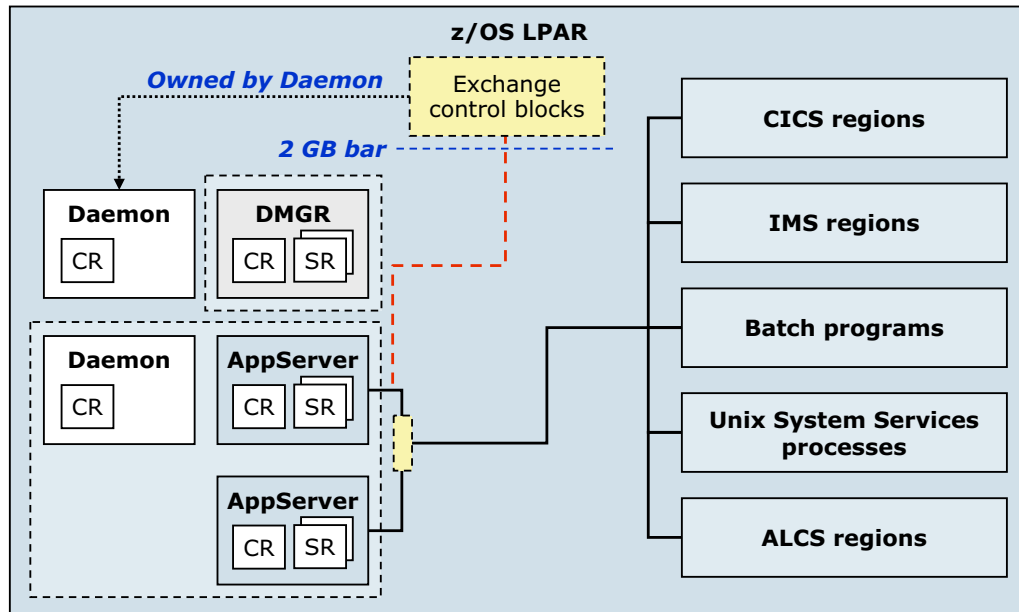


Figure 17-28 WOLA overview

As shown in Figure 17-28, WOLA communicates with the following external address spaces:

- ▶ CICS regions
- ▶ IMS regions
- ▶ Batch programs
- ▶ UNIX System Services processes
- ▶ Airlines line control (ALCS) programs

The following supported languages can be used to invoke the callable services:

- ▶ Cobol
- ▶ C/C++
- ▶ PL/I
- ▶ High level assembler

WOLA represents a way to link WebSphere Application Server for z/OS and external address spaces in a secure, optimized, high-speed, cross-memory, and bidirectional manner. WOLA is a high throughput solution with a low impact per transaction or exchange.

However, because WOLA is a low-level, cross-memory exchange mechanism between the daemon and specific WOLA-enabled control regions, it is limited to a single z/OS operating system only. The daemon-owned shared space exchange control blocks reside above the 2 GB bar. External address space always registers with the local daemon through the **BBOA1REG** call and provides target environment values together with registration name to initiate communication.

The WOLA function is provided as a complement to currently offered solutions as a fast and efficient way to invoke business logic and services inside WebSphere Application Server for z/OS.

For more information about the WOLA feature, go to the following website:

<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101490>

17.12.2 Enabling WebSphere for z/OS Optimized Local Adapters

To enable WOLA support in WebSphere Application Server V8.5:

1. Use the copyZOS.sh script to off load WOLA modules from UNIX System Services, as shown in Example 17-19. It automatically creates a 35-track load library with the name specified on the script arguments.

Example 17-19 Using copyZOS to create symbolic links and to copy WOLA modules

```
/WAS_product_image_path/bin/copyZOS.sh OLAMODS YOUR_DESIRED_LOADLIB
```

2. Use the olaRar.py script to install the WOLA resource adapter and J2C connection factory, as shown in Example 17-20. The cell scope WAS_DAEMON_ONLY_enable_adapter variable is set during the process, and the configuration is validated and saved.

Example 17-20 Install WOLA adapter using the olaRAR.py script

```
${WAS_INSTALL_ROOT}/bin/wsadmin.sh -lang jython -f  
/WAS_product_image_path/util/zos/OLASamples/olaRar.py bboce11 bbonode
```

3. Click **Environment** → **WebSphere variables**. Select **All scopes**, and verify the existence of the WAS_DAEMON_ONLY_enable_adapter variable that covers your desired scope, as shown in Figure 17-29.

WebSphere Variables

Use this page to define substitution variables. Variables specify a level of indirection for some system-defined variables and directories. Variables have a scope level, which is either server, node, cluster, or cell. Values at one scope level can override values at a greater level. When a variable has conflicting scope values, the more granular scope value overrides values at greater levels. Variables override node variables, which override cluster variables, which override cell variables.

Scope: =All scopes

Show scope selection drop-down list with the all scopes option

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, [see the scope settings help](#).

All scopes

Preferences

New... Delete

| Select | Name | Value | Scope |
|--------|-------------------------|-------|-------|
| | Filter: *enable_adapte* | | |

To filter the following table, select the column by which to filter, then enter filter criteria (wildcards: *,?,%).

Filter: Name Search terms: *enable_adapte* Go

You can administer the following resources:

| | | | |
|--------------------------|--|------|--------------|
| <input type="checkbox"/> | WAS_DAEMON_ONLY_enable_adapter | true | Cell=bboce11 |
|--------------------------|--|------|--------------|

Total 237 Filtered total: 1

Figure 17-29 New WAS_DAEMON_ONLY_enable_adapter variable in cell scope

- Verify the existence of the new resource adapter on the desired scope, as shown in Figure 17-30.

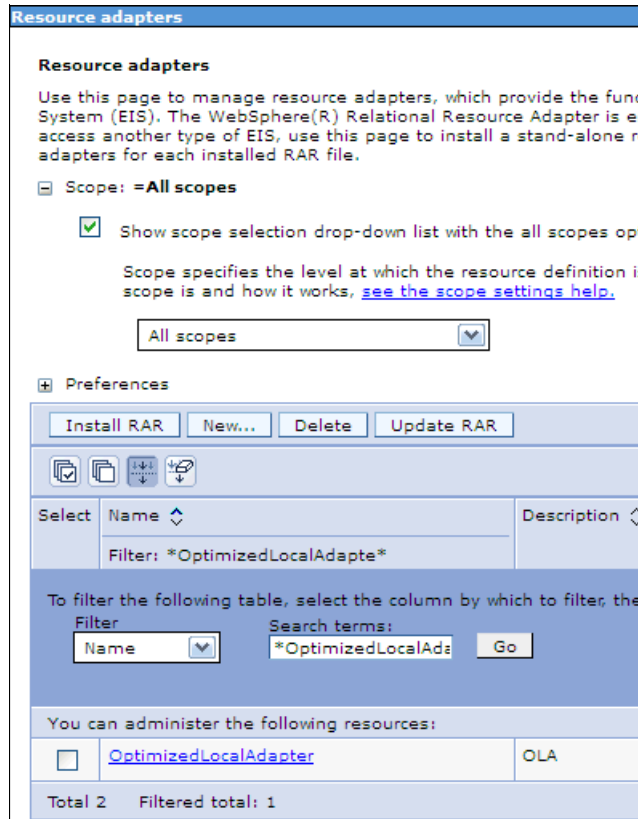


Figure 17-30 New OLA resource adapter

- Verify the existence of the new J2C connection factory on the desired scope, as shown in Figure 17-31.

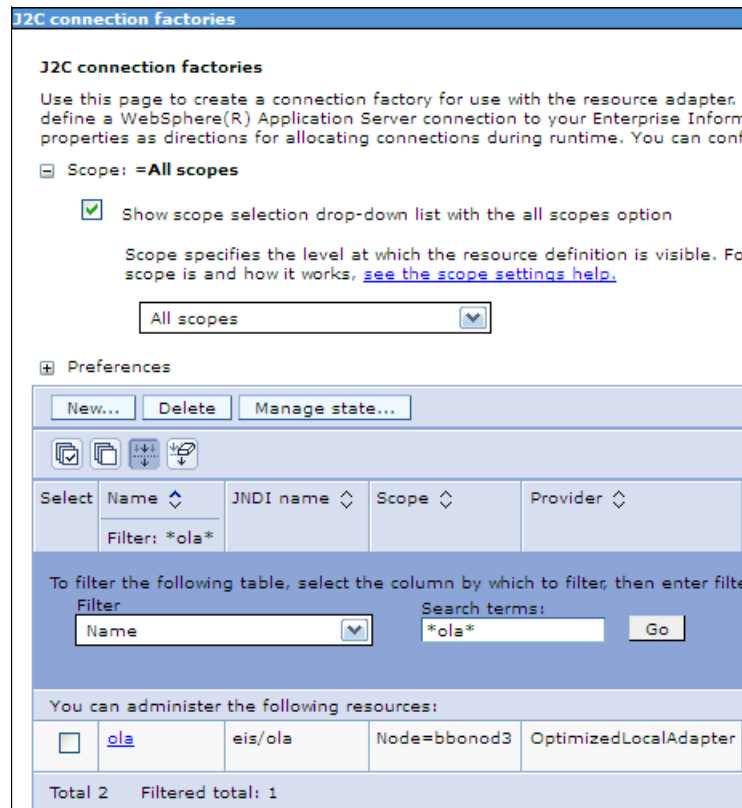


Figure 17-31 New OLA J2C connection factory

- Restart the WebSphere Application Server daemon. (This action also stops all of the connected cell servers on the same z/OS LPAR.)
- Verify the presence of the BBOM0001I message enable_adapter:1 during the daemon start.

For more information about enabling WOLA support in WebSphere Application Server, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftdat_enableconnector.html

For WOLA samples, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcdat_olasamples.html

For information about enabling WOLA support in CICS, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftdat_enableconnectorcics.html

For information about enabling WOLA support in IMS, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftdat_enableconnectorims.html

For more information about using WOLA with ALCS, go to the following website:

<http://www-01.ibm.com/software/htp/tpf/alcs/pubs/wassamp1.pdf>

For an overview of WOLA APIs and error and reason codes, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multipatform.doc%2Fae%2Fcdat_olaapis.html

In WebSphere Application Server for z.OS V8.5, WOLA also participates in the JCA failover scenarios. It can fail over to another external address space (for example, CICS region) on the same LPAR.

For more information, refer to 18.3, “Failover and failback” on page 674.

17.13 IBM HTTP Server Status monitoring page

Refer to 14.5.5, “IBM HTTP server status monitoring page” on page 515 for information about the IBM HTTP Server status monitoring page.

HTTP Server plug-in properties might affect workload distribution and performance.

For more information, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multipatform.doc%2Fae%2Fuwsv_plugin_props.html

17.14 Tools

There is a wide variety of WebSphere Application Server V8.5 tools available that can help with application server performance monitoring or application profiling. You can use these tools during problem determination and as a means of measuring the performance of your application and the components that it encompasses. When using these tools in production environments, consider the possible performance costs that are associated with collecting data.

Refer to 19.4.6, “z/OS monitoring” on page 731 for more information about performance monitoring tools.

Also, consult the following resources:

- ▶ To understand how to monitor Performance Monitoring Infrastructure (PMI) data with Tivoli Performance Viewer, go to the following website:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.multipatform.doc%2Fae%2Ftprf_tpvmonitor.html
- ▶ For RMF usage, refer to the *z/OS V1R12.0 Resource Measurement Facility User's Guide*, SC33-7990
- ▶ For information about IBM Tivoli Composite Application Manager for Application Diagnostics V7.1, go to the following website:
<http://www-01.ibm.com/software/tivoli/products/composite-application-mgr-diagnostics/>
- ▶ To download IBM Support Assistant, go to the following website:
<http://www-01.ibm.com/software/support/isa/>

You can find a complete list of IBM Support Assistant add-ons at the following website:

http://www-01.ibm.com/support/docview.wss?uid=swg27013116&cmp=101AM&ct=101AMES3&cr=01_Net&cm=S&csr=Websphere&co=0n&cot=A&cd=2011-04-10&cpg=CIOP&cn=Impact2011_ISA

- ▶ To download WLMQUE with Rexx Alternate Library, go to the following website:

<http://www-03.ibm.com/systems/z/os/zos/features/wlm/tools/wlmque.html>

- ▶ To download SMF120.9 browser with instructions, go to the following website:

<https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=zosos390>

- ▶ To download the Extended Dynamic Cache Monitor, go to the following website:

<http://www.ibm.com/developerworks/apps/download/index.jsp?contentid=210854&filename=cachemonitor.zip&method=http&locale=>

- ▶ Visit the IBM tools and toys for z/OS page at the following website:

<http://www-03.ibm.com/systems/z/os/zos/features/unix/tools/>



Clustering and high availability

High availability is a system design approach that tries to eliminate single points of failure and ensure system availability by employing redundancy.

The high-availability framework that is delivered with WebSphere Application Server for z/OS is complemented by a degree of availability that is provided by hardware components, such as IBM z/Architecture® and the z/OS operating system. Parallel Sysplex as a zSeries clustering technology in this topology provides efficient redundancy with a view of the systems as a single logical computing environment. A coupling facility in a sysplex is used for data sharing and reliable messaging between the members of the complex. Several logical partitions or independent machines can be used to prevent unplanned software or hardware outages. The operating system takes advantage of the self-healing attributes of the hardware and extends them by adding functions, such as recovery services for all operating system code, address space isolation, and storage key protection. Functions, such as Workload Manager (WLM), Resource Recovery Services (RRS), and automatic restart manager (ARM), assure the availability of applications. A sysplex distributor with defined dynamic virtual IP address (DVIPA) handles IP address availability with the possibility of a host role assignment and takes advantage of workload balancing.

This chapter focuses on the z/OS system and introduces high-availability concepts and practices for WebSphere Application Server for z/OS features. It includes the following topics:

- ▶ Clustering on z/OS systems
- ▶ High availability
- ▶ Failover and fallback
- ▶ Enabling multiple servants
- ▶ Additional resources

18.1 Clustering on z/OS systems

Clustering technology is an integral part of WebSphere Application Server for z/OS V8. In this section, we provide information about the concept of clustering and how to create a cluster on z/OS systems.

18.1.1 Clustering for scalability and failover

Clustering as a high availability approach is used extensively in WebSphere Application Server to provide for scalability and failover protection at the same time. A *cluster* consists of multiple copies of the same component with the expectation that at least one of the copies will be available to service a request. In general, the cluster works as a unit where there is collaboration among the individual copies to ensure that the request can be directed toward a copy that can service the request.

A WebSphere Application Server cluster is composed of individual cluster members, with each member containing the same set of applications. In front of a WebSphere Application Server cluster is a *workload distributor* that routes the workload to individual members. z/OS Workload Manager (WLM) assigns system resources to units of work from the workload, making a best attempt to allow all work to meet the specified goals.

Clusters can be vertical within an LPAR (that is, two or more cluster members residing in the same z/OS system), or they can be placed horizontally across LPARs on different machines to obtain the highest availability in the event that an LPAR or machine that contains a member has an outage. In a vertical cluster, the servers compete with each other for resources.

For more information about clustering in WebSphere Application Server for z/OS V8, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcrun_srvgrp.html

18.1.2 Creating a cluster on a z/OS system

To create a WebSphere Application Server for z/OS cluster:

1. Click **Servers** → **Clusters** → **WebSphere application server clusters**.
2. Click **New** in the WebSphere Application Server clusters view, as shown in Figure 18-1.

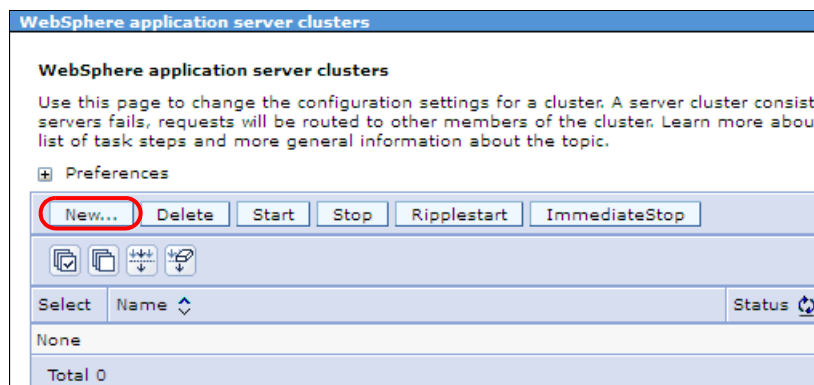


Figure 18-1 WebSphere application server clusters view

3. Enter basic cluster information, as shown in Figure 18-2. If you are converting an existing application server into the cluster, leave the Short name field empty to default to the generic short name of that application server. Click **Next**.

The screenshot shows the 'Create a new cluster' wizard at Step 1. The left sidebar lists four steps: Step 1 (highlighted), Step 2, Step 3, and Step 4. The main area is titled 'Enter basic cluster information' and contains the following fields and options:

- * Cluster name: WTSCPLX8
- Short name: (empty)
- Prefer local. Specifies whether enterprise bean requests client resides when possible.
- Configure HTTP session memory-to-memory replication

At the bottom, there are 'Next' and 'Cancel' buttons, with 'Next' highlighted by a red box.

Figure 18-2 Enter basic cluster information

4. Create the first cluster member. In the Select basis for first cluster member section, select **Create the member by converting an existing application server**, as shown in Figure 18-3. Click **Next**.

The screenshot shows the 'Create a new cluster' wizard at Step 2. The left sidebar lists four steps: Step 1, Step 2 (highlighted), Step 3, and Step 4. The main area is titled 'Create first cluster member' and contains the following fields and options:

- The first cluster member determines the server settings for the cluster member template is created from the first member and stored as part of the cluster data members are copied from this template.
- * Member name: BBOACR1
- Select node: bbonode(ND 8.0.0.0)
- Short name: BBOACR1
- * Weight: 2 (0..20)
- Generate unique HTTP ports
- Select how the server resources are promoted in the cluster: Cluster
- Select basis for first cluster member:
 - Create the member using an application server template. defaultZOS
 - Create the member using an existing application server as a template. bbocell/bbonode(ND 8.0.0.0)/BBOACR1
 - Create the member by converting an existing application server. bbocell/bbonode(ND 8.0.0.0)/BBOACR1
 - None. Create an empty cluster.

At the bottom, there are 'Previous', 'Next', and 'Cancel' buttons, with 'Next' highlighted by a red box.

Figure 18-3 Create first cluster member

5. Add additional cluster members by specifying the member name and short name. Click **Add Member**, as shown in Figure 18-4. Click **Next**.

Create a new cluster

Create a new cluster

Step 1: Enter basic cluster information

Step 2: Create first cluster member

→ **Step 3: Create additional cluster members**

Step 4: Summary

Create additional cluster members

Enter information about this new cluster member, and click Add Member to add this member to the cluster. A server configuration template is created from the first member, and additional cluster members are copied from this template.

* Member name

Select node

Short name

* Weight
 (0..20)

Generate unique HTTP ports

Add Member

Use the Edit function to modify the properties of a cluster member in this list. Use the Delete function to remove a cluster member from this list. You are not allowed to edit or remove the first member.

| Select | Member name | Nodes | Version |
|-------------------------------------|-------------|---------|------------|
| <input checked="" type="checkbox"/> | BBOACR1 | bbonode | ND 8.0.0.0 |
| <input type="checkbox"/> | BBOACR2 | bbonod2 | ND 8.0.0.0 |
| <input type="checkbox"/> | BBOACR3 | bbonode | ND 8.0.0.0 |
| Total 3 | | | |

Figure 18-4 Create additional cluster members

6. Review your setup in the summary, and click **Finish**, as shown Figure 18-5.



Figure 18-5 Summary

7. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
8. Restart the converted application server by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Replication note: When you select HTTP replication, a replication domain of the same name as the cluster name is created automatically with a Single replica option.

For more information about creating a cluster, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_wlm_cluster_v61.html

18.2 High availability

The high availability framework that is provided with the product eliminates single points of failure and provides peer-to-peer failover for applications and processes that are running within the product environment.

In this section, we provide an overview of the high-availability features in WebSphere Application Server for z/OS.

18.2.1 High availability manager

High availability manager provides services for product components so that they can make themselves highly available. By default, the high availability manager instance runs on every application server, proxy server, node agent, and deployment manager in a cell.

A cell can be divided into multiple high-availability domains known as *core groups*. Each high availability manager instance establishes network connectivity with all other high availability manager instances in the same core group, using a specialized, dedicated, and configurable transport channel. The transport channel provides mechanisms that allow the high availability manager instance to detect when other members of the core group start, stop, or fail. Automatic restart manager (ARM), Tivoli System Automation, or other automation software can be configured to restart failed WebSphere Application Server controllers.

Within a core group, high availability manager instances are elected to coordinate high availability activities. An instance that is elected is known as a *core group coordinator*. The coordinator is highly available, such that if a process that is serving as a coordinator stops or fails, another instance is elected to assume the coordinator role, without loss of continuity.

The high availability manager periodically runs a number of background tasks, such as checking the health of highly-available singleton services that it is managing. Most of these background tasks consume trivial amounts of CPU.

The exceptions are the regularly scheduled Discovery and Failure Detection Protocols:

- ▶ The Discovery Protocol discovers when other core group processes start and open network connections to these other members.
- ▶ The View Synchrony Protocol establishes reliable messaging with other core group members after the connections are opened.
- ▶ The Failure Detection Protocol detects when other core group members stop or become unreachable because of a network partition.

Distribution and Consistency Services

The Distribution and Consistency Services (DCS) transport chain provides the underlying group services framework for the high availability manager, such that each application server process knows the health and status of JVMs and singleton services. DCS provides a view of synchronous services to the high availability manager. DCS itself uses reliable multicast messaging (RMM) as its publish and subscribe message framework.

RMM is an ultra high speed publish and subscribe system that WebSphere uses internally for its core group communication fabric and for DRS traffic. WebSphere Application Server for z/OS running in sysplex environment can use cross-system coupling facility services as an alternate protocol provider to enhance DCS performance. It reduces the consumption of system resources for DCS traffic and especially view changes.

High availability manager provides the following services:

- ▶ Memory-to-memory replication
- ▶ Singleton failover
- ▶ Workload management routing
- ▶ On-demand configuration routing

We describe these services in the sections that follow.

Memory-to-memory replication

The data replication service (DRS) that is provided with the WebSphere Application Server is used to replicate HTTP session data, stateful EJB sessions, and dynamic cache information among cluster members. When DRS is configured for memory-to-memory replication, the transport channels that are defined for the high availability managers are used to pass this data among the cluster members.

Singleton failover

Singleton failover is a cluster-based service. Singleton services that use this framework include the transaction managers for cluster members and the default messaging provider, also known as the service integration bus.

Workload management routing

In the following section, we refer to workload management (WLM) as the internal WebSphere component and Workload Manager as the z/OS system component.

Workload management propagates the following classes or types of routing information:

- ▶ Routing information for the default messaging engine, which is also referred to as the service integration bus
- ▶ Routing HTTP requests through the IBM WebSphere Application Server proxy server
- ▶ Routing Web Services Addressing requests through the IBM WebSphere Application Server proxy server
- ▶ Routing Session Initiation Protocol (SIP) requests.

WLM uses the high availability manager to both propagate the routing information and to make it highly available. Although WLM routing information typically applies to clustered resources, it can also apply to non-clustered resources, such as stand-alone messaging engines.

The Workload Manager for z/OS system component is working together with the sysplex distributor to provide intelligent routing and workload balancing. When the dynamic virtual IP address (DVIPA) is used as the daemon IP name for the cell, Workload Manager capabilities are used for workload balancing and failover of IIOP requests between the LPARs. Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP).

In a cell, one location service daemon definition exists for each sysplex node group. A location service daemon process runs on each system that has a node in a sysplex node group in that cell. When a client makes a remote call to an enterprise bean, a location service daemon determines which server or servers are eligible to process the request and routes the request to the selected server. This mechanism is highly efficient.

For more information about Workload Manager, refer to 15.2.3, “Workload management benefits” on page 538.

On-demand configuration routing

In a WebSphere Application Server Network Deployment system with high availability manager enabled, the on-demand configuration routing is used for IBM WebSphere Application Server proxy server routing.

State data exchange

The high availability manager provides a specialized messaging mechanism that enables processes to exchange information about their current state. Each process sends or posts information related to its current state and can register to be notified when the state of the other processes changes. This mechanism is commonly referred to as the bulletin board. The workload management (WLM) component uses this mechanism to build and maintain routing table information. Routing tables built and maintained using this mechanism are highly available.

For more information about high availability configuration on z/OS, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcontainer_update_ha_apps_zos.html

For information about how to disable high availability manager on a process, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_ha_ham_enable.html

Disabling the high availability manager: When you disable the high availability manager on a core group member, make sure that you disable it on all the other core group members as well. You can define a special core group for the high availability manager disabled processes.

Do not disable the high availability manager on administrative processes, such as node agents and the deployment manager, unless the high availability manager is disabled on all processes that they manage.

Also, leave the high availability manager enabled on any application server that produces or consumes either IIOF or messaging engine routing information.

For more information about IIOF support with high availability infrastructure disabled, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_wlm_cluster_routetable.html

18.2.2 Core groups

A *core group* is a high-availability domain that consists of a set of processes in the same cell that can directly establish high-availability relationships. Highly available components can fail

over only to another process in the same core group, and replication can occur only between members of the same core group.

A cell must contain at least one core group, although multiple core groups are supported. By default, a cell has a single core group called *DefaultCoreGroup*. A single core group is usually sufficient. However, some topologies or special circumstances require multiple core groups. Bridges can be established between core groups.

By default, every deployment manager, node agent, application server, and proxy server is a member of a core group and has the high availability manager service enabled. When a process is created, it is added automatically to a core group. The core group membership is stored in a WebSphere Application Server configuration document. You can move processes from one core group to another. When a core group member starts, the core group transport and the associated default Discovery Protocol, default Failure Detection Protocol, and View Synchrony Protocol also start.

Figure 18-6 shows the high availability manager protocol traffic within a core group.

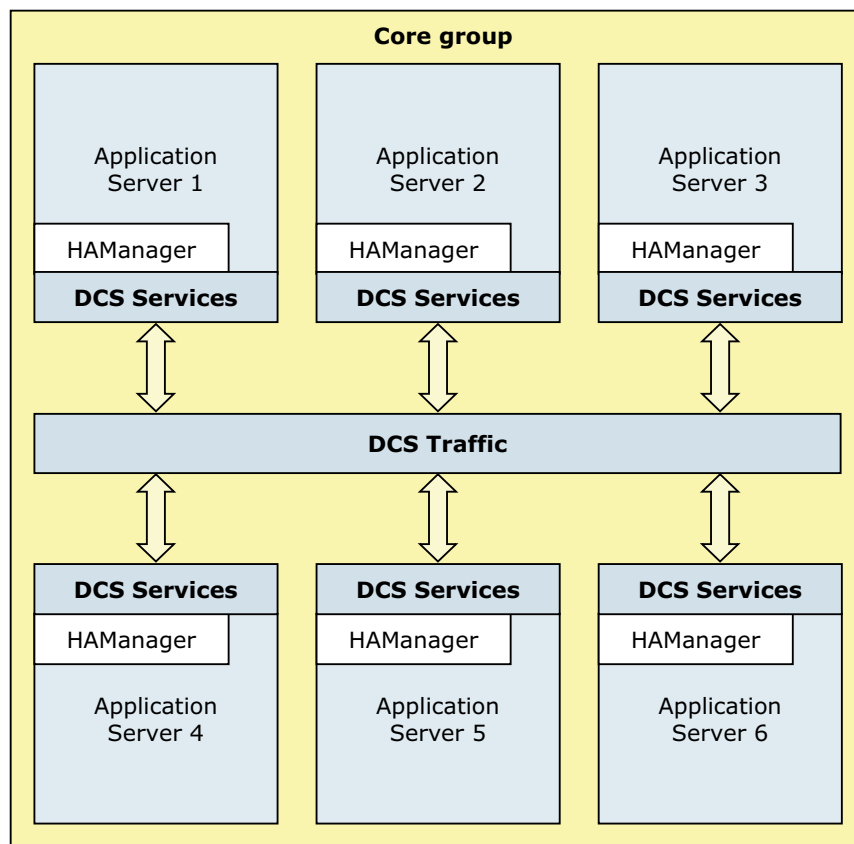


Figure 18-6 Core group DCS traffic

Each core group contains a core group coordinator to manage its high-availability relationships and a set of high-availability policies that are used to manage the highly-available components within that core group.

The following aspects are managed by the core group coordinator:

- ▶ Maintaining all group information, including the group name, group members, and the policy of the group
- ▶ Keeping track of the states of group members as they start, stop, or fail, and communicating that information to every member
- ▶ Assigning singleton services to group members and handling failover of services based on core group policies

When a JVM process with the active coordinator no longer active (because it is stopped or crashes), the high availability manager elects the first inactive server in the preferred coordinator servers list. If there are no servers available, the high availability manager elects the lexically lowest named inactive server.

The newly elected coordinator initiates a state rebuild, sending a message to all JVMs in the core group to report their states. This operation is the most processor-intensive operation of a coordinator.

Creating a new core group

To create a new core group:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Click **New** to create a new core group, as shown in Figure 18-7.

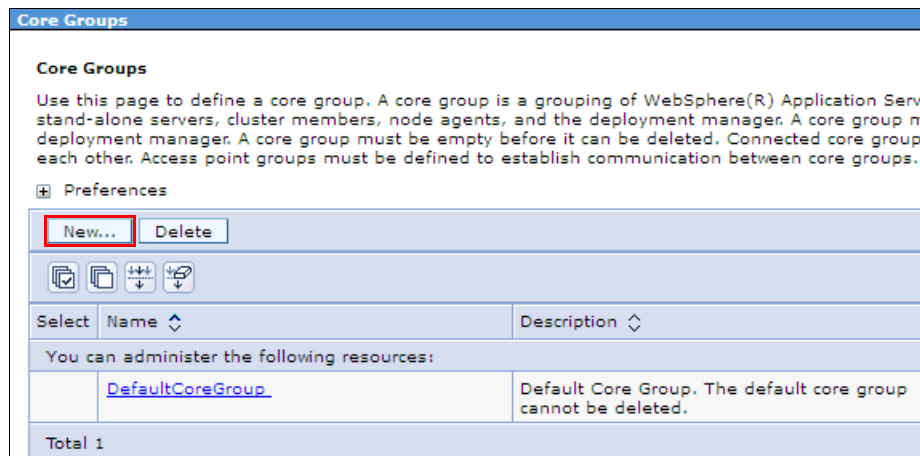


Figure 18-7 Create a new core group

- Specify the name, the number of coordinators, and the transport memory size in the Configuration tab, as shown in Figure 18-8. Click **Apply**.

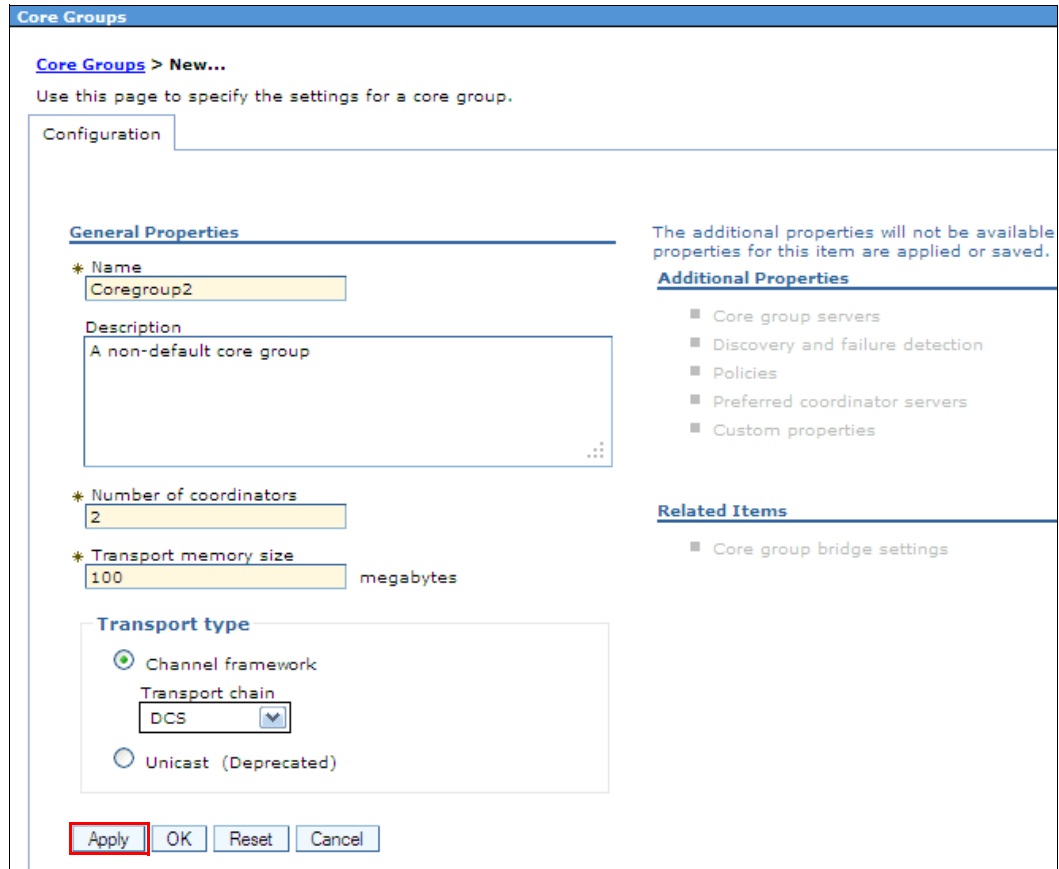


Figure 18-8 New core group window in core group view

- Add Additional Properties.
- Click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

Inherited values: The number of coordinators and the transport memory size values for the new core group are inherited from the DefaultCoreGroup.

The core group must not have more than 50 members. If your cell has more than 50 processes defined, consider creating a second core group and moving the overflow members into it.

Moving core group members between core groups

You can move processes from one core group to another as long as the following core group requirements are not violated:

- ▶ A non-empty core group retains at least one node agent or deployment manager as a member of that group. (The high availability manager configuration change listeners are available only on the node agent or deployment manager servers.)

- ▶ All members of a cluster must be members of the same core group. If one or more of the servers that you are moving belongs to a cluster, you must move all of the members of that cluster. A core group can span multiple product clusters.

To move core group members to another core group:

1. Click **Servers** → **Core Groups** → **Core group settings**, as shown in Figure 18-9.
2. Select your desired core group, as shown in Figure 18-9.

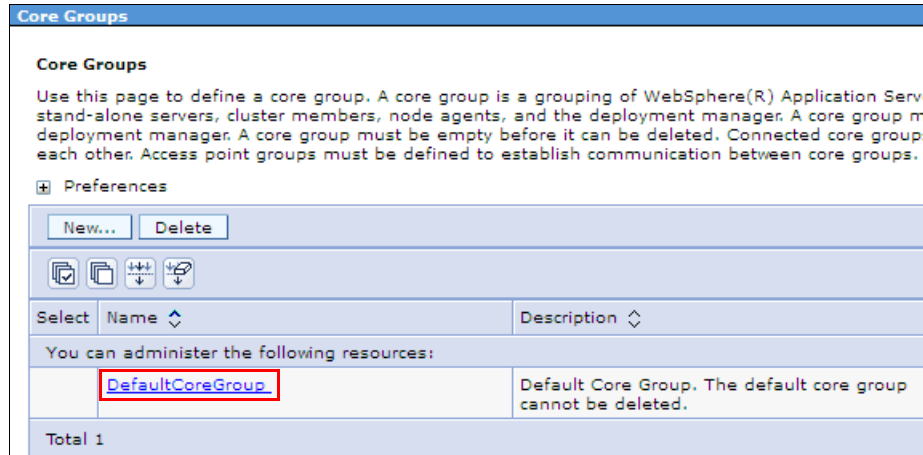


Figure 18-9 Core Groups view

3. Click **Core group servers** under Additional Properties, as shown in Figure 18-10.

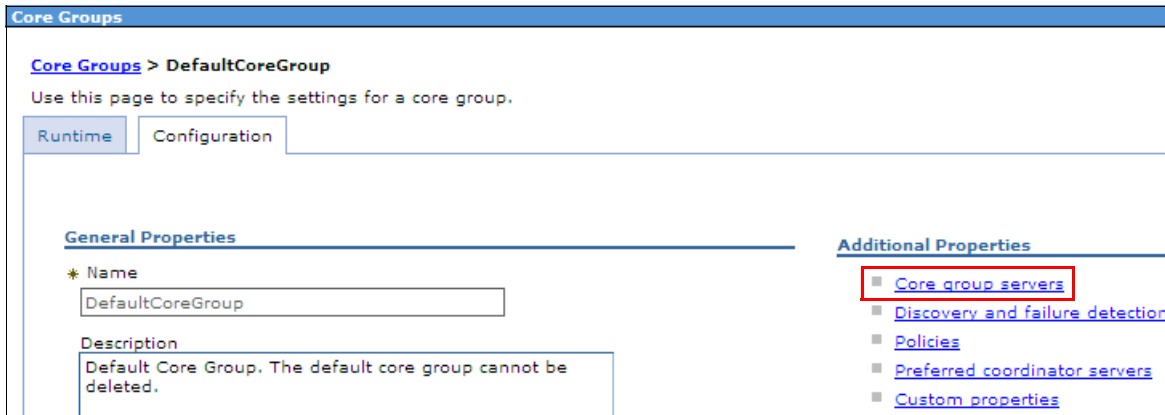


Figure 18-10 Specifying additional properties

- Select the members that you want to move to another core group, and click **Move**, as shown in Figure 18-11.

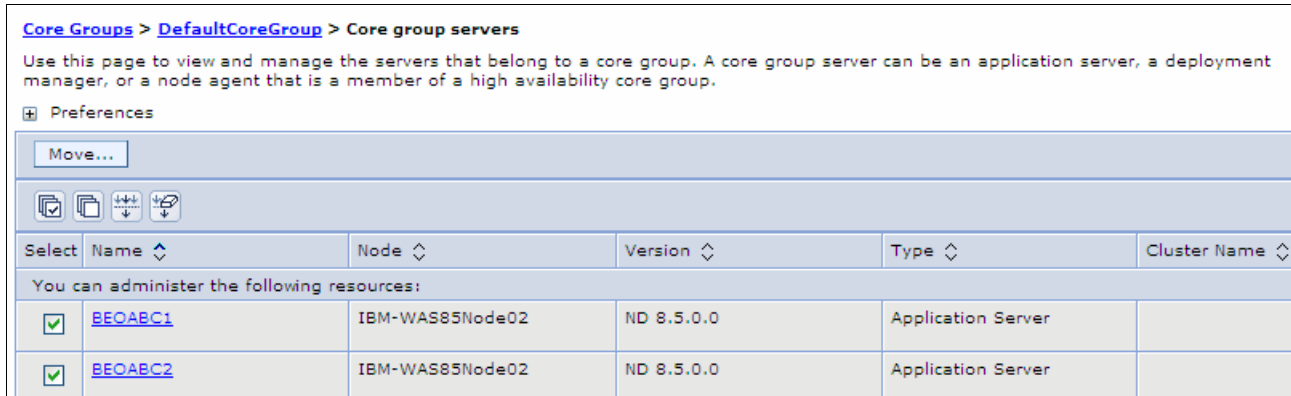


Figure 18-11 Select members

- From a drop-down menu, select a core group name to which to move members, and click **Apply**, as shown in Figure 18-12.

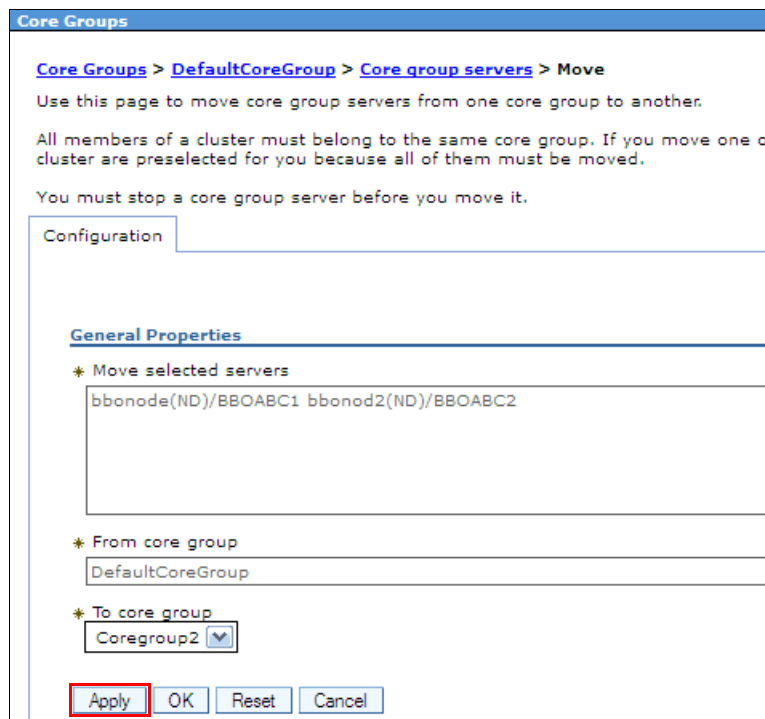


Figure 18-12 Move core group servers

- Click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

Important: You cannot move a core group member to another core group while it is running.

Each process can be a member of only one core group, and all members of a given cluster must belong to the same core group.

You need to follow a special procedure when moving node agents or deployment manager processes. For more information, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_ha_addcgmems.html

Setting core group custom properties

High availability protocols include the following major categories:

- ▶ A collection of lower level protocols, which are also referred to as the *lower-level wire format protocols*. The setting for the IBM_CS_WIRE_FORMAT_VERSION core group custom property determines the protocol version that is used for this group of protocols.
- ▶ A collection of higher level protocols, which are also referred to as the *high availability manager protocols*. The setting for the IBM_CS_HAM_PROTOCOL_VERSION core group custom property determines the protocol version that is used for this group of protocols.

To set or change core group custom properties:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 18-13.

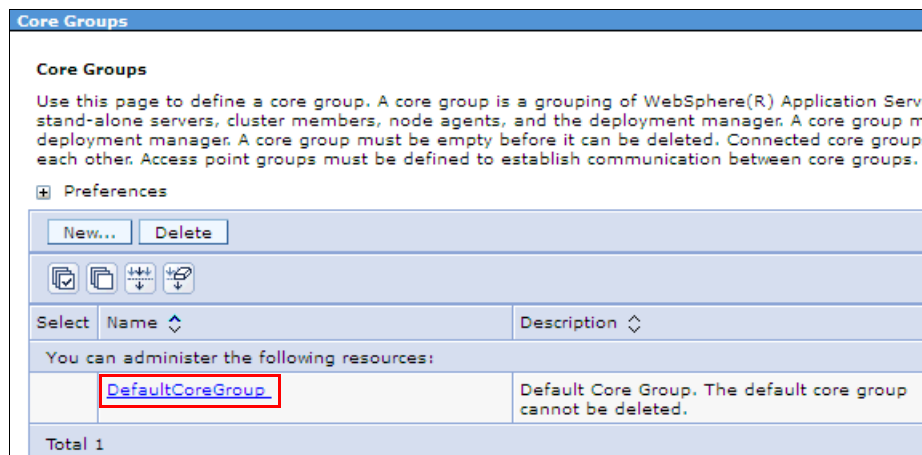


Figure 18-13 Select the core group

3. Select **Custom properties** under Additional Properties, as shown in Figure 18-14.

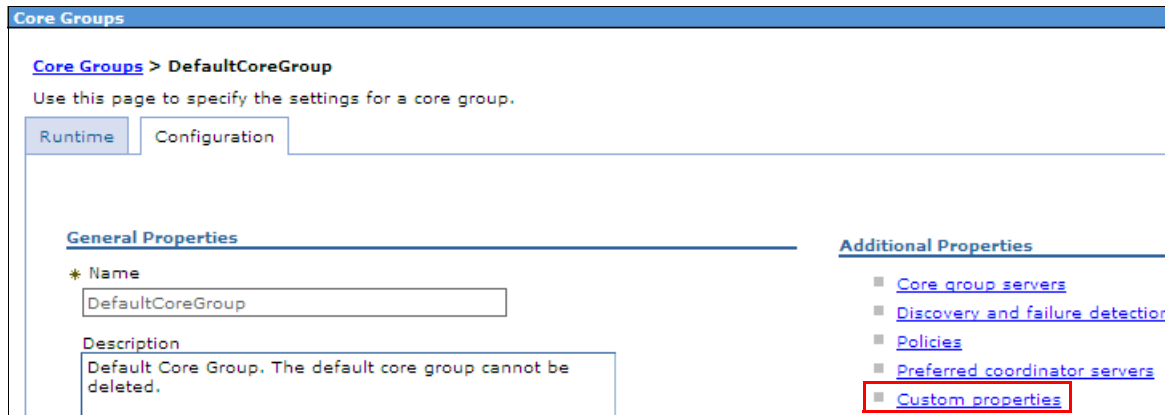


Figure 18-14 Specifying additional properties

4. Click **New** to define the set of properties for IBM_CS_HAM_PROTOCOL_VERSION and IBM_CS_WIRE_FORMAT_VERSION, as shown in Figure 18-15.

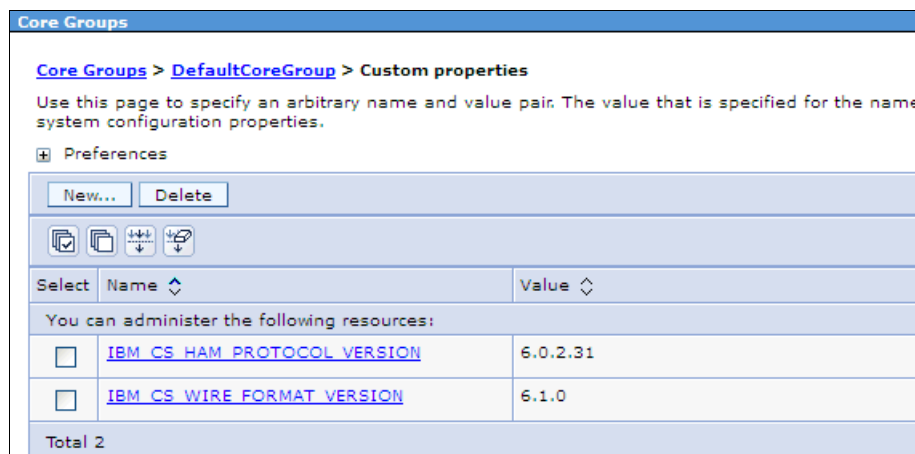


Figure 18-15 Custom properties

5. Click **Apply**.
6. Click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

Hint: By default, the protocol is set to the oldest supported version. Investigate the latest version for the group of protocols that your installation supports and add the protocol custom properties, as shown in Figure 18-15, for protocol improvements.

For a list of core group custom properties along with supported protocol versions, visit:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.webspere.zseries.doc%2Fae%2Furun_ha_cg_custprop.html

The DCSV0005I and HMGR0226I messages indicate the currently used version of the protocols, as shown in Example 18-1 and Example 18-2.

Example 18-1 Message HMGR0226I output

```
ExtendedMessage: BB000222I: HMGR0226I: The core stack configuration parameter
IBM_CS_WIRE_FORMAT_VERSION has been set to 6.0.2.31.
```

Example 18-2 Message DCSV0005I output

```
ExtendedMessage: BB000222I: DCSV0005I: DCS Stack DefaultCoreGroup at Member
bbocell\bodmgr\dmgr: Started. Stack version information: DCSBV_WAS6_1_20060409.
Stack protocol information: 61002.
```

Support note: If you are running on Version 7.0.0.1 or later, set the `IBM_CS_HAM_PROTOCOL_VERSION` core group custom property to 6.0.2.31 for all core groups to avoid a possible high-availability state outage during core group bridge failover. When this custom property is set to 6.0.2.31, the remaining bridges recover the high-availability state of the failed bridge without the data being unavailable in the local core group.

Bridging core groups

If members of different core groups need to share WLM routing information, use the *core group bridge service* to connect these core groups. The core group bridge service uses access point groups to connect the core groups. A core group access point defines a set of bridge interfaces that resolve to IP addresses and ports. The core group bridge service uses this set of bridge interfaces to enable members of one core group to communicate with members of another core group.

To create a core group bridge service for two core groups within the same cell with mesh topology, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group bridge settings**.
2. Select **Access point groups** under Additional Properties, as shown in Figure 18-16.

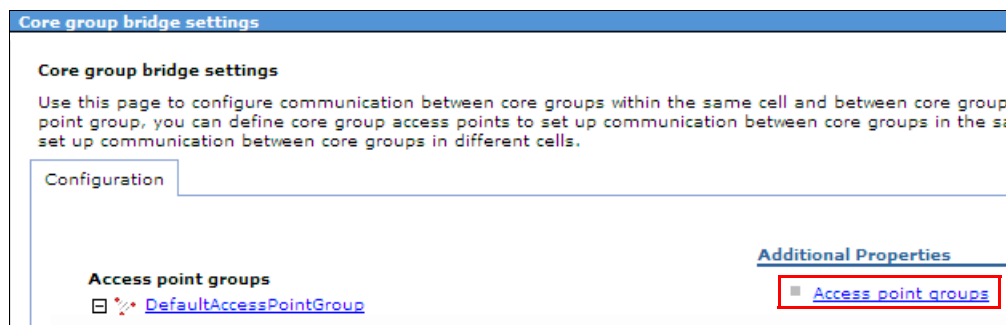


Figure 18-16 Access point groups

3. For our topology, we select **DefaultAccessPointGroup**, as shown in Figure 18-17. To use chain topology, you need to create multiple access point groups.

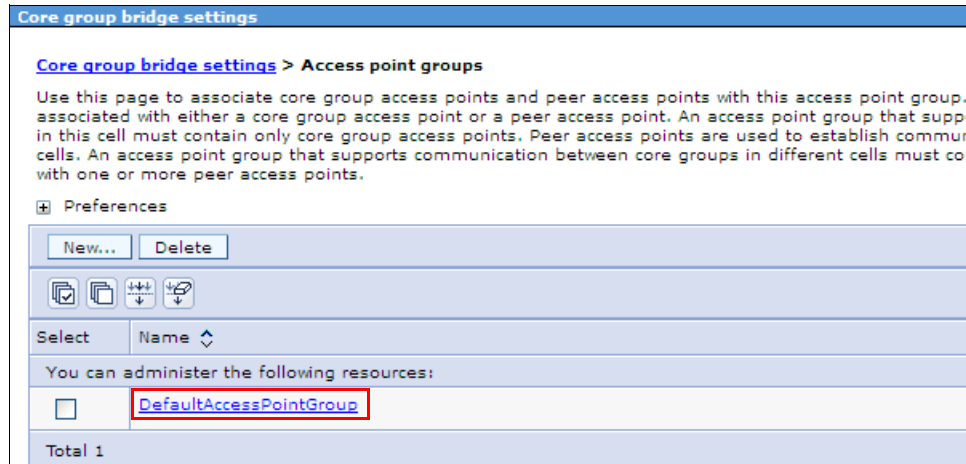


Figure 18-17 Access point groups window

4. Select **Core group access points**, as shown in Figure 18-18.

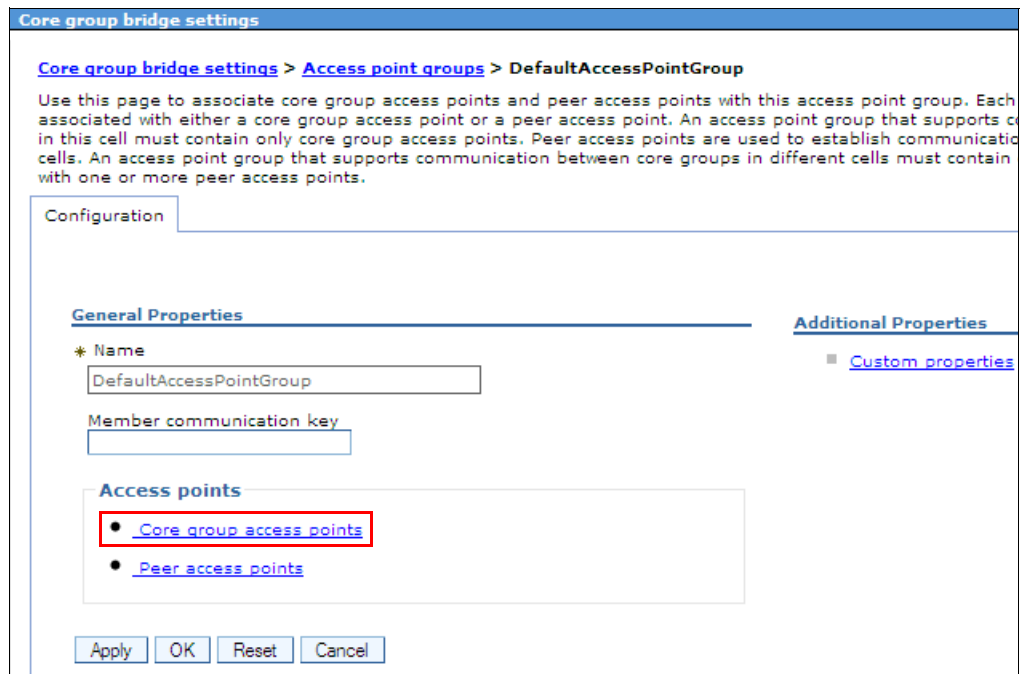


Figure 18-18 DefaultAccessPointGroup configuration window

5. Select **CGAP_1\DefaultCoreGroup**, and click **Show Detail**, as shown in Figure 18-19.

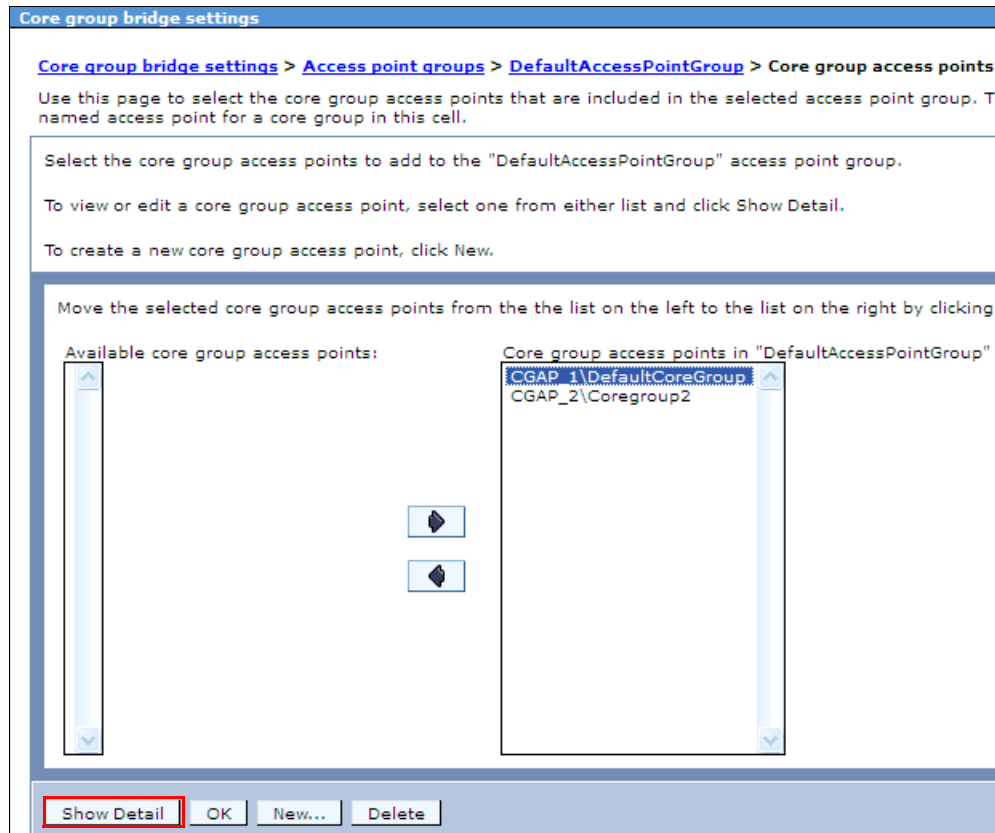


Figure 18-19 Core group access points window

6. Select **Bridge interfaces** from Additional Properties, as shown in Figure 18-20.

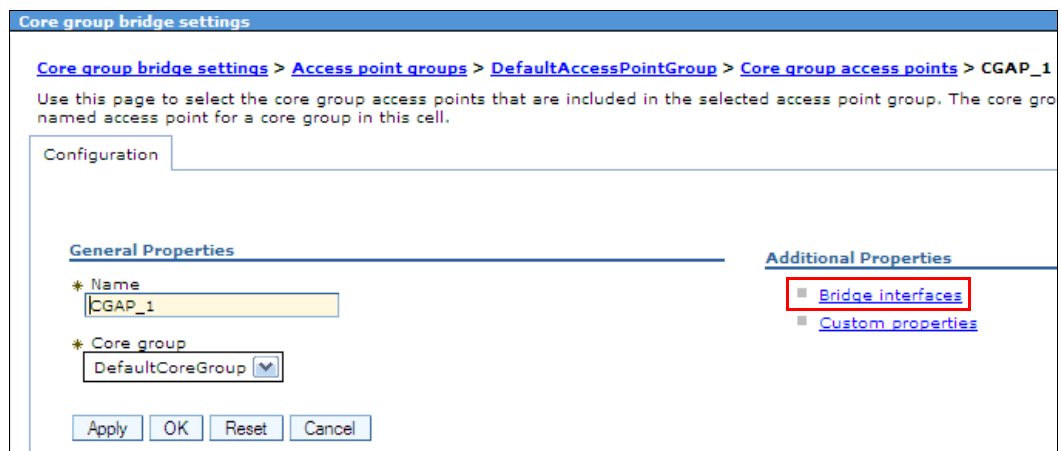


Figure 18-20 CGAP_1 Core group access point window

- Click **New** to create a new Bridge interface for the CGAP_1 Core group access point, as shown in Figure 18-21.

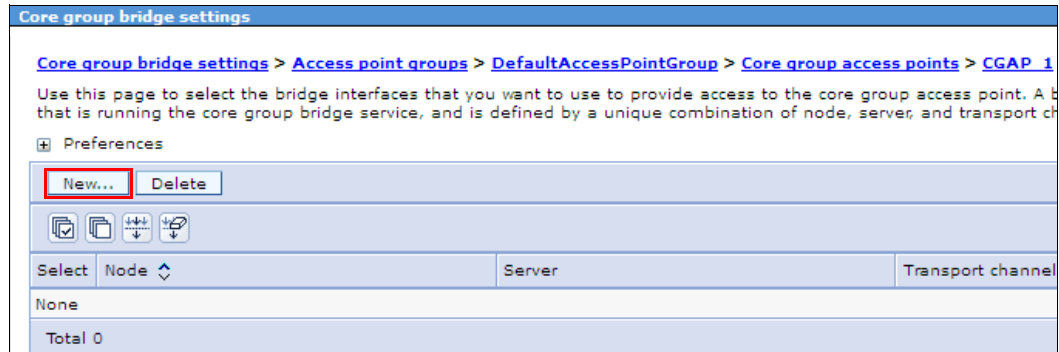


Figure 18-21 CGAP_1 Bridge interfaces window

- Select a process from the Bridge interfaces drop-down menu, as shown in Figure 18-22. Consider administrative processes first. Click **Apply**.

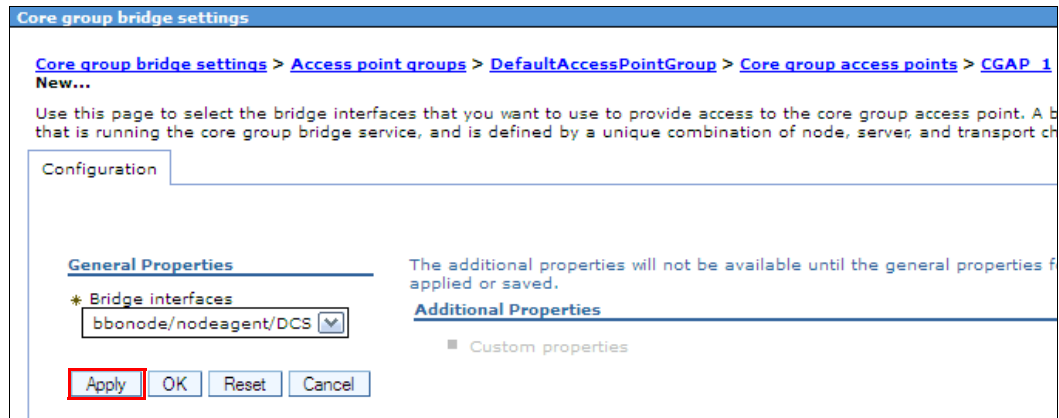


Figure 18-22 New Bridge interfaces window

- Click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
- For high-availability purposes, define one more bridge interface by repeating steps 7 to 9 for the first core group.

- Back on the Core group access points window, select **CGAP_2\Coregroup2**, and click **Show Detail**, as shown in Figure 18-23.

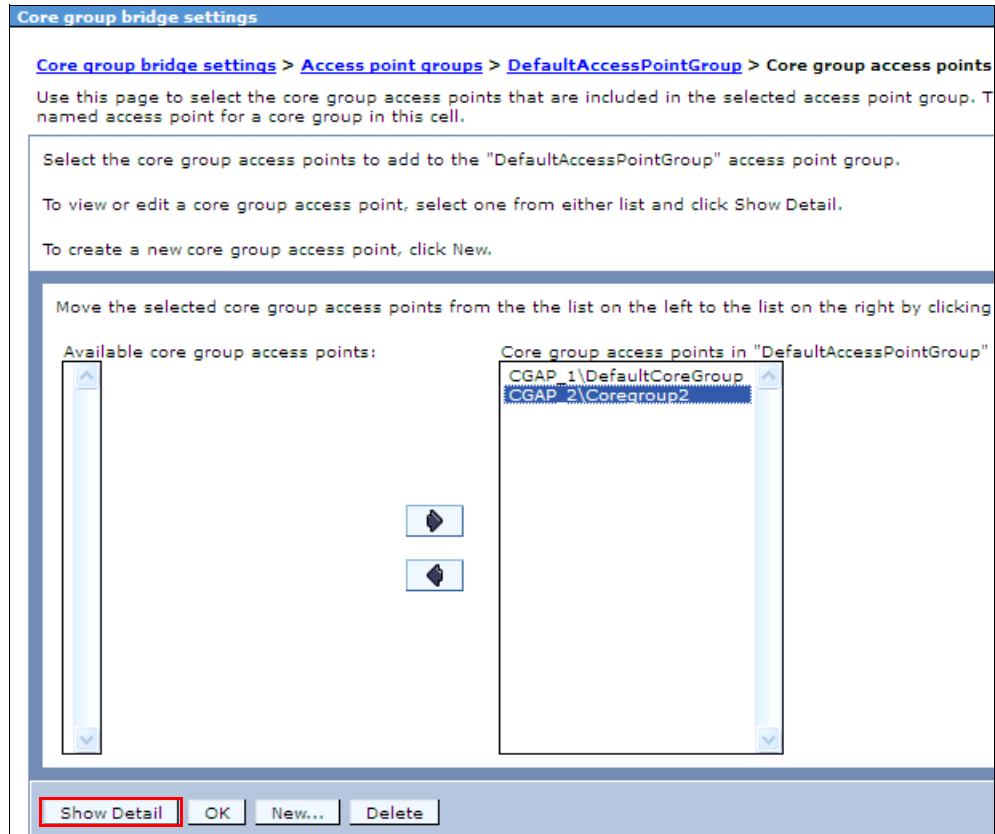


Figure 18-23 Core group access points window

- Select **Bridge interfaces** from Additional Properties, as shown in Figure 18-24.

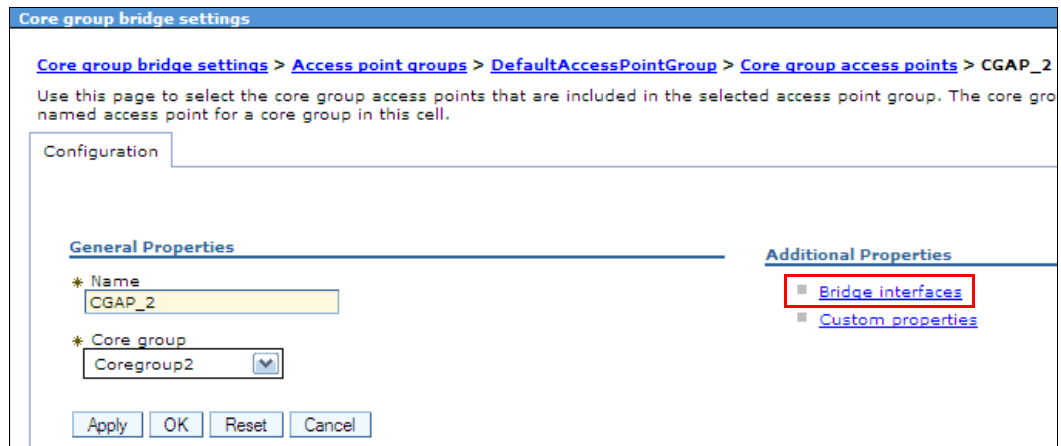


Figure 18-24 CGAP_2 Core group access point window

- Click **New** to create a new bridge interface for the CGAP_2 Core group access point, as shown in Figure 18-25.

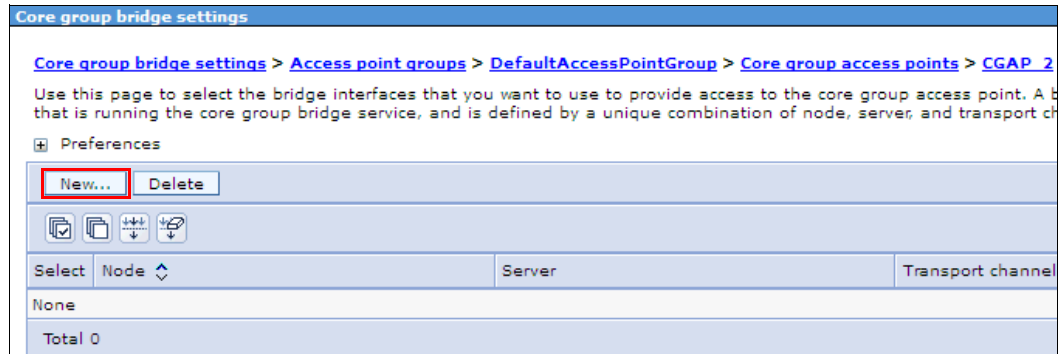


Figure 18-25 CGAP_2 Bridge interfaces window

- Select a process from the **Bridge interfaces** drop-down menu, as shown in Figure 18-26. Consider administrative processes first. Click **Apply**.

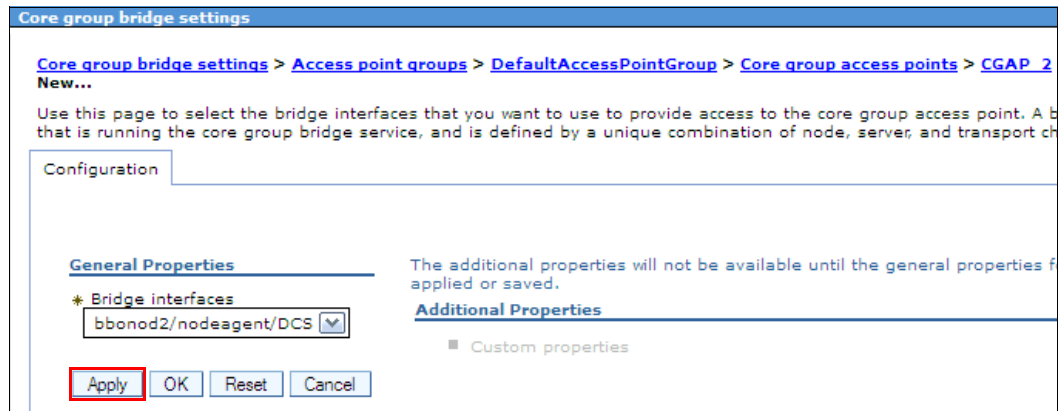


Figure 18-26 New Bridge interfaces window

- Click **Review** → **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
- For high-availability purposes, define one more bridge interface by repeating steps 13 to 15.

17. Verify your topology setup, as shown in Figure 18-27.

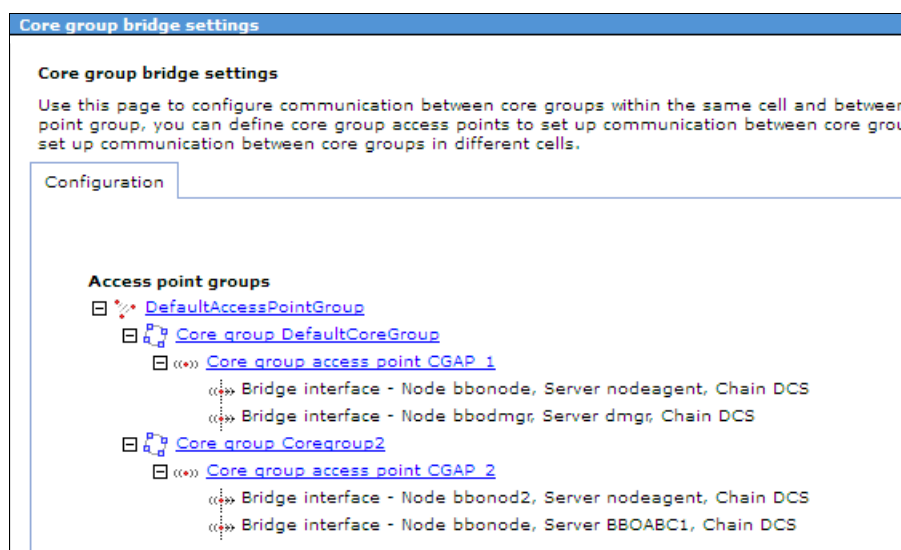


Figure 18-27 Access point groups overview

18. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
19. Fully shut down and then restart all core group bridges in the affected access point groups by clicking either **Servers** → **Server Types** → **WebSphere application servers** or **System administration** → **Node agents** or **System administration** → **Deployment manager**. (You must start the deployment manager manually from the system console, using the SDSF command prompt, or using your automation procedures.)

Complete a full shutdown: When you make a change in the core group bridge configuration, including the addition of a new bridge or the removal of an existing bridge, you must *fully shut down* and then restart all core group bridges in the affected access point groups.

Heap size note: If you have multiple large core groups bridged together, consider increasing the heap size of the bridge interfaces and high-availability coordinators by 512 MB (as a starting point) and fine-tune heap settings using verbose GC trace monitoring. To increase core group memory settings, consider setting the IBM_CS_DATASTACK_MEG and transport buffer size to 100.

Bridge interfaces: Do not create more than two bridge interfaces per core group or use productive servers as bridge interfaces. Ensure that these interfaces are on different nodes for high-availability purposes.

Use administrative processes, such as node agents, deployment manager, or, where possible, dedicated application servers. Process utilization can be high for core bridge interfaces during a core group start.

For more information about core group bridge service, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_ha_coregroupbridge.html

Setting up preferred coordinator servers

By default, the high availability manager elects the lexically lowest named core group process to be the coordinator. The name of the process consists of *cell name*, *node name*, and *process name*.

Because a coordinator takes up additional resources in the JVM, you might want to override the default election mechanism by providing your own list of preferred coordinator servers in the WebSphere Administrative Console.

To set preferred coordinator servers:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 18-28.

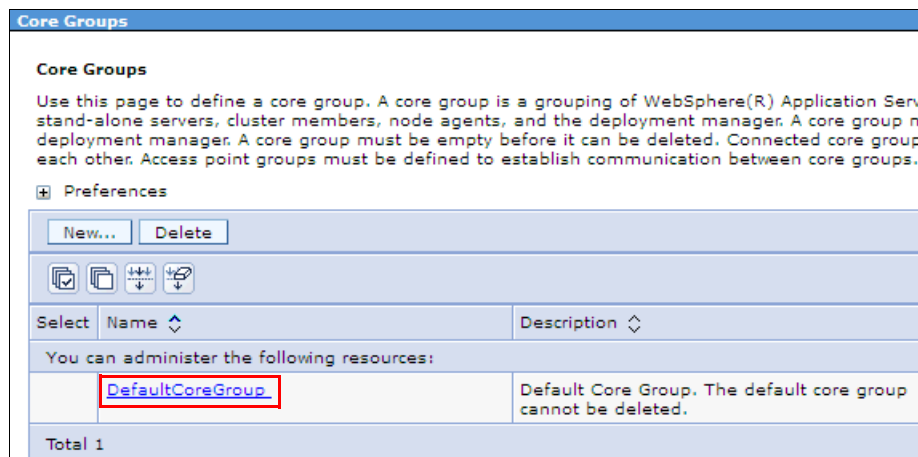


Figure 18-28 Selecting the core group

3. Click **Preferred coordinator servers** under Additional Properties, as shown in Figure 18-29.

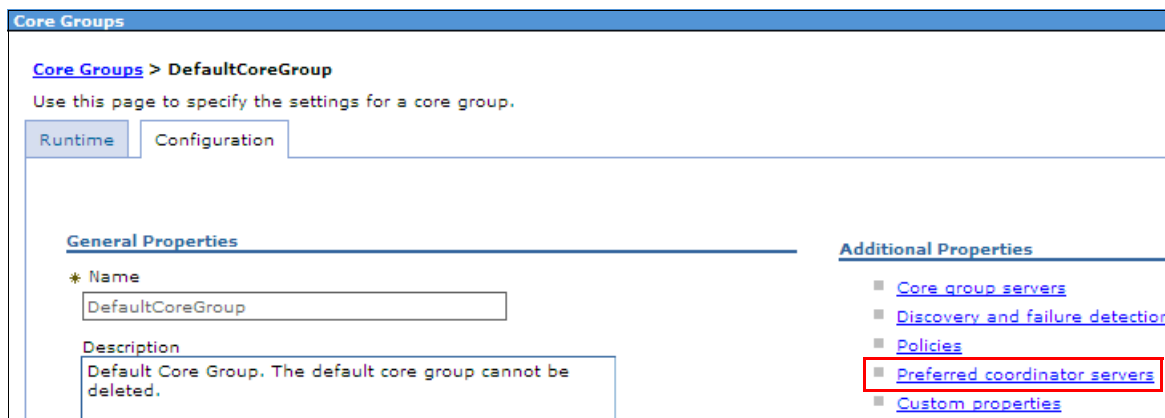


Figure 18-29 Specify preferred coordinator servers

4. Add the core group servers to the list of preferred coordinator servers, and organize their sequence, as shown in Figure 18-30. Click **OK**.

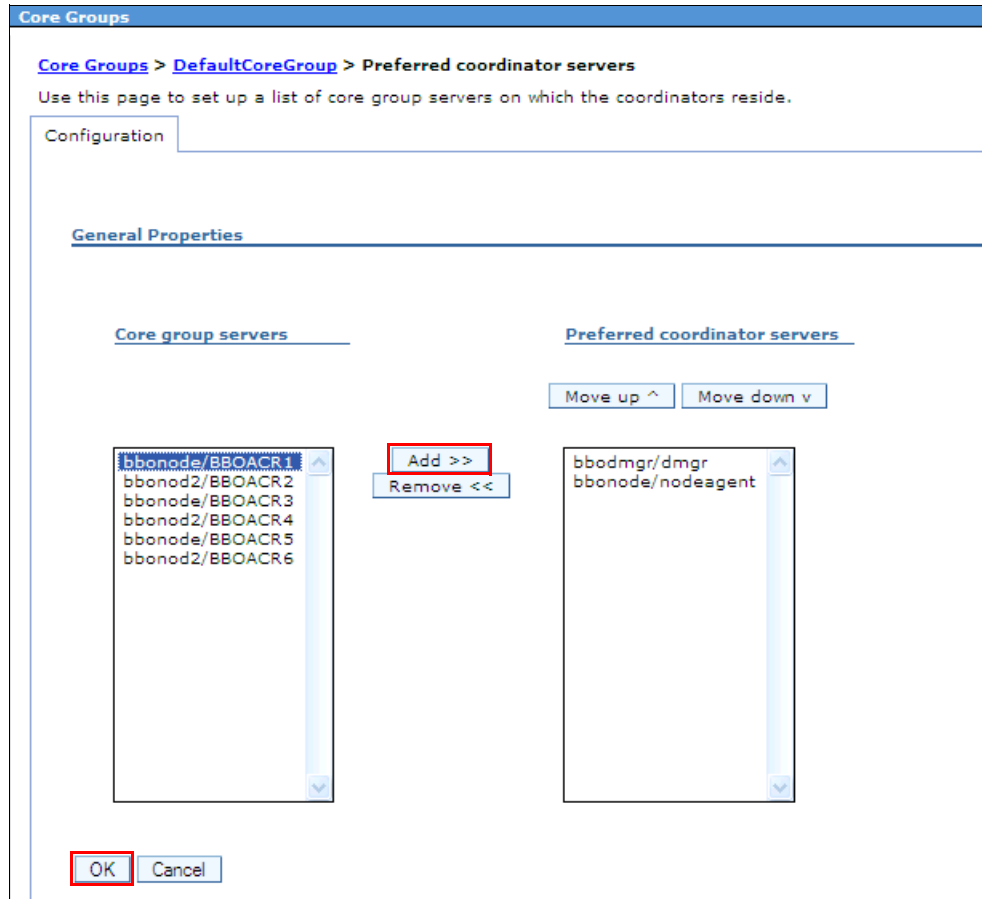


Figure 18-30 Specific core group preferred coordinator servers configuration window

5. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

DCS alternate protocol providers

In general, alternate protocol providers, such as the z/OS cross-system coupling facility (XCF)-based provider, uses less system resources than the default Discovery Protocol and Failure Detection Protocol, especially during times when the core group members are idle.

An alternate protocol provider generally uses less system resources because it does not perform the member-to-member TCP/IP pinging that the default protocol providers use to determine whether a core group member is still active. If you decide to use the z/OS XCF-based protocol provider, understand that at system start, the server process is joined as a member to an XCF group. The XCF group contains all of the active members for the core group. XCF provides notification to all of the members of this group when a member joins the group and when a member can no longer be contacted because the server shut down or because XCF determines that the server process has terminated.

To enable the high availability manager DCS signalling using XCF as alternate protocol:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 18-31.

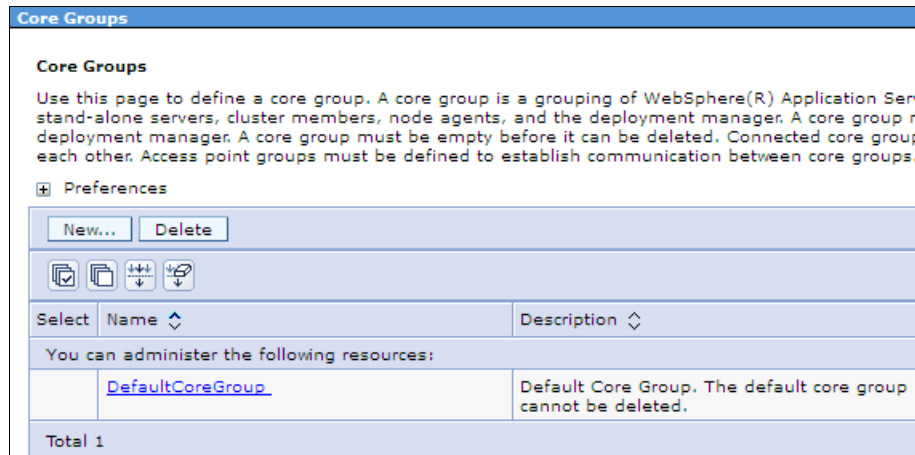


Figure 18-31 Selecting the core group

3. Click **Discovery and failure detection** from Additional Properties, as shown in Figure 18-32.

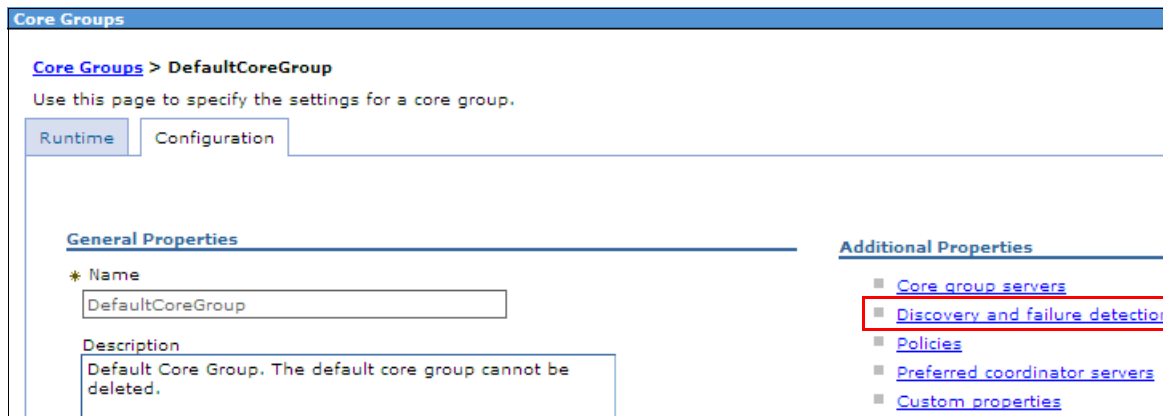


Figure 18-32 Discovery and failure detection

4. Select the **Use alternative protocol providers** option, and enter the class name `com.ibm.ws.xcf.groupservices.LivenessPluginZoSFactory`, as shown in Figure 18-33. Click **Apply**.

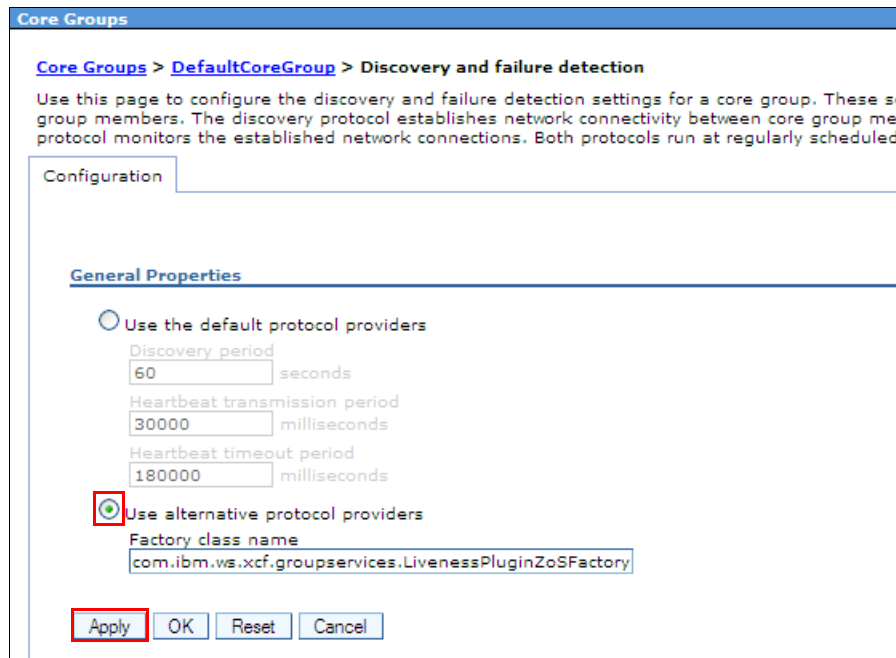


Figure 18-33 Discovery and failure detection window

5. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
6. Restart all of the core group members by clicking **Servers** → **Server Types** → **WebSphere application servers** or **System administration** → **Node agents** or **System administration** → **Deployment manager**. (You must start the deployment manager manually from the system console, using the SDSF command prompt, or using your automation procedures.)

Consideration: Ensure that the core group includes only servers of Version 7 or higher and that all the members are running on the same z/OS operating system (homogenous environment). Bridged core groups also need to adhere to this restriction because they need to use the same DCS protocol.

Verify that IBM VTAM® starts with the XCFINIT=YES option in ATCSTRxx to use XCF services.

For more information about XCF signalling, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_ha_cfg_alternate_protocol.html

18.2.3 High-availability policies and groups

High-availability groups are part of the high availability manager framework. A high-availability group provides the mechanism for building a highly-available component and enables the component to run in one of several different processes. A high-availability group cannot extend beyond the boundaries of a core group.

A high-availability group is associated with a specific component. The members of the group are the set of processes where it is possible to run that component. Therefore, a product administrator cannot directly configure or define a high-availability group and its associated set of members. Instead, high-availability groups are created dynamically at the request of the components for which they need to provide a highly-available function.

Every high-availability group has an associated policy. The policy determines which members of a high-availability group are active at a given point in time. The policies that are available for high-availability groups to use are stored as part of the core group configuration. The same policy can be used by several different high-availability groups, but all of the high-availability groups to which the policy applies must be part of the same core group.

Policy rules are applied in the following circumstances:

- ▶ A member joins or leaves a high-availability group
- ▶ The state of a member changes, for example, from idle to disabled

High-availability policies

To work with high-availability policies, complete the following steps:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 18-34.

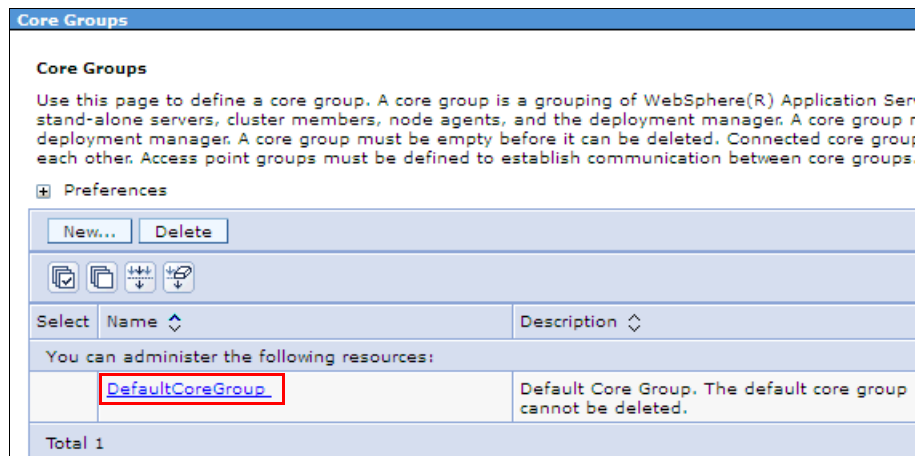


Figure 18-34 Select a core group

3. Click **Policies** from Additional Properties, as shown in Figure 18-35.

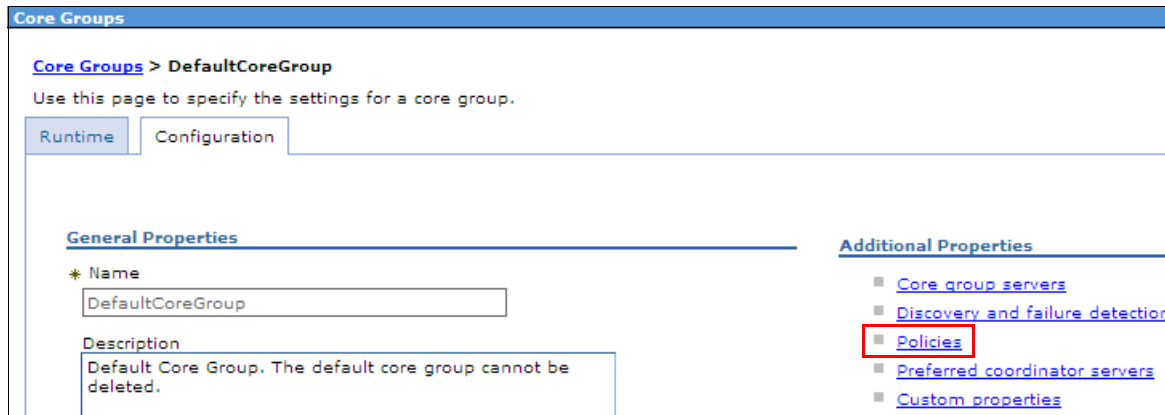


Figure 18-35 Policies

4. You can create new policies to use with components. Figure 18-36 shows the default available policies.

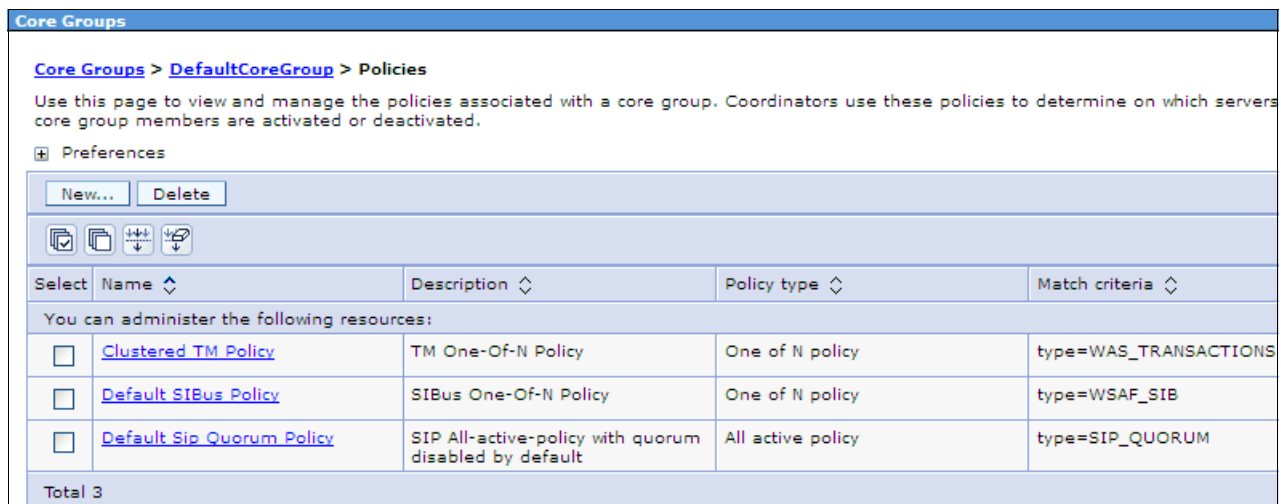


Figure 18-36 Policies window

Configuration changes: The high availability manager dynamically detects policy configuration changes. Therefore, policy setting changes go into effect as soon as you save and propagate these changes. You do not need to restart the server.

Policies note: Do not delete the IBM default provided policies or change the existing policy. Configure a new policy with more name=value pair matching. Ensure that the component that matches a policy supports that policy type.

At least one policy must match and that policy must be the policy with the most number of matches.

Do not configure identical match criteria for multiple policies in the same core group. If there is an ambiguous match, the high availability manager puts the high-availability group in an error state and does not make any of the group members active.

For instructions about how to create a new high-availability policy, go to the following website:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_ha_newpolicy.html

High availability groups

To view high availability groups:

1. Click **Servers** → **Core Groups** → **Core group settings**.
2. Select your desired core group, as shown in Figure 18-37.

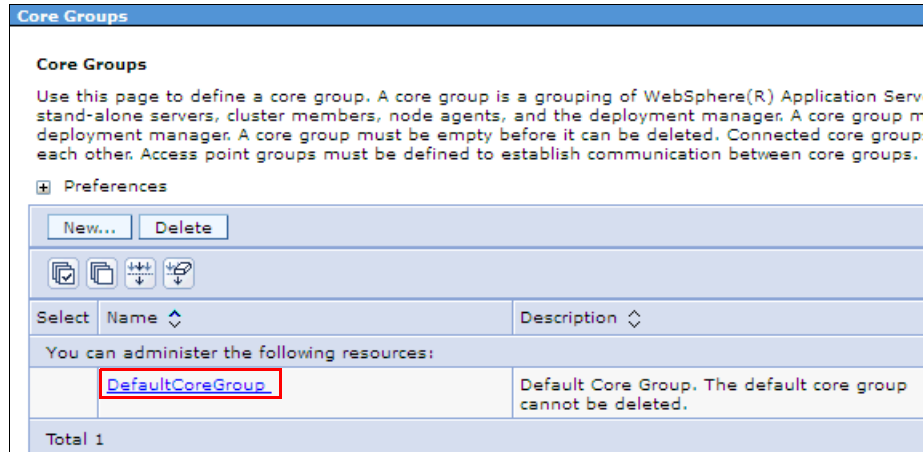


Figure 18-37 Select a core group

3. Go to the **Runtime** tab, as shown in Figure 18-38.

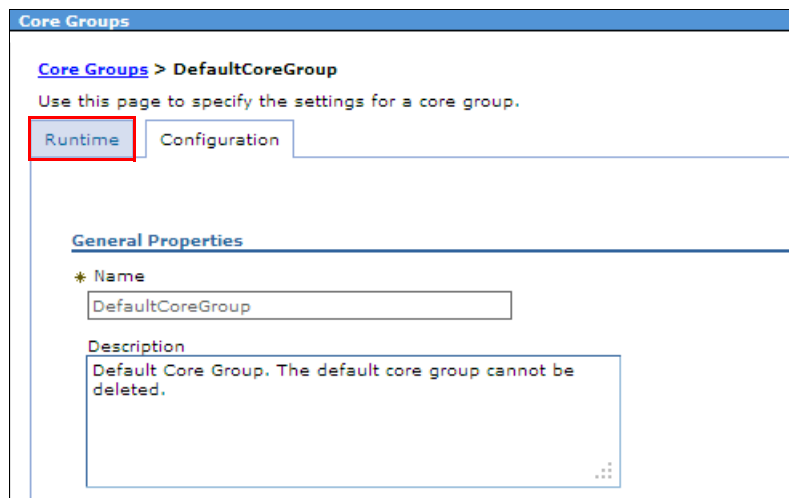


Figure 18-38 Runtime properties

4. Click **Show servers** to display core group servers that are hosting active high-availability group members. To display high-availability groups, specify the name=value pairs or * in the Group name properties field, and click **Show groups**, as shown in Figure 18-39.



Figure 18-39 Group name properties

18.3 Failover and failback

Failover is one of the techniques of fault tolerance. It is used to make the system to some degree survive faults and failures.

In this section, we discuss WebSphere Application Server for z/OS failover capabilities.

18.3.1 High availability and failover of singletons

Singleton failover is a cluster-based service. High availability manager has to be enabled on all the members. Failover and subsequent failback depend on the options selected in the high availability service policy. The Preferred servers list option is provided to allow the service to run exclusively on the servers in the list in a preferred order. External resources and any other dependencies must be available to all the members that are to run the service in case of a failure of the primary. External clustering software can be used to complement the failover and failback processing.

WebSphere Application Server provides failover and recovery for the following singleton services:

- ▶ Transaction service
- ▶ Service integration bus

Transaction service

Transaction service peer recovery processing can only take place between members of the same server cluster. The Transaction Manager supports three different high-availability policies to achieve the recovery of transaction logs in a highly-available manner:

- ▶ One of N policy

This policy is the default style of peer recovery initiation. If an application server fails, the high availability manager selects another server to perform peer recovery processing on behalf of the failed server.

► Static policy

This style of peer recovery must be explicitly configured. If an application server fails, the operator can use the Administrative Console to select another server to perform the recovery processing.

► No Operation policy

This style of peer recovery must be explicitly configured. It indicates that external clustering software is monitoring the Transaction Manager and will fail over to an application server that is configured by the external clustering software to perform the recovery processing.

For more information about transactional high availability, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcjta_trans_ha.html

To enable transaction service log recovery, complete the following steps:

1. Click **Servers** → **Server types** → **WebSphere application servers**.
2. Select your desired application server, as shown in Figure 18-40.

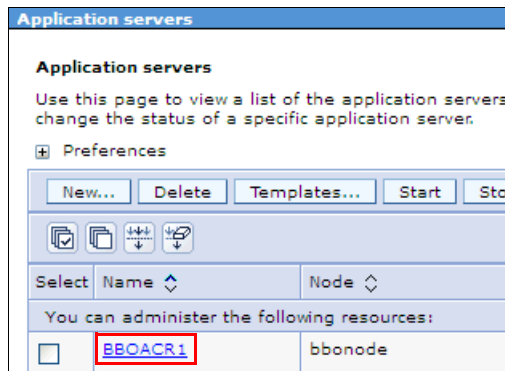


Figure 18-40 Select the application server

3. Under the Container Settings section, expand **Container Services** and then click **Transaction service**, as shown in Figure 18-41.

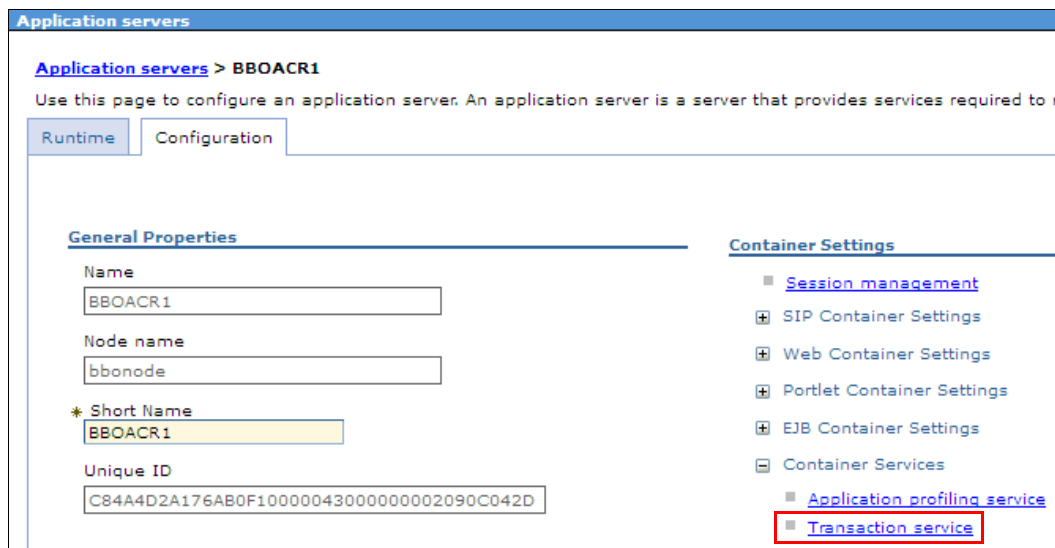


Figure 18-41 Transaction service

- Specify the **Transaction log directory** value, as shown in Figure 18-42. The directory specified must be unique in the cluster and accessible to all the cluster members. It must be allocated on a fast disk.

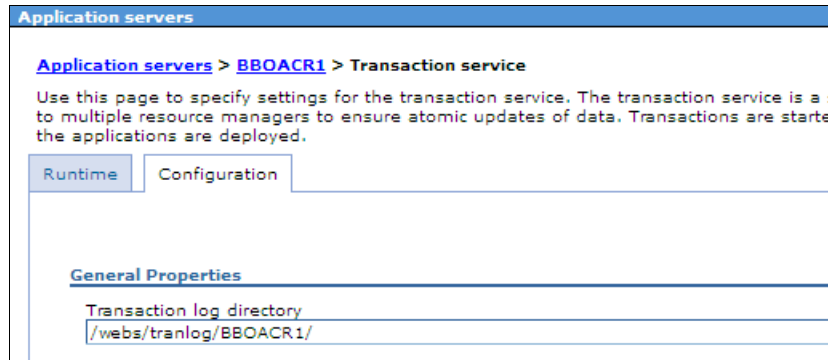


Figure 18-42 Specify the Transaction log directory

- Click **Apply**.
- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
- Repeat steps 2 to 6 for all the cluster members.
- Click **Servers** → **Clusters** → **WebSphere application server clusters**.
- Select your desired application server cluster, as shown in Figure 18-43.

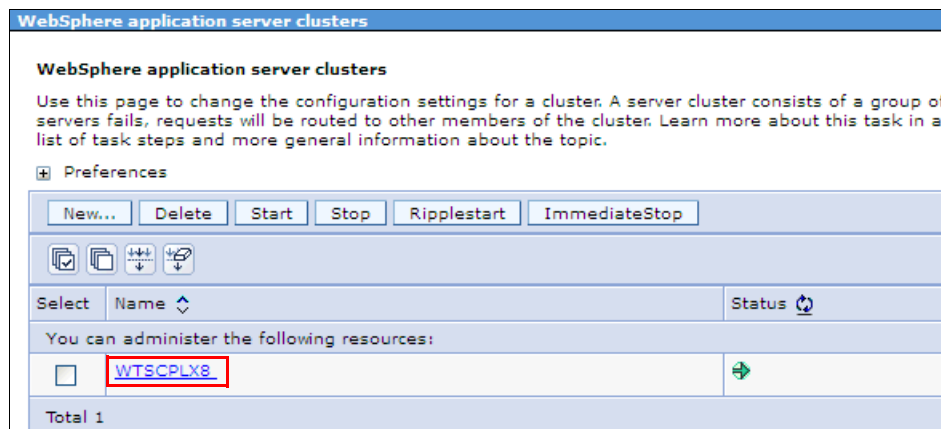


Figure 18-43 Select application server cluster

10. Enable the **Enable failover of transaction log recovery** option, as shown in Figure 18-44. Click **Apply**.

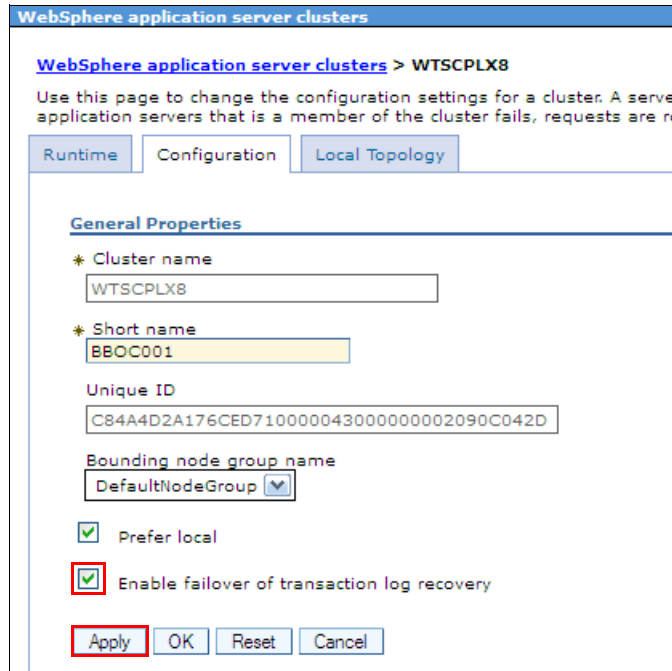


Figure 18-44 Enable failover

11. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
12. Restart the affected servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Service integration bus

For bus messaging engines, use a policy type of One of N. Thus, although the messaging engine can be defined on every server in the cluster, the high availability manager ensures that it is active only on one of the servers in the group and will always be active on one of the servers as long as one is available.

For more information about making the SIBus highly available, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcjt0010_.html

To configure a service integration bus (SIBus) with data store persistence:

1. Click **Service integration** → **Buses**.

2. Click **New** to create a new SIBus, as shown in Figure 18-45.

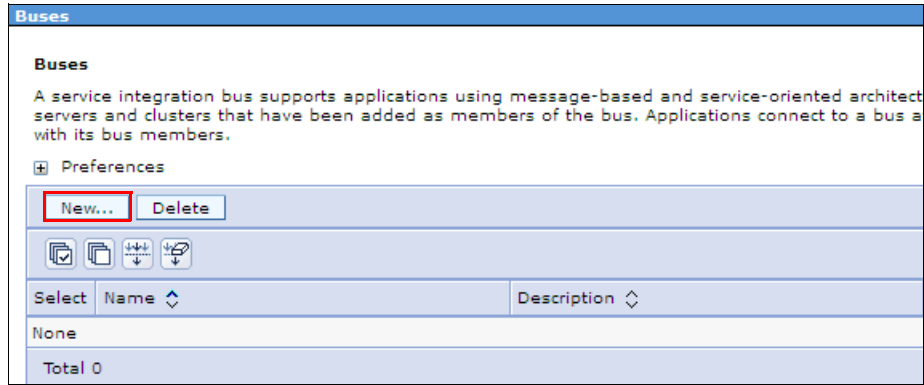


Figure 18-45 New bus

3. Enter the name for the new bus and indicate the bus security, as shown in Figure 18-46. Click **Next**.

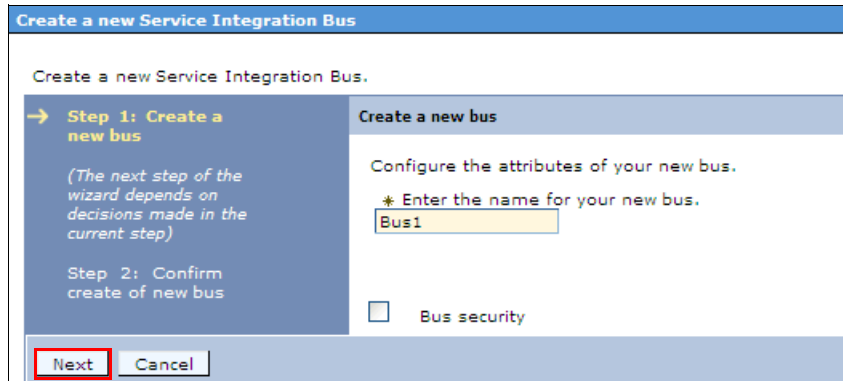


Figure 18-46 Create a new bus

4. Confirm the information and then click **Finish**, as shown in Figure 18-47.

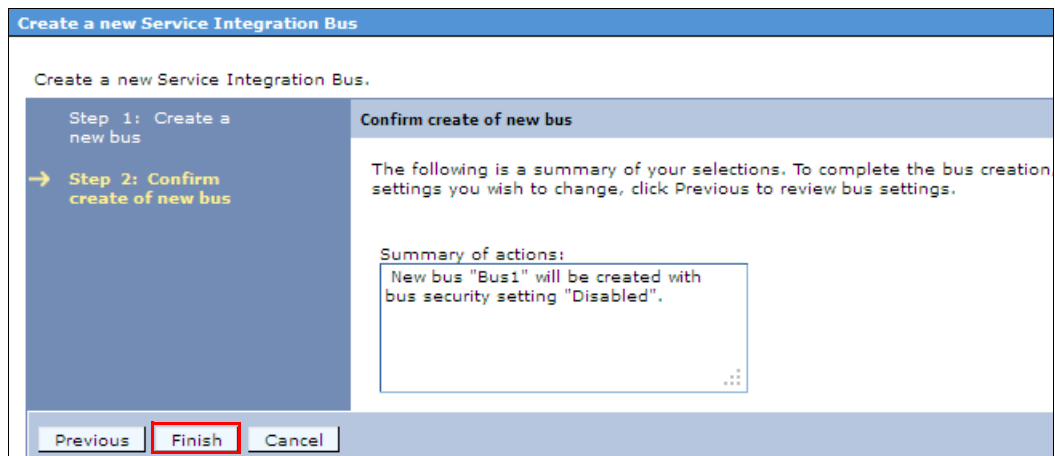


Figure 18-47 Confirm the creation of a new bus

- Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes** and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
- Select the newly created SIBus, as shown in Figure 18-48.

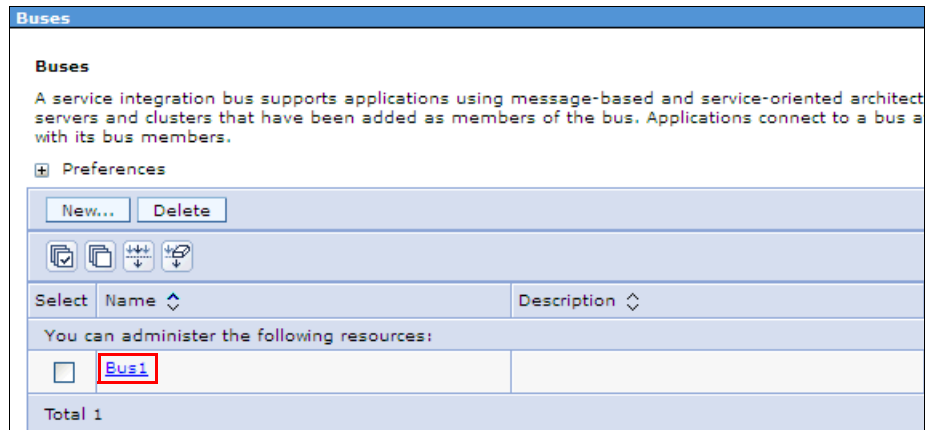


Figure 18-48 Buses view

- In the Topology section, select **Bus members**, as shown in Figure 18-49.

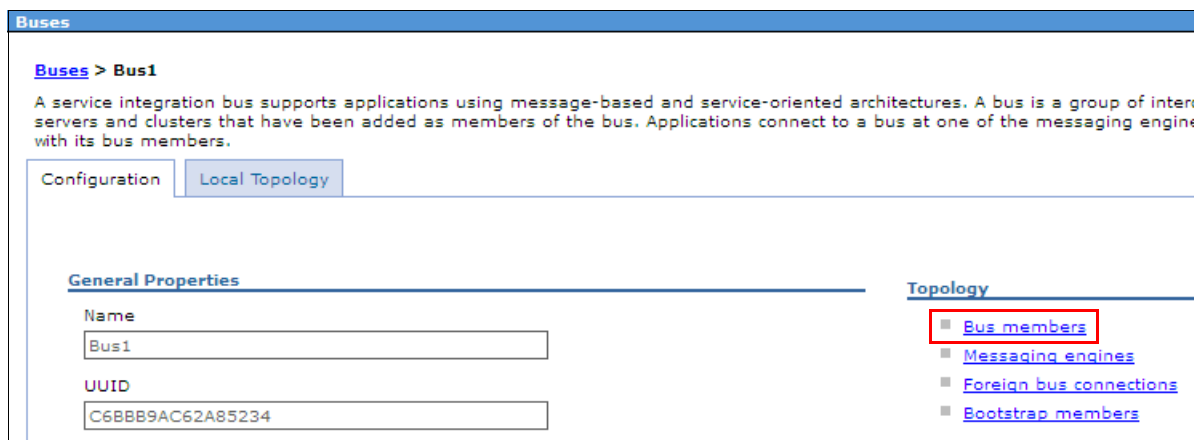


Figure 18-49 Bus members

- Click **Add** to create a new bus member, as shown in Figure 18-50.

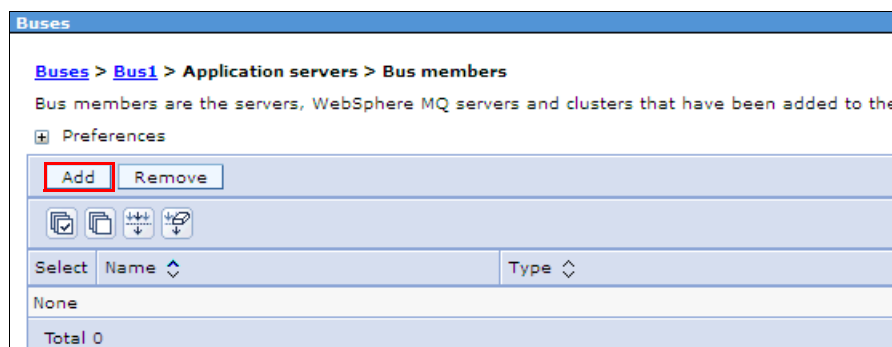


Figure 18-50 Add a new bus member

- Select the **Cluster** option and then select the cluster name from the drop-down menu, as shown in Figure 18-51. Click **Next**.

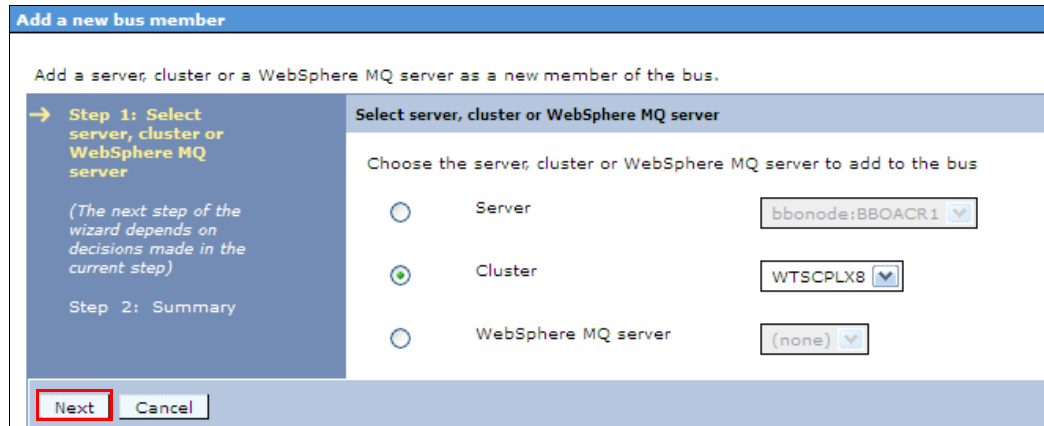


Figure 18-51 Select server, cluster, or WebSphere MQ server window

- Ensure that the **Messaging engine policy assistance settings** option is enabled and then select the **Custom** policy type, as shown in Figure 18-52. Click **Next**.

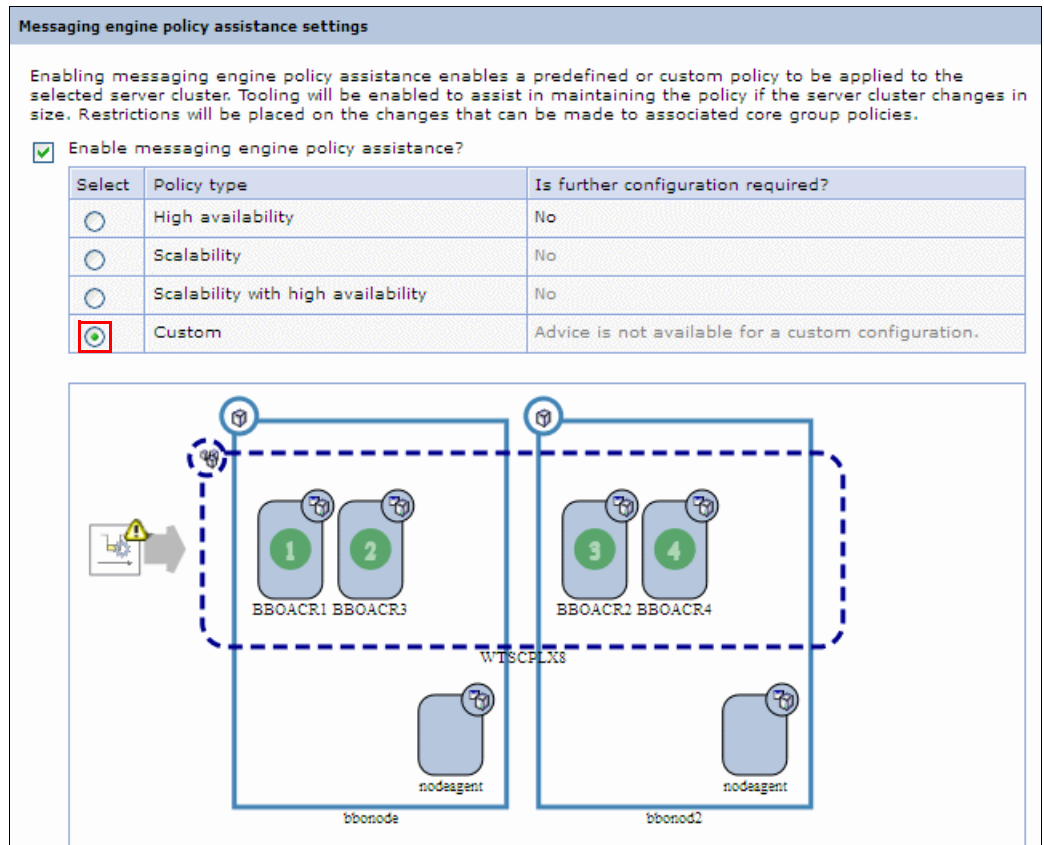


Figure 18-52 Select the Custom messaging engine policy assistance

11. Select the **Data store** option, as shown in Figure 18-53.

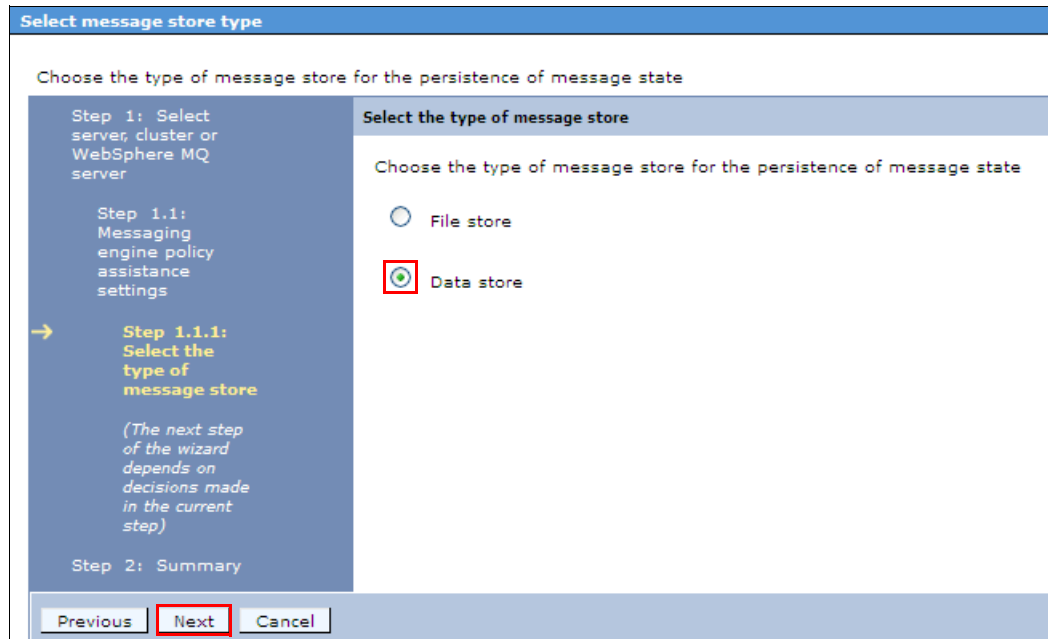


Figure 18-53 Select Data store type of message store

12. Click **Add** to create a new messaging engine, as shown in Figure 18-54.

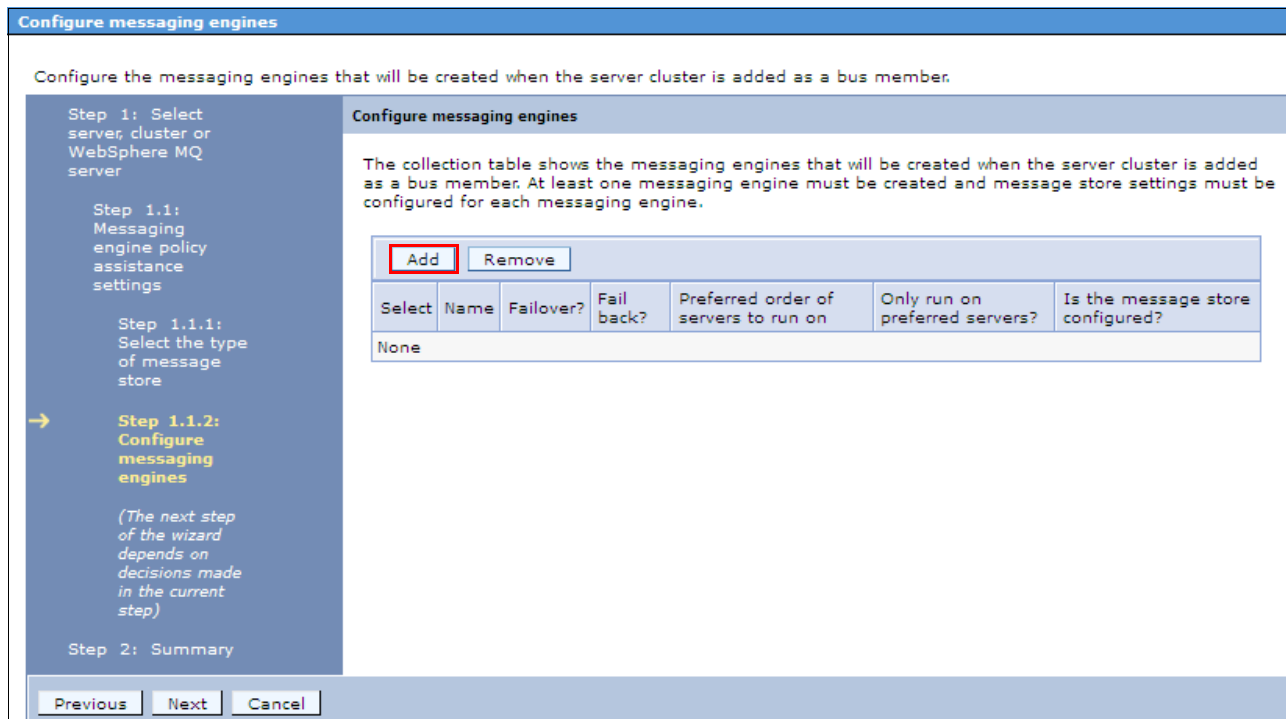


Figure 18-54 Add a new messaging engine

13. We chose to enable failback and failover of the messaging engine by selecting the following options:
- The messaging engine fails over to other servers in the server cluster.
 - The messaging engine fails back to a preferred server if one is available (as specified in the preferred servers list).

Move suitable servers from the available servers list to the preferred servers list by selecting them and clicking the **Add** button. When you are done, click **Next**, as shown in Figure 18-55.

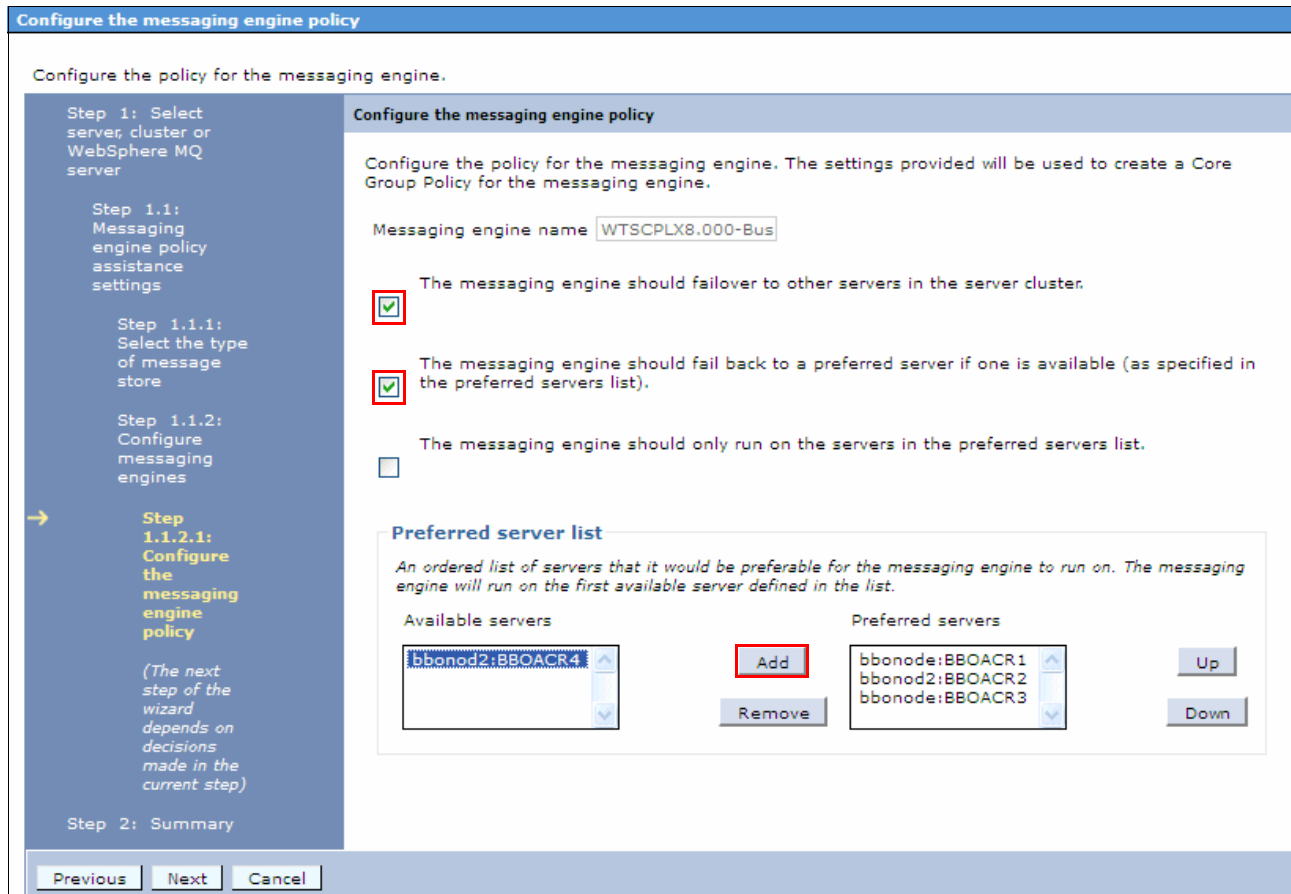


Figure 18-55 Configure the messaging engine policy

14. Enter the Data source JNDI name and Schema name, and select an Authentication alias. Clear **Create tables**, and click **Next**, as shown in Figure 18-56.

The screenshot shows a wizard window titled "Configure a data store". The main heading is "Configure the properties for a data store". On the left, a vertical list of steps is shown, with "Step 1.1.2.2: Specify data store properties" highlighted and an arrow pointing to it. The right pane is titled "Specify data store properties" and contains the following fields and options:

- "Specify the properties for the data store"
- "* Data source JNDI name" with a text input field containing "jdbc/DB9B_T4"
- "Schema name" with a text input field containing "IBMWSSIB"
- "Authentication alias" with a dropdown menu showing "(none)"
- An unchecked checkbox labeled "Create tables"

At the bottom of the wizard, there are three buttons: "Previous", "Next" (which is highlighted with a red box), and "Cancel".

Figure 18-56 Specify data store properties

15. Verify the messaging engine configuration, and click **Next**, as shown Figure 18-57.

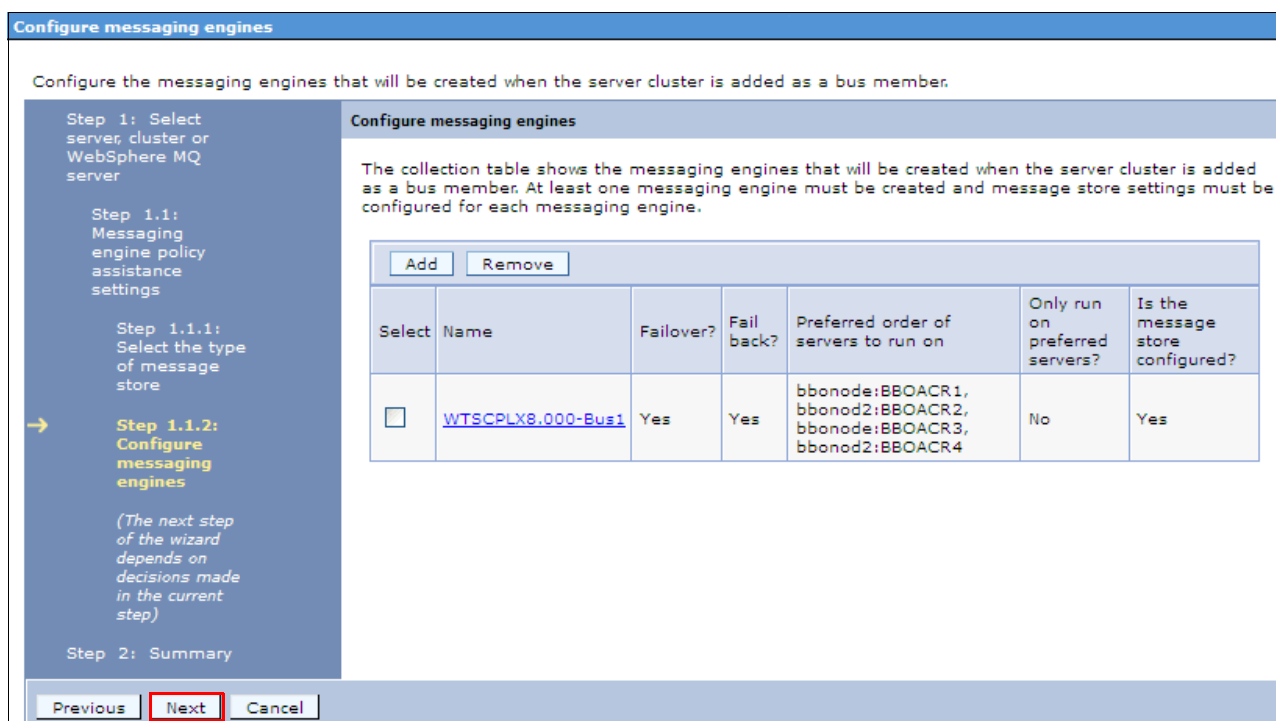


Figure 18-57 Configure messaging engines

16. Click **Finish**, as shown in Figure 18-58.

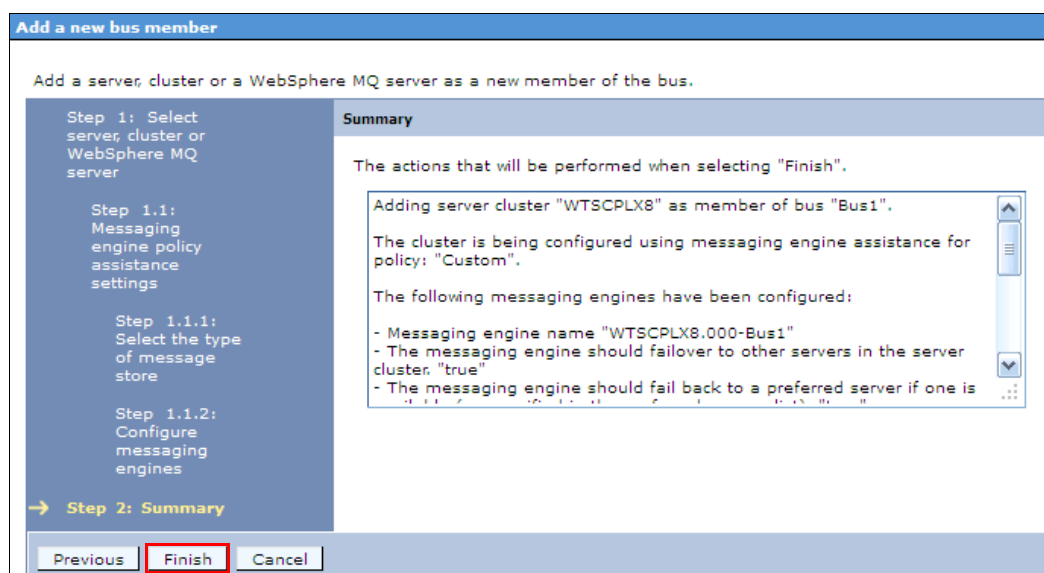


Figure 18-58 Summary window

17. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

18. Restart the affected servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

SIBus note: SIBus data store tables on z/OS have to be created manually using the `sibDDLGenerator.sh` script, as shown in the Example 18-3.

Example 18-3 sibDDLGenerator.sh DB2 for z/OS example

```
${WAS_INSTALL_ROOT}/bin/sibDDLGenerator.sh -system db2 -version 9.1 -platform zos  
-schema schemaname -user userid -create -database databasename -storagegroup  
storagename -statementend ";"
```

To configure external high-availability frameworks for messaging engine with a “No operation” policy, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftjt0031_.html

WebSphere Application Server V8.5 provides a feature to reconnect to a standby gateway MQ queue manager when an active queue manager fails or becomes unavailable. For information about high availability of messaging engines that are connected to WebSphere MQ, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcjt0003_.html

For information about how to create a WebSphere MQ link between SIB and WebSphere MQ queue manager, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftjc0002A_.html

18.3.2 Data replication domains

Replication domains are used for replication by the HTTP session manager, dynamic cache service, and stateful session bean failover components. All components that need to share information must be in the same replication domain. Use different replication domains for each type of consumer. You can use same replication domain for HTTP sessions and stateful session beans. Configuring one replication domain in this case ensures that the backup state information is located on the same backup application servers. Cache replication must use separate replication domain.

To create a new replication domain:

1. Click **Environment** → **Replication domains**.

2. Click **New** to create a new replication domain, as shown in Figure 18-59.

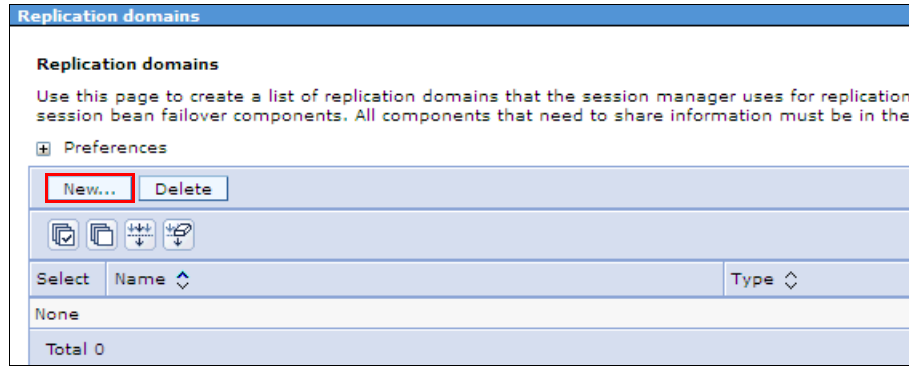


Figure 18-59 New Replication domains

3. Specify the Name, Request timeout value, and the Number of replicas, as shown in Figure 18-60. Click **Apply**.

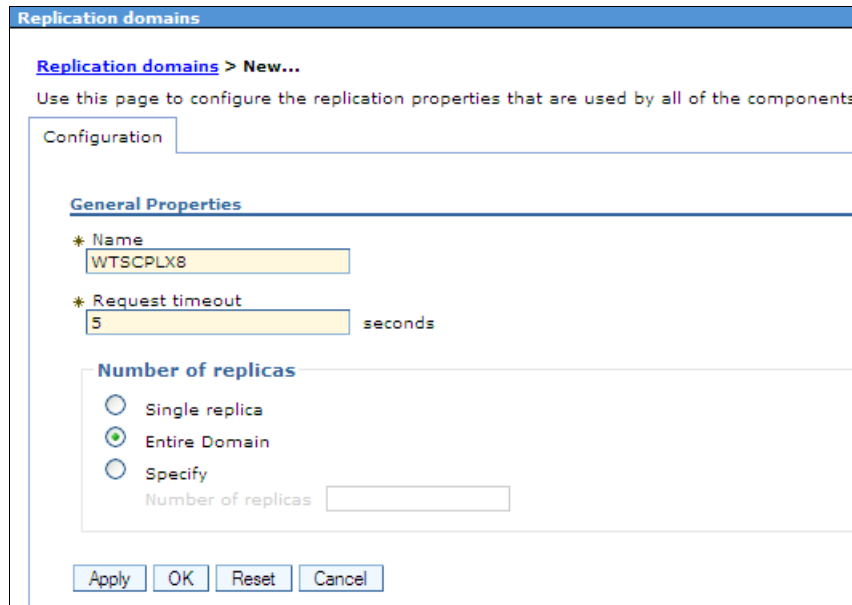


Figure 18-60 Replication domain configuration

4. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
5. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

For more information about data replication, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcrun_drs_replication.html

18.3.3 Session management replication

If the session persistence database is hosted on local data sharing DB2 for z/OS, the performance is comparable to or better than memory-to-memory replication scheme.

To enable memory-to-memory session data replication in the cluster:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select your desired application server, as shown in Figure 18-61.

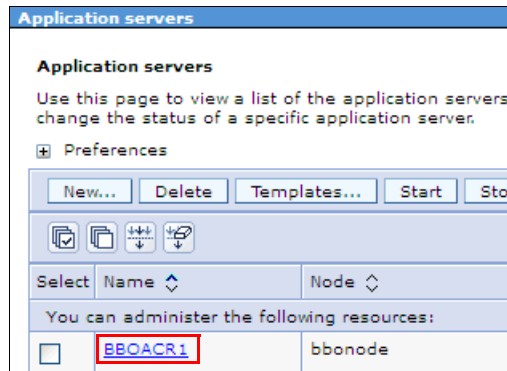


Figure 18-61 Specify the application servers

3. In the Container Settings section, select **Session management**, as shown in Figure 18-62.

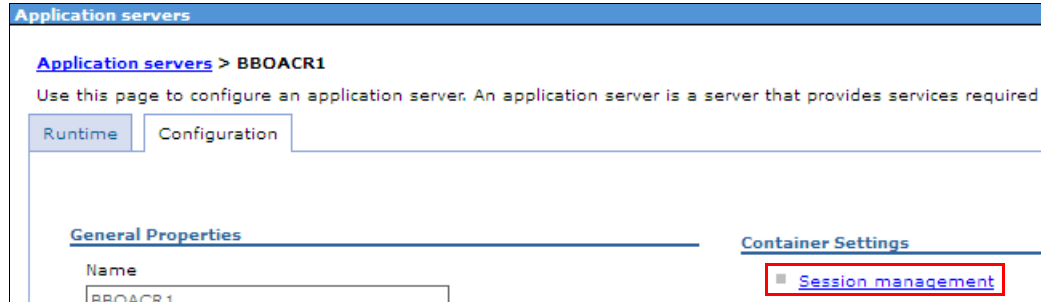


Figure 18-62 Session management

4. Under Additional Properties, select **Distributed environment settings**, as shown in Figure 18-63.

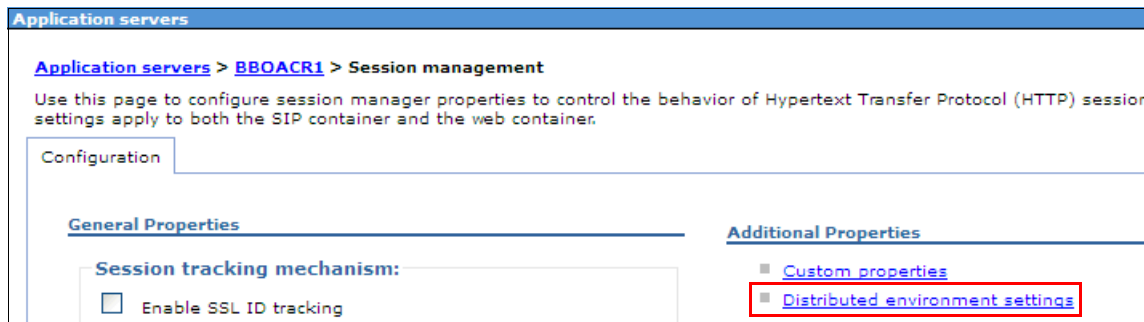


Figure 18-63 Distributed environment settings

5. Click **Memory-to-memory replication**, as shown in Figure 18-64.

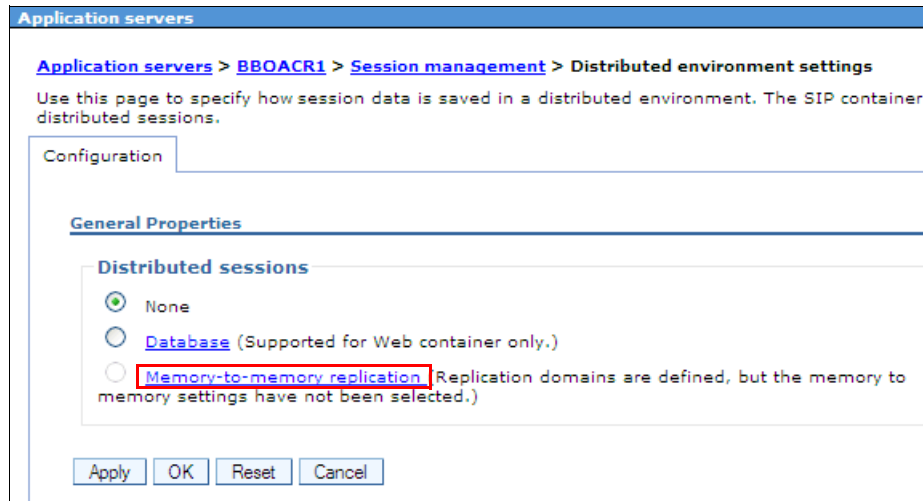


Figure 18-64 Memory-to-memory replication

6. Select the desired Replication domain and Replication mode. Click **Apply**, as shown in Figure 18-65.

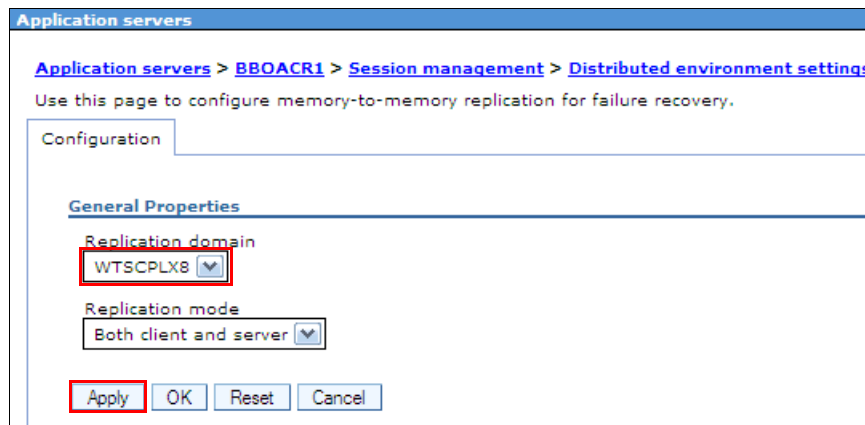


Figure 18-65 Memory-to-memory replication configuration

7. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
8. Repeat enablement for every application server in the cluster.
9. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

18.3.4 EJB stateful session bean replication

To enable stateful EJB sessions replication:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select the desired application server, as shown in Figure 18-66 on page 689.

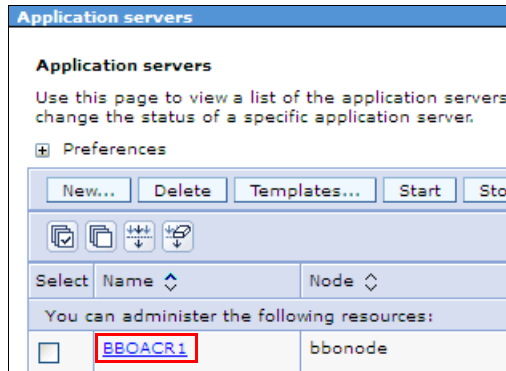


Figure 18-66 Select the application server

3. In the Container Settings section, expand **EJB Container Settings** and then select **EJB container**, as shown in Figure 18-67.

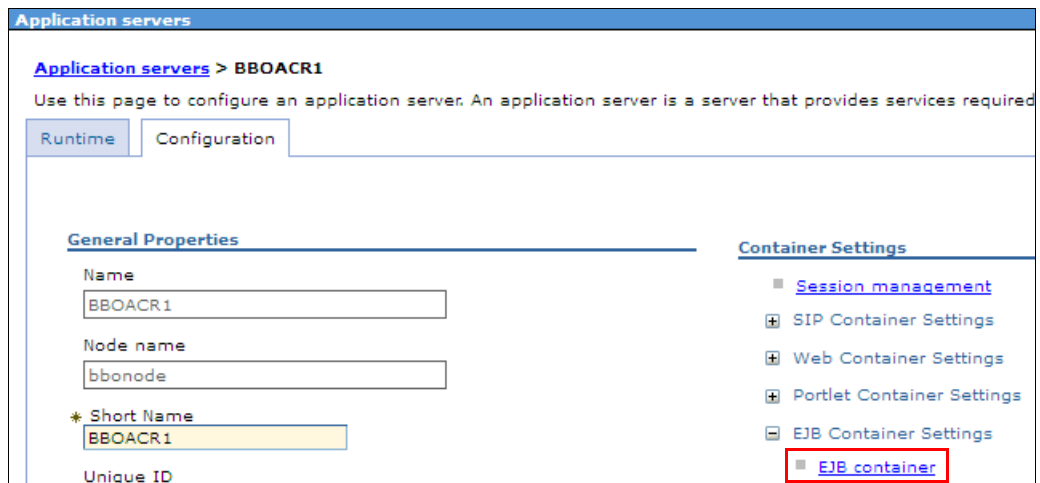


Figure 18-67 EJB container

4. Click the **memory-to-memory replication** link, as shown in Figure 18-68.

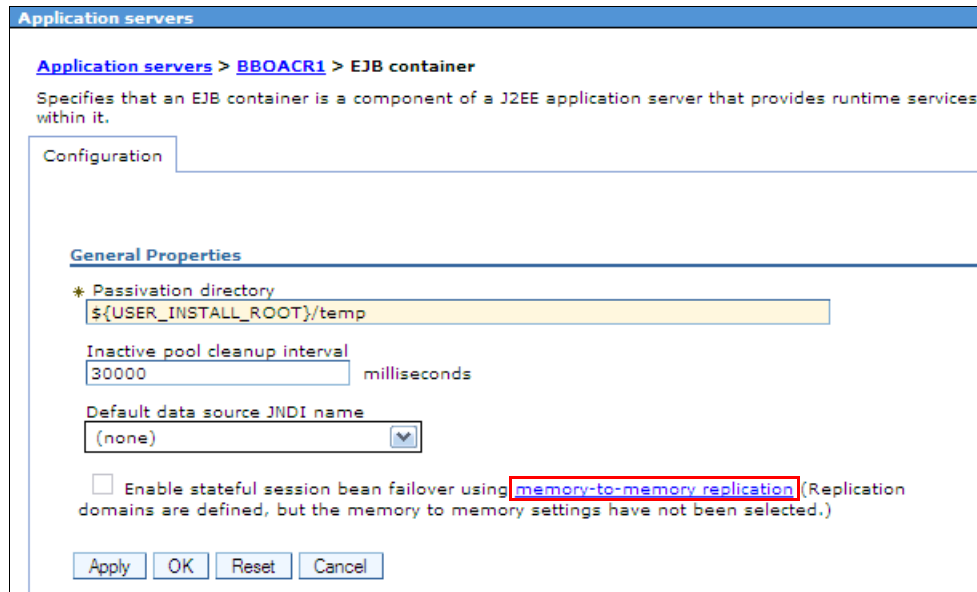


Figure 18-68 Select memory-to-memory replication link

5. Select the desired Replication domain and Replication mode. Click **Apply**, as shown in Figure 18-69.

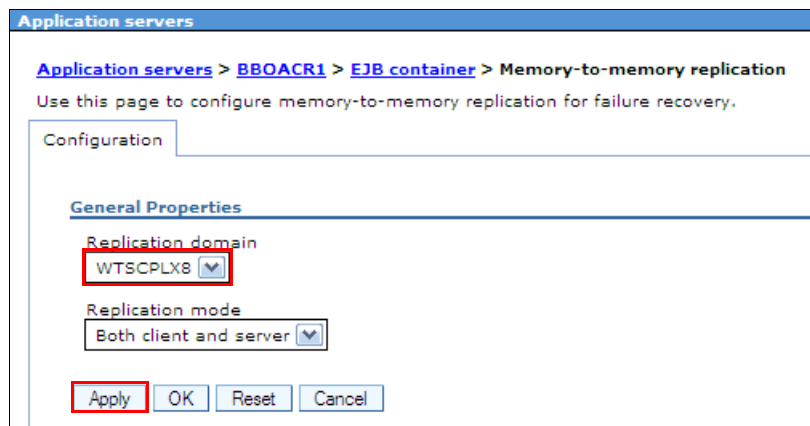


Figure 18-69 Memory-to-memory replication configuration

6. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

7. In the EJB container window, click **Apply**, as shown in Figure 18-70.

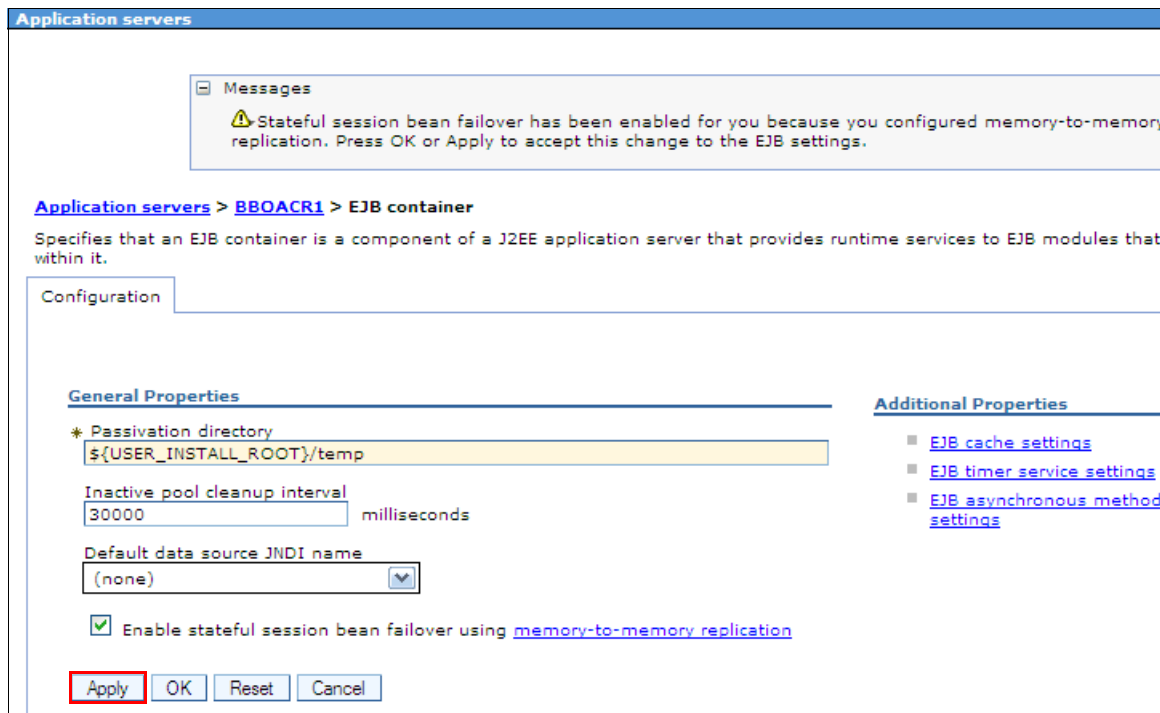


Figure 18-70 EJB container window

8. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
9. Repeat enablement for every application server in the cluster.
10. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Attention: Remember that with failover enabled, your application must never use both a local (EJBLocalObject) and remote (EJBObject) reference to the same stateful session bean instance.

To enable failover of only some stateful session beans, override the global EJB container settings at either the application or EJB module level.

For more information about Stateful session bean failover for the EJB container, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Fae%2Fcejb_sfsbf.html

To learn how to enable servant failover of EJB container in an unmanaged server, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftejb_sfsbfzos.html

18.3.5 Cache replication

You can make cache elements highly available by using memory-to-memory replication. To enable cache replication in WebSphere Application Server:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select your desired application server, as shown in Figure 18-71.

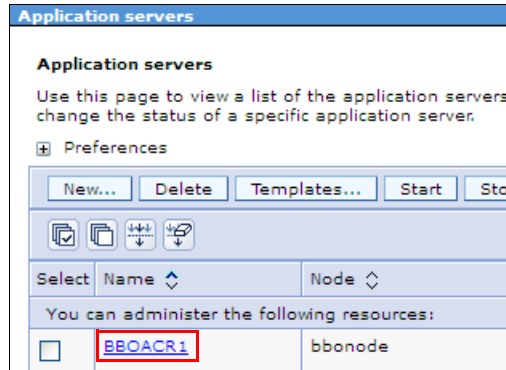


Figure 18-71 Select the application server

3. In the Container Settings section, expand **Container Services** and then select **Dynamic cache service**, as shown in Figure 18-72.

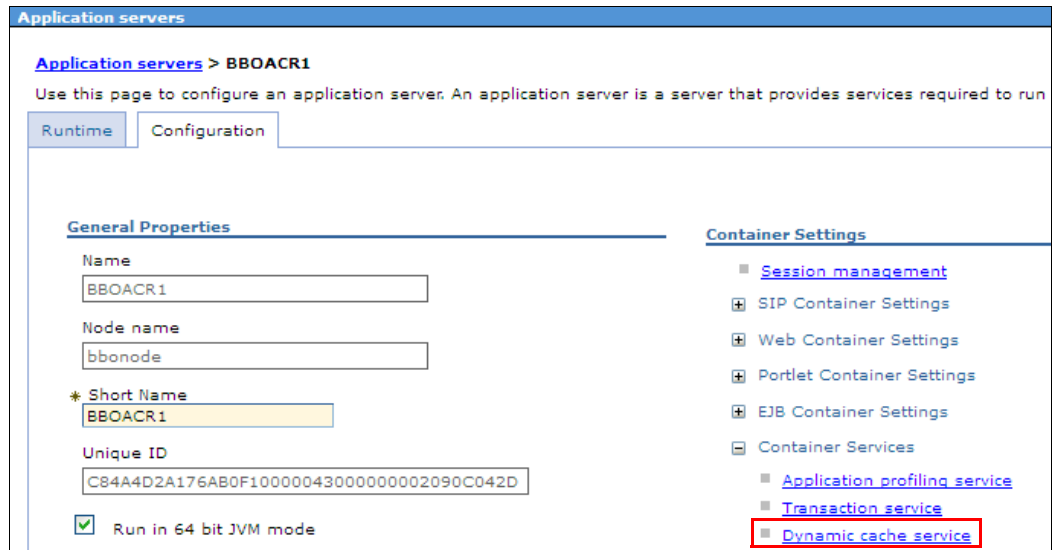


Figure 18-72 Dynamic cache service

4. Select the **Enable cache replication** option and then select the desired Replication domain and Replication type. Click **Apply**, as shown in Figure 18-73.

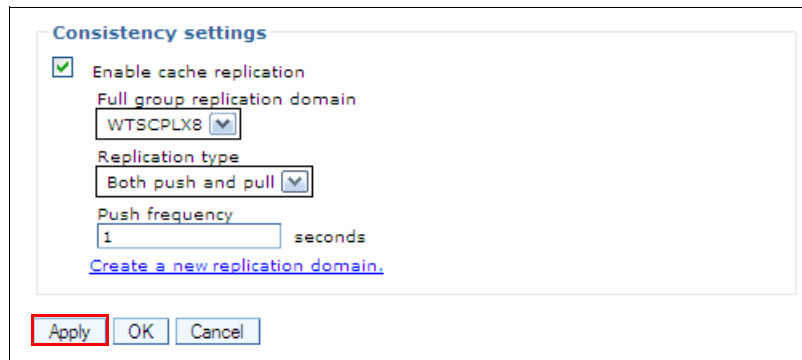


Figure 18-73 Consistency settings

5. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.
6. Repeat enablement for every application server in the cluster.
7. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Summary: To summarize the sharing policy, it is important to understand that the push only replication type is good for workloads with a high probability of other clusters handling cache hits. The both push and pull replication type is good for unknown or variable probability of cross-cluster cache hits.

The “Number of replicas” property for any replication domain that is used by the dynamic cache service must be set to **Entire domain**.

To learn how to enable cache replication on a single server in a non-clustered environment, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftdyn_cachereplication.html

18.3.6 Resource workload routing

The resource workload routing feature works for JCA J2C connection factories, including WebSphere optimized local adapters and JDBC data sources. It includes failover and subsequent failback from a predefined alternate or backup resource. The same enablement procedure for resource workload routing applies for both JCA and JDBC resources.

To enable resource workload routing for JDBC Data source:

1. Click **Resources** → **JDBC** → **Data sources**.

2. Select your desired data source, as shown in Figure 18-74.

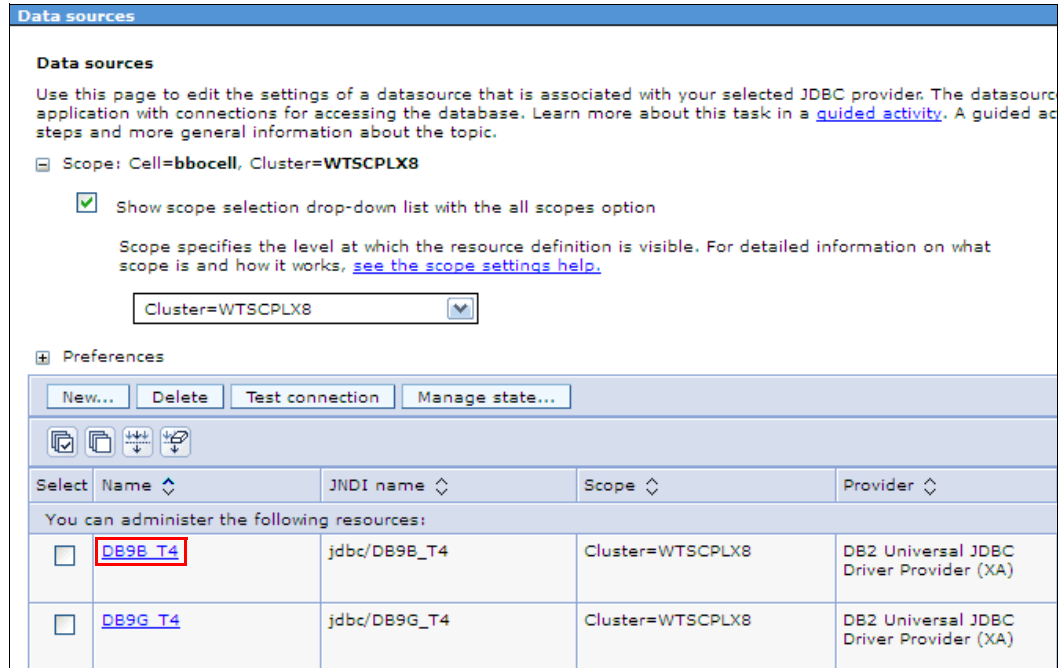


Figure 18-74 Select data source

3. In the Additional Properties section, select **Connection pool properties**, as shown in Figure 18-75.

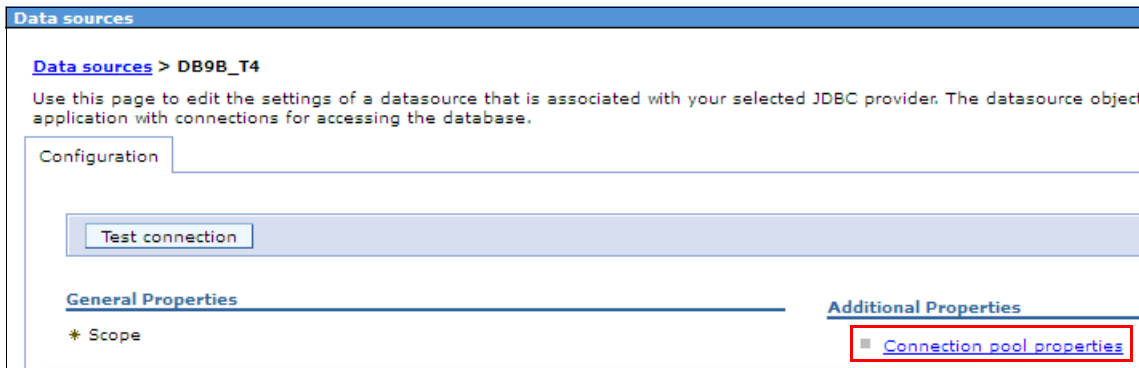


Figure 18-75 Connection pool properties

4. Select the **Connection pool custom properties** link, as shown in Figure 18-76.

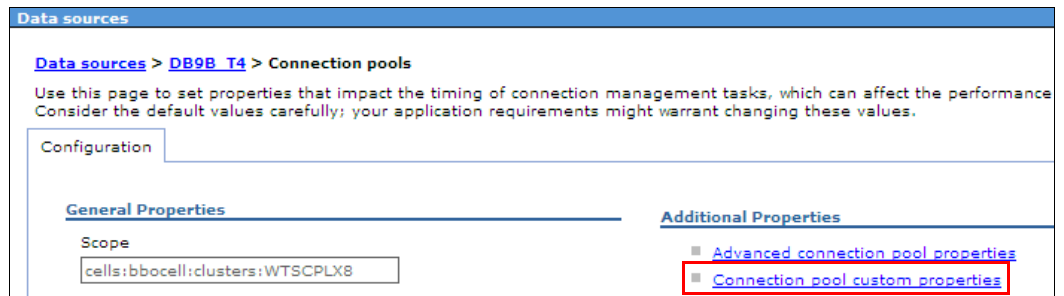


Figure 18-76 Connection pool custom properties

5. Click **New** to add a new custom property, as shown in Figure 18-77.

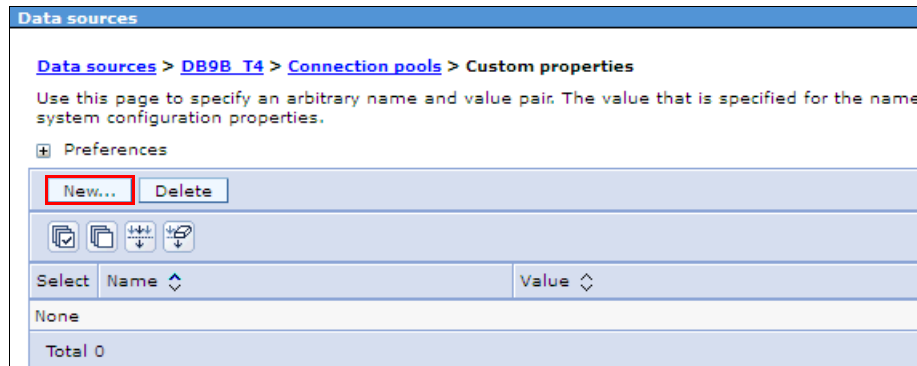


Figure 18-77 Add new custom property

6. Enter a name of alternateResourceJNDIName, and enter a value of your mirror failover data source JNDI name. Click **Apply**, as shown in Figure 18-78.

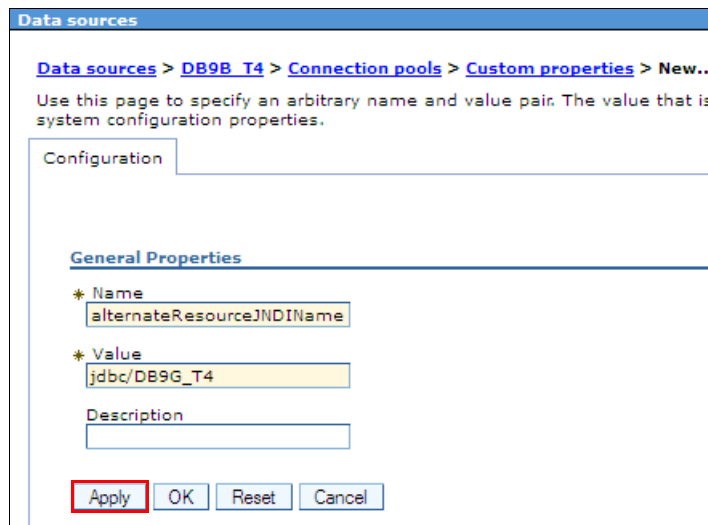


Figure 18-78 New Connection pools custom property configuration

7. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**.

8. Repeat steps 5 to 7 to create additional custom properties to tailor the failover options, as shown in Figure 18-79.

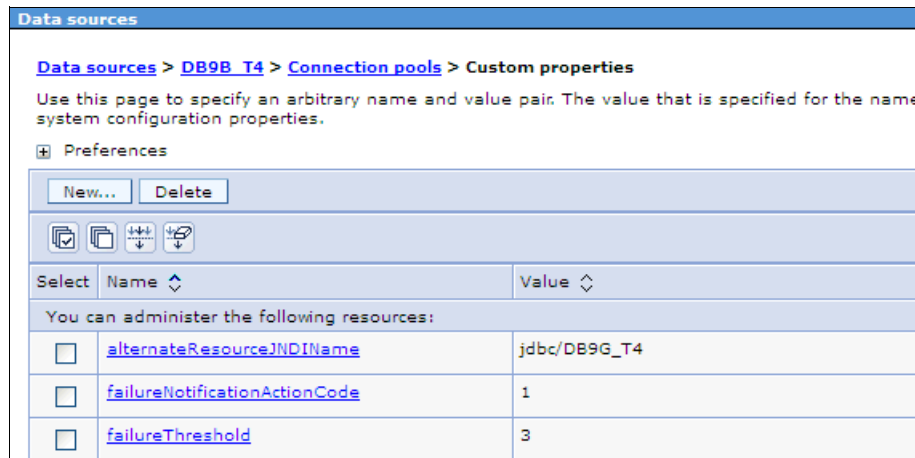


Figure 18-79 Connection pools custom properties window

9. Repeat steps 2 to 8 on your mirrored failover data source to enable failback to the primary data source.
10. Restart the affected application servers by clicking **Servers** → **Server Types** → **WebSphere application servers**.

Test: You are encouraged to test the suitability of this feature with your system environment and resources before enabling failover. Mbean support for resource workload routing is provided. The alternate resource configuration must mirror the primary resource configuration.

You can use the commands in the following examples to manually initiate or to enable or disable failover and failback for the specified JNDI name:

- ▶ **F servername,FAILOVER, 'jdbc/DB9B_T4'**

The output is shown in Example 18-4.

Example 18-4 Output of the F servername,FAILOVER command

```
ExtendedMessage: BB000222I: J2CA0690I: The FailOverToAlternateResource
operation issued for the resource with a JNDI name of jdbc/DB9B_T4 completed
succesfully.
```

- ▶ **F servername,FAILBACK, 'jdbc/DB9B_T4'**

The output is shown in Example 18-5.

Example 18-5 Output of the F servername,FAILBACK command

```
ExtendedMessage: BB000222I: J2CA0690I: The FailBackToPrimaryResource operation
issued for the resource with a JNDI name of jdbc/DB9B_T4 completed succesfully.
```

- ▶ **F servername,DISABLEFAILOVER, 'jdbc/DB9B_T4'**

The output is shown in Example 18-6.

Example 18-6 Output of the F servername,DISABLEFAILOVER command

ExtendedMessage: BB000222I: J2CA0690I: The DisableResourceFailOver operation issued for the resource with a JNDI name of jdbc/DB9B_T4 completed successfully.

- ▶ **F servername,ENABLEFAILOVER, 'jdbc/DB9B_T4'**

The output is shown in Example 18-7.

Example 18-7 Output of the F servername,ENABLEFAILOVER command

ExtendedMessage: BB000222I: J2CA0690I: The EnableResourceFailOver operation issued for the resource with a JNDI name of jdbc/DB9B_T4 completed successfully.

For more information about the resource workload routing feature, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcdat_dsfailover.html

18.3.7 High-availability application update rollout

To allow for automatic pause or resume of server during application update rollout:

1. Click **System administration** → **Node agents**.
2. Select the node agent that is related to the cluster that will be involved in the high-availability rollout, as shown in Figure 18-80.

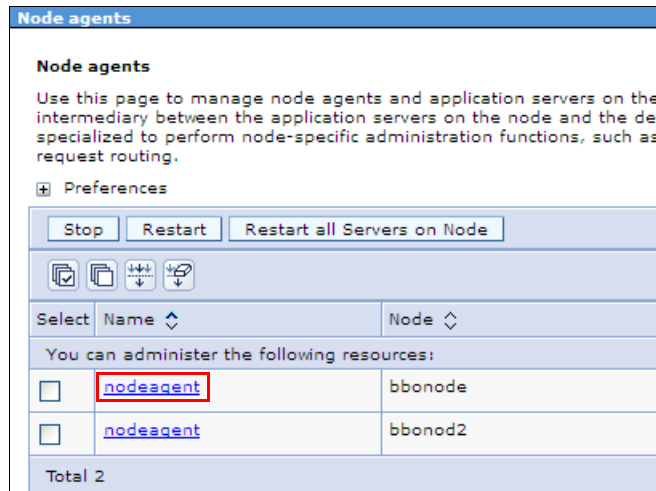


Figure 18-80 Select node agent

3. In the Additional Properties section, select **Administration services**, as shown in Figure 18-81.

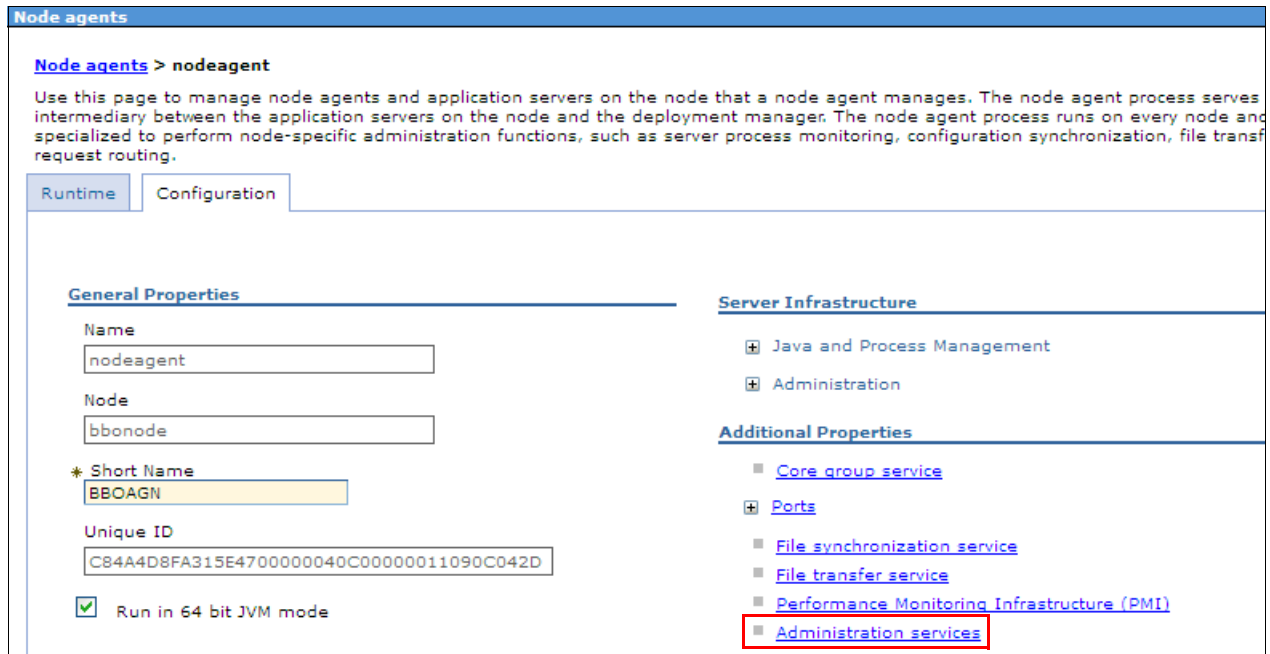


Figure 18-81 Administration services

4. In the Additional Properties section, select **Custom properties**, as shown in Figure 18-82.

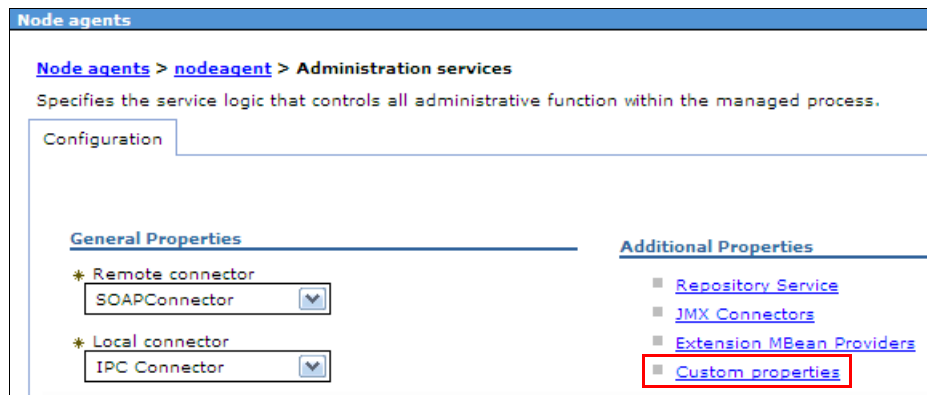


Figure 18-82 Custom properties

5. Click **New** to add a new custom property, as shown in Figure 18-83.



Figure 18-83 New custom property

6. Enter a name of `com.ibm.websphere.zos.rollout.pauseresume` with a value of `true`. Click **Apply**, as shown in Figure 18-84.

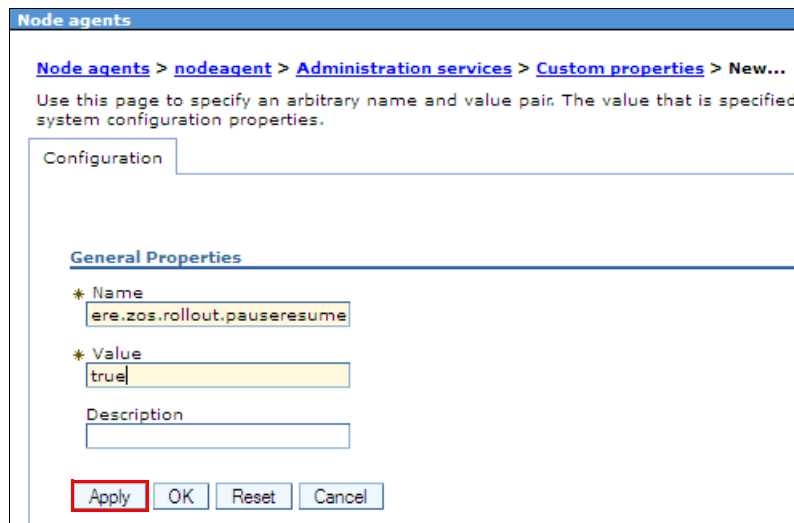


Figure 18-84 New administration service custom property

7. Click **Review**, click **Synchronize changes with nodes**, and click **Save**. Alternatively, click **System administration** → **Nodes**, and click **Synchronize** with the appropriate node or nodes selected, or enable the **Synchronize changes with Nodes** option in **System administration** → **Console Preferences**. Click **Save** and **OK**.

8. Repeat steps 5 to 7, but enter a custom property with a name of `com.ibm.websphere.zos.mvsservices.enable` and a value of `true` so that the Administration services Custom properties view contains the properties shown in Figure 18-85.

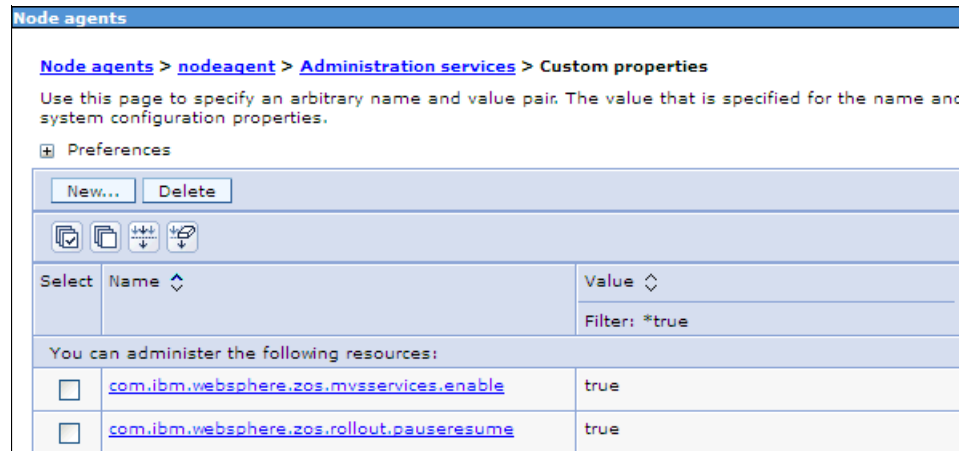


Figure 18-85 Administration services custom properties window

9. Repeat steps 2 to 8 for all the node agents tied to your cluster.
10. Restart the affected node agents.

Manual rollout procedure: For the manual rollout procedure, disable nodeagent Automatic synchronization and Startup synchronization under **System administration** → **Node agents** → **Node agent name** → **File synchronization service**.

For information about a staged application deployment rollout, go to the following website:

<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101641>

For information about updating a high-availability application manually with a server stop, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcontainer_update_ha_apps_zos.html

18.4 Enabling multiple servants

Enabling multiple servants on your systems can improve resource availability on your system. With WLM you can control the number of servants that are running and how they are performing. WLM manages the response time and throughput of transactions according to their assigned service class, the associated performance objectives, and the availability of system resources.

Based on your workload requirements you can control the maximum or minimum number of servants running on a server. The minimum value is used for starting up a basic number of servers and the maximum value is used for capping the number of address spaces that are started by WLM for each server instance.

To use this service, the multiple server instance needs to be enabled from the administrative console:

1. Select **Servers** → **Server Types** → **WebSphere Application Server** → **server_name**.
2. Under **Server Infrastructure** → **Java and Process Management** → **Server Instance**, (Figure 18-86).

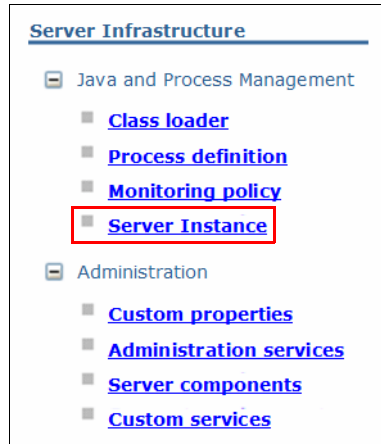


Figure 18-86 Selecting Server Instance

3. Select the **Multiple Instance Enabled** option and then set the values for the maximum and minimum number of instances, (Figure 18-87). If you leave these fields blank, WLM will determine the numbers.

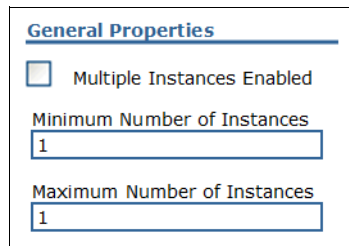


Figure 18-87 Configuration value setting

4. Click **Apply** to apply the changes and then click **Save** to save the changes.

You can use the z/OS **MODIFY** command to dynamically increase the minimum and maximum values. For example, your server is currently configured with minimum set to two and maximum set to four, and currently you have only two servants running, but you want all four to be running. Issue the following **MODIFY** command to change the minimum value and start the additional servants.

```
F Z85SR02B,WLM_MIN_MAX=4,6
```

The above command changed the minimum value to 4 and maximum value to 6. Similarly if we want to modify from min=4, max=6 to min=5 and max=6, we can use the following command:

```
F Z85SR02B,WLM_MIN_MAX=5,6
```

When using the **MODIFY** command, if the minimum value is greater than the configured value, the excess servants are started. If the minimum value is less than the configured value, the excess servants are stopped.

Similarly, if the current number of servants running is four, and there are requirements for more servants, WLM starts the additional servants until the maximum configured value is reached.

For more information about configuring the servant region, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrun_control_servants.html

18.4.1 Balancing workload with WLM

When using multiple servants, there can be situations where one servant is utilized more than the other servants because WLM will not bring another servant region in use until the first one is fully used or a problem is encountered with the servant region.

To help evenly distribute the workload between servant regions, you can use the environment variable for enabling WLM stateful session placement.

To set the variable on: `wlm_stateful_session_placement_on`.

This tells WLM to balance session affinities across different servant regions. Session affinity is when a session is established in a particular JVM, subsequent requests that belong to the same session are assigned to the JVM that has the established session affinity.

By using the `wlm_stateful_session_placement_on` variable, WLM balances the load between servant regions. So if a new user request comes in, WLM directs the request to another servant region.

This setting is mainly used for applications having requirements for session affinity.

18.4.2 Balancing workload without WLM

There can be situations where you do not want to use WLM for load balancing but rather based load balancing on application requirements. In this case, work distribution can be controlled using WebSphere queuing, following an algorithm based on application requirements, such as round robin or session affinity. The application server custom property `server_work_distribution_algorithm` can be used.

To use this variable for load balancing:

- ▶ Set the `server_use_wlm_to_queue_work` property to 0:

```
server_use_wlm_to_queue_work 0
```

- ▶ Set the `server_work_distribution_algorithm` to 0, 1 or 2, based on one of the following settings:

0 - Hot Thread: In this case, each new work request is assigned to the first servant that has a thread available to process the request. If none of the servants have an available thread, the request is queued into the global work queue that is shared by all servants. The request is then selected from the global work queue when the next thread becomes available, regardless of which servant owns that thread.

1- Round Robin: In this case, new work requests are distributed evenly across all servants. If all of the servant threads are already processing other work requests, the new request is added to the request queue for a specific servant. The queued request is then selected when it becomes the top request in the queue and a thread becomes available in that servant.

2 - Hot Robin: In this case, new work requests are distributed evenly across all servants. If the assigned servant does not have a thread available to process the request, the request is reassigned to another servant that has an available thread. If none of the servants have an available thread, the new request is queued into the global work queue that all of the servants share. The request is then selected from the global work queue when the next thread becomes available, regardless of which servant owns that thread.

We set the environment variable, `server_use_wlm_to_queue_work`, to zero, which implies that work load management has a minimal role to play. With this variable setting, WLM does not start additional servants, although WLM will still classify the work.

This arrangement works for both stateful and stateless applications.

18.5 Additional resources

For more information about high availability, consult the following resources:

- ▶ For more information about support for client reroute for applications that use DB2, go to the following Information Center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftdat_clientreroute.html

- ▶ To learn more about achieving node isolation with intermediate symbolic links, go to the following website:

<http://www-03.ibm.com/support/techdocs/atmsastr.nsf/WebIndex/WP100396>



Monitoring z/OS systems

Being able to measure and monitor system interactions helps IT to provide business continuity. Monitoring capabilities play a key role in successfully managing enterprise systems. In WebSphere Application Server, there are a number of tools that can contribute to the monitoring strategy of an organization and to provide insights into the performance of the application server.

In this chapter, we provide an introduction to these tool sets and talk about the additional tools that are available for WebSphere Application Server on the z/OS platform.

We cover the following topics:

- ▶ Overview
- ▶ Monitoring from the administrative console
- ▶ IBM Tivoli Composite Application Manager for WebSphere Application Server
- ▶ Additional resources for monitoring

19.1 Overview

IT environments are complex, involving many different servers working together to deliver the electronic functions of business. In a single-user interaction, it is typical that information can be retrieved from many systems. Consider the simple WebSphere Application Server for z/OS environment in Figure 19-1.

The stars in Figure 19-1 highlight that even a simple web application request can pass through an entire series of dependent servers to successfully complete a request. JEE is a component-based architecture, requiring that a request interact and use n number of these components to complete. Monitoring system components and their performance can become complex, yet it is critical to understanding the overall performance of an application.

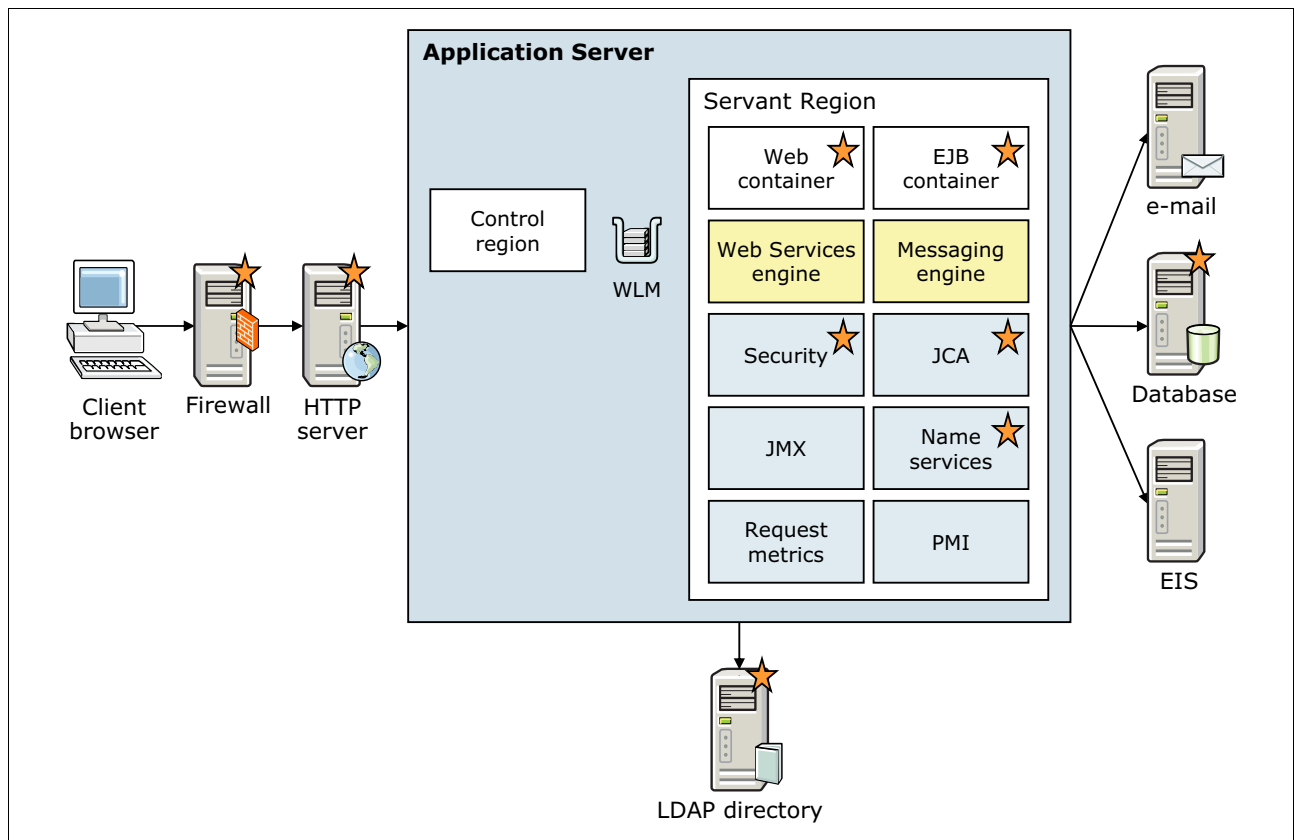


Figure 19-1 Simple web system topology

Monitoring the systems contributes to overall systems management by:

- ▶ Establishing an understanding of the performance baseline and the runtime behaviors that constitute normal operations
- ▶ Measuring performance and identifying poorly performing systems and components
- ▶ Identifying service failures and assisting in root cause identification

The WebSphere Application Server monitoring tools rely primarily on information gathered from two core data infrastructures:

- ▶ Performance Monitoring Infrastructure (PMI), which is a collection of statistical agents scattered throughout the application server that gather statistical data on the performance of the application server components. For additional information, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcpref_pmidata.html

- ▶ Request metrics, which are primarily a set of timing agents that track a request as it navigates the components of the application server. A key differentiation of request metrics is that they are measured at the request level. The focus of a request metric is to record the time spent by individual requests in different components of the application and at the end of the request, provide a record of where the time was spent in the request. For additional information, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftprf_monitoringappflow.html

Other information structures on which WebSphere Application Server V8.5 monitoring tools rely are:

- ▶ WebSphere Application Server for z/OS relies on Work Load Manager (WLM) to collect some of the accounting and performance data.
- ▶ Resource Measurement Facility (RMF) and System Management Facility (SMF) records to provide accounting and performance information to WebSphere.

19.2 Monitoring from the administrative console

The PMI and request metrics tools can be used from the administrative console to monitor WebSphere Application Server for z/OS. For more information, see the following sections in *Chapter 16, "Monitoring distributed systems" on page 553*:

- ▶ 16.2, "Enabling monitoring infrastructures" on page 555
- ▶ 16.3, "Viewing the monitoring data" on page 567
- ▶ 16.4, "Monitoring examples" on page 575

19.2.1 PMI Monitoring

Performance Monitoring Infrastructure (PMI) can monitor each layer in the entire application flow from web server, web container, EJB container, and data source. You can view the process response time that is monitored by request metrics, through the Application Response Measurement (ARM) interface and system log files.

PMI is used for monitoring at run time, and it has no correlation with data across different layers. You need to enable third-party logging and then all logs need to be analyzed together to find the details for particular transactions. This helps to identify the bottleneck layer and response time for each and every layer, and can be analyzed. You get to know which layer is taking maximum time for transaction processing, and based on application behavior further actions can be taken.

For additional information about the request metrics, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fcprf_requestmetrics.html

19.2.2 Monitoring Dynamic Caching

Install the CacheMonitor.ear in the administrative console, which provides the GUI for monitoring cache information at runtime. Manipulation of data is possible through the cache monitor. At times there can be information in the cache, which can cause junk data transfers to users, which is more applicable in heavy content applications. You can use this feature to monitor and clear the data as required.

For additional information about working with the cache monitor, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fwelc6tech_dyn_mon.html

19.2.3 Monitoring web services through PMI

There are specific APIs available for monitoring. To enable PMI for web services:

1. In the administrative console, go to **Monitoring and Tuning** → **Performance Monitoring Infrastructure**, shown in Figure 19-2.

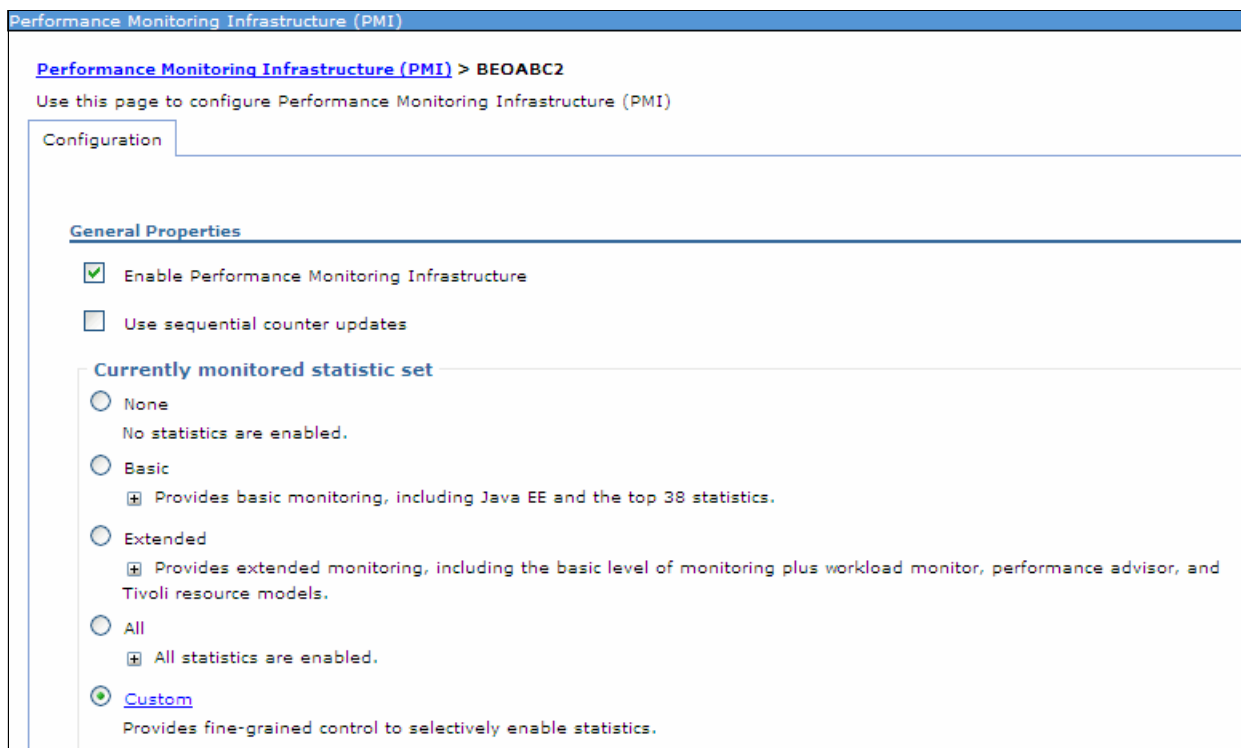


Figure 19-2 Selecting PMI level

2. Select **custom** → **web services**. Select the boxes that are required to cover the business requirement of monitoring according to the application, as shown in Figure 19-3 on page 709.

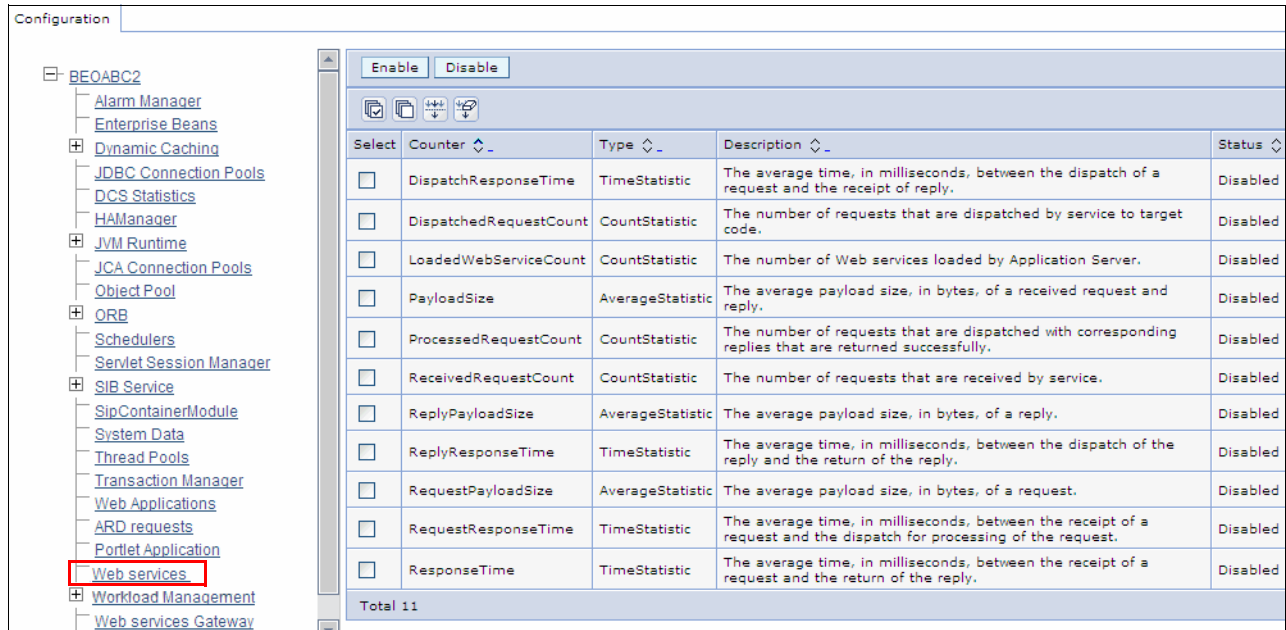


Figure 19-3 Selecting web services and options

Tip: Enable the PMI level and options according to business requirements only because additional monitoring makes it difficult to find relevant information, and also it is performance overhead.

For additional information about monitoring web services applications, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftwbs_performance.html

19.3 IBM Tivoli Composite Application Manager for WebSphere Application Server

IBM Tivoli Composite Application Manager (ITCAM) for WebSphere is shipped with WebSphere Application Server V8.5. After it is installed, it is embedded in the application server and can be configured to monitor application performance, providing real-time status information in the WebSphere administrative console. Further information about ITCAM for WebSphere Application Server is available at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/ecam.html

ITCAM for WebSphere can easily be integrated with ITCAM for Application Diagnostics to provide deep dive diagnostics data, real-time monitoring, and management. ITCAM for Application Diagnostics requires additional licensing. More information is available at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/planning_an_installation/overview_of_itcam_for_application_diagnostics.html

19.3.1 Installing the data collector

The first step in enabling ITCAM is to install and configure a data collector for the monitored servers. ITCAM for WebSphere can be installed from WebSphere media using the IBM Installation Manager. For information about the installation, refer to *ITCAM Program Directory*, GI11-8919-01.

19.3.2 Configuring Tivoli Composite Application Manager for WebSphere metrics

After installing ITCAM for WebSphere, complete the following steps:

1. Navigate to the bin installation root directory, and run the **createcfg.sh** command (or job CYEZISRT), as shown in Example 19-1, using the following parameters:

| | |
|-----------------|--|
| -config | Specifies the itcam configuration directory. |
| -install | Specifies the itcam installation directory. |
| -owner | Specifies the user and group owners. |

Example 19-1 Creating an ITCAM configuration directory

```
/opt/zWebSphere/V8R5/itcamdc/WebSphere/DC/bin # createcfg.sh -config
/wasv85config/wpcell/wpdmnode/ecam -install
/opt/zWebSphere/V8R5/itcamdc/WebSphere/DC -owner WPADMIN:WPCFG
```

```
+-----+
| ITCAM 7.2 for WebSphere Application Server      HCYE720 210878 |
|                                     |
|                               Create runtime configuration directory |
|                                     |
+-----+
```

Create a ITCAM 7.2 for WebSphere Application Server runtime configuration home directory.

You only need to run this script once. All Data Collectors can share the same configuration home.

```
Thu Jul 21 20:24:20 EDT 2011
uid=0(STC) gid=0(TSO)
```

```
-config /wasv85config/wpcell/wpdmnode/ecam
-install /opt/zWebSphere/V8R5/itcamdc/WebSphere/DC
-owner WPADMIN:WPCFG
```

For the ITCAM installation directory path you should enter the logical path (symbolic link) instead of the physical path

(canonical path name). Doing this will provide flexibility in cloning and for service migration.

Enter ITCAM 7.2 for WebSphere Application Server installation directory path
"/opt/zWebSphere/V8R5/itcamdc/WebSphere/DC":

Answer: ""

Enter directory name in which to create the ITCAM 7.2 for WebSphere Application Server configuration root
"/wasv85config/wpcell/wpdmnode/ecam":

Answer: ""

You may set a new owner for the configuration home files and directories.

The owner may be entered as "user" or "user:group".
An owner of "none" will bypass setting the owner.

Enter owner for the configuration home "WPADMIN:WPCFG":

Answer: ""

You may set permissions for the configuration home files and directories.

Permissions are entered in chmod format, which may be either numeric, like "664", or symbolic, like "a+rX,u+w,g+w,o-w".
Permissions specified as "none" will bypass setting permissions.

Enter permissions for the configuration home "none":

Answer: ""

```
+-----+  
| Summary of chosen configuration parameters |  
+-----+
```

createcfg will create an ITCAM 7.2 for WebSphere Application Server configuration home with the following parameters:

- 1) ITCAM installation path : /opt/zWebSphere/V8R5/itcamdc/WebSphere/DC
- 2) Config home path : /wasv85config/wpcell/wpdmnode/ecam

Existing runtime : No
Owner will be : WPADMIN:WPCFG

Do you wish to proceed with configuration home creation? (y|n|q): y

Answer: "y"

For directory /wasv85config/wpcell/wpdmnode/ecam:

```
Successfully created runtime subdirectory
Successfully created runtime/custom subdirectory
Successfully created bin subdirectory
Successfully created itcamdc symbolic link.
Successfully created toolkit symbolic link.
Successfully created plugins symbolic link.
Successfully created symbolic link for bin/amupdate.sh.
Successfully created symbolic link for bin/cye_collector.sh.
Successfully created symbolic link for bin/reconfig_dc_was.sh.
Successfully created symbolic link for bin/setmode.sh.
Successfully created symbolic link for bin/setupdc.sh.
Successfully created symbolic link for bin/unconfig.sh.
Successfully created symbolic link for bin/configDataCollector.jacl.
Successfully created symbolic link for bin/findServers.jacl.
Successfully created symbolic link for bin/reconfig_dc_was.jacl.
Successfully created symbolic link for bin/unconfigDataCollector.jacl.
Successfully edited setmode.cntl
Successfully created itcam.properties file.
Successfully created toolkit_global_custom.properties file.
```

Setting ownership of /wasv85config/wpcell/wpdmnode/ecam to WPADMIN:WPCFG

Configuration home directory created:

```
/wasv85config/wpcell/wpdmnode/ecam
```

*** Successful completion of createcfg script. ***

2. Navigate to the **wsadmin.sh** command directory, and ensure that you can connect to the WebSphere stand-alone or deployment manager at the SOAP or RMI port.

Note: Ensure that the user who runs the **wsadmin.sh** command has enough region size because the command starts a Java Virtual Machine and demands some memory. Verify the SIZE parameter in the RACF TSO segment.

3. Navigate to your ITCAM configuration directory, and run the **setupdc.sh** command in prompt mode to configure the data collector on your WebSphere, as shown in Example 19-2.

Note: To run the **setupdc.sh** command, you need at least 512 MB of virtual storage. Check the parameter ASSIZEMAX on the OMVS segment for the user who will run it.

Example 19-2 Configuring the data collector

```
/wasv85config/wpcell/wpdmnode/ecam/bin # setupdc.sh
```

```
+-----+
| ITCAM 7.2 for WebSphere Application Server      HCYE720 211506 |
|
|                                     Data Collector Configuration
|
+-----+
```


+-----+

This script will configure a Data Collector instance for a
WebSphere Application Server

Thu Jul 21 21:20:37 EDT 2011
uid=0(DFS) gid=0(TSO)

Default local host name detected: WTSC58.ITS0.IBM.COM
Default local IP address detected: 9.12.4.8
Local host name used for local node resolution: WTSC58.ITS0.IBM.COM
Local IP address used for local node resolution: 9.12.4.8

Enter the path of the WAS user install root 'u/WAS85':
/wasv85config/wpcell/wpnodea

Answer: "/wasv85config/wpcell/wpnodea"

Searching for wsadmin.sh. Please wait...
Found wsadmin.sh in the following locations:

- 1) /wasv85config/wpcell/wpnodea/AppServer/bin/wsadmin.sh
wsadmin.sh WAS home is /wasv85config/wpcell/wpnodea/AppServer
Node wpnodea in cell wpcell

- 2) /wasv85config/wpcell/wpnodea/AppServer/profiles/default/bin/wsadmin.sh
wsadmin.sh WAS home is /wasv85config/wpcell/wpnodea/AppServer
Node wpnodea in cell wpcell

Choose a wsadmin.sh under an Application Server node
containing the Application Server to be configured,
preferably one under the profiles/default/bin directory.

Enter the number of the wsadmin.sh to use '2': 2

Answer: "2"

Local WAS node is wpnodea in cell wpcell
Using wsadmin:
/wasv85config/wpcell/wpnodea/AppServer/profiles/default/bin/wsadmin.sh
Searching for local Application Servers
This may take some time. Please wait...

Realm/Cell Name: <default>

Username: wpadmin

Password: passw0rd

WASX7209I: Connected to process "dmgr" on node wpdmnode using SOAP connector;
The type of process is: DeploymentManager

WASX7303I: The following options are passed to the scripting environment and
are

available as arguments that are stored in the argv variable:

"u/tmp/am_servers5
0333959"

```
#####  
#  
# findServers for ITCAM 7.2 HCYE720 210878 Thu Jul 21 21:29:28 EDT 2011 #  
#  
#####
```

Node: wpmodea

1) wpsr01a

Server "wpsr01a" has been selected
Checking WebSphere Application Server version...
Found WAS Version 8.5.0.0 in package ND

The following runtime directory will be created:

/wasv85config/wpcell/wpdmnode/ecam/runtime/was85.wpmodea.wpsr01a

```
+-----+  
|  
| ITCAM for WebSphere DC Configuration  
|  
| The ITCAM for WebSphere data collector can be configured to  
| apply the required PMI settings and enable ITCAM. Otherwise  
| ITCAM will be configured as disabled; it may be enabled at a  
| later time using the WebSphere Integrated Console. Enabling  
| at a later time will require a restart of the server.  
|  
+-----+
```

Do you wish to apply the required PMI settings and enable ITCAM? (y|n) Yn": y

Answer: "y"

Required PMI settings will be applied and ITCAM will be enabled.

```
+-----+  
|  
| ITCAM for WebSphere DC Configuration  
|  
| The ITCAM for WebSphere data collector can be configured to  
| to use ITCAM Managing Server for additional deep-dive analysis.  
| The ITCAM Managing Server is installed separately on a UNIX  
| or Windows host.  
|  
| See the product documentation for more information.  
|  
+-----+
```

Do you wish to configure the Data Collector
to use the ITCAM Managing Server? (y|n) Yn":

Answer: ""

JVM Garbage Collection information will not be gathered.

```
+-----+
| Summary of chosen configuration parameters |
+-----+
```

Setup will create an ITCAM runtime with the following parameters:

1) wsadmin script :

/wasv85config/wpcell/wpnodea/AppServer/profiles/default/bin/wsadmin.sh

2) WAS server name : wpsr01a

Cell Name : wpcell
Node Name : wpnodea
WAS Version : 8.5.0.0 ND
WAS Platform : was85

Deployment : Network Deployment
64-bit mode : 64bit

3) Product : ITCAM 7.2 for WebSphere Application Server

Product Home : /wasv85config/wpcell/wpdmnode/ecam

ApplyPMI/
Enable ITCAM : Yes

Managing Server : No

Enter 'y' to continue, item number to respecify, or 'q' to quit: y

Answer: "y"

ITCAM configuration for wpsr01a created in
/wasv85config/wpcell/wpdmnode/ecam/runtime/was85.wpnodea.wpsr01a
Configuring Application Server. This may take some time. please wait...
Realm/Cell Name: <default>

Username: wpadmin

Password: passwOrd

WASX7209I: Connected to process "dmgr" on node wpdmnode using SOAP connector;
The type of process is: DeploymentManager

WASX7303I: The following options are passed to the scripting environment and
are available as arguments that are stored in the argv variable:

"Ý/tmp/input.properties50333959"

```
#####  
# #  
# configDataCollector Version ITCAM 7.2 211371 Thu Jul 21 21:33:40 EDT 2011 #  
# #  
#####
```

response file /tmp/input.properties50333959

product=eCAM

```

server.platform=z/OS

server.id=wpsr01a(cells/wpcell/nodes/wpnodea/servers/wpsr01a|server.xml#Server_
_1184194176402)
server.version=85
server.runtime=${ITCAMDCHOME}/runtime/was85.wpnodea.wpsr01a
server.variable=ITCAMDCHOME=/wasv85config/wpcell/wpdmnode/ecam
server.variable=AM_HOME=/wasv85config/wpcell/wpdmnode/ecam/itcamdc
server.genericJvmArguments=-Xshareclasses:none -Xverify:none
-agentlib:am=${ITCAMDCHOME}/runtime/was85.wpnodea.wpsr01a/

server.genericJvmArguments=-Xbootclasspath/p:${ITCAMDCHOME}/toolkit/lib/bcm-bo
otstrap.jar:${ITCAMDCHOME}/itcamdc/lib/ppe.probe-bootstrap.jar

server.genericJvmArguments=-Djava.security.policy=${ITCAMDCHOME}/itcamdc/etc/d
atacollector.policy
server.systemproperty=am.home=/wasv85config/wpcell/wpdmnode/ecam/itcamdc

server.environment=LIBPATH=${ITCAMDCHOME}/runtime/was85.wpnodea.wpsr01a/lib:${
ITCAMDCHOME}/toolkit/lib:
server.classpath=
server.environment=NLSPATH=${ITCAMDCHOME}/toolkit/msg/%L/%N.cat
server.genericJvmArguments=
server.systemproperty=TEMAGCCollector.gclog.path=
server.enablePMI=Yes
server.pmiservice.statisticSet=custom
control.systemproperty=ITCAM_DC_ENABLED=true

control.systemproperty=ws.ext.dirs=${ITCAMDCHOME}/itcamdc/lib:${ITCAMDCHOME}/i
tcamdc/lib/ext:${ITCAMDCHOME}/itcamdc/lib/ext/was:${ITCAMDCHOME}/toolkit/lib:${
ITCAMDCHOME}/toolkit/lib/ext

product=eCAM
serverid=wpsr01a(cells/wpcell/nodes/wpnodea/servers/wpsr01a|server.xml#Server_1
1
84194176402)
server=wpsr01a
node=wpnodea
product home=/wasv85config/wpcell/wpdmnode/ecam
product platform=z/OS
Server major version is 85.

Start setting attributes for wpsr01a
Configuring JVM Args
Modify JVM command line arguments
create new variable AM_OLD_JVM_ARGS

Final merged generic JVM arguments:

-Xshareclasses:none
-Xverify:none
-agentlib:am=${ITCAMDCHOME}/runtime/was85.wpnodea.wpsr01a/
-Xbootclasspath/p
:${ITCAMDCHOME}/toolkit/lib/bcm-bootstrap.jar

```

```

    :${ITCAMDCHOME}/itcamdc/lib/ppe.probe-bootstrap.jar
    -Djava.security.policy=${ITCAMDCHOME}/itcamdc/etc/datacollector.policy

create new variable AM_CONFIG_JVM_ARGS
Configuring System Properties
Create/Modify JVM system properties
newProperty={name am.home} {value /wasv85config/wpcell/wpdmnode/ecam/itcamdc}
new am.home = "/wasv85config/wpcell/wpdmnode/ecam/itcamdc"
create new property am.home = "/wasv85config/wpcell/wpdmnode/ecam/itcamdc"
newProperty={name TEMAGCCollector.gclog.path} {value {}}
new TEMAGCCollector.gclog.path = ""
create new property TEMAGCCollector.gclog.path = ""
Configuring Environment Variables
Create/Modify java process with 2 environment variables
Creating new environment
LIBPATH=${ITCAMDCHOME}/runtime/was85.wpnodea.wpsr01a/lib:${ITCAMDCHOME}/toolkit
/lib:
Creating new environment NLSPATH=${ITCAMDCHOME}/toolkit/msg/%L/%N.cat
Configuring server variable ITCAMDCHOME
create new variable ITCAMDCHOME
Configuring server variable AM_HOME
create new variable AM_HOME
Configuring server region classpath
Configuring Control Region system properties
Create/Modify JVM system properties
newProperty={name ITCAM_DC_ENABLED} {value true}
new ITCAM_DC_ENABLED = "true"
create new property ITCAM_DC_ENABLED = "true"
newProperty={name ws.ext.dirs} {value
${ITCAMDCHOME}/itcamdc/lib:${ITCAMDCHOME}
/itcamdc/lib/ext:${ITCAMDCHOME}/itcamdc/lib/ext/was:${ITCAMDCHOME}/toolkit/lib:
${ITCAMDCHOME}/toolkit/lib/ext}}
new ws.ext.dirs =
"${ITCAMDCHOME}/itcamdc/lib:${ITCAMDCHOME}/itcamdc/lib/ext:${I
TCAMDCHOME}/itcamdc/lib/ext/was:${ITCAMDCHOME}/toolkit/lib:${ITCAMDCHOME}/toolk
it/lib/ext"
create new property ws.ext.dirs =
"${ITCAMDCHOME}/itcamdc/lib:${ITCAMDCHOME}/itcamdc/lib/ext:${ITCAMDCHOME}/itcam
dc/lib/ext/was:${ITCAMDCHOME}/toolkit/lib:${ITCAMDCHOME}/toolkit/lib/ext"
PMI service is already enabled
Configuring PMI Service statistics_set
Disabling SMF data collection
Removing WebSphere variable AM_HOME
server_SMF_server_interval_enabled control region environment property retained
server_SMF_container_interval_enabled control region environment property
retained
server_SMF_interval_length control region environment property retained

Validating state of server wpsr01a on node wpnodea
Validation completed successfully
WAS validation log file =
/wasv85config/wpcell/wpnodea/AppServer/profiles/default/logs/wsadmin.valout

Saving changes to server wpsr01a on node wpnodea
Saving complete

```

Synchronizing with node wpsnodea
Synchronization completed successfully on wpsnodea

Successfully configured data collector for server wpsr01a
wsadmin.sh return code is 0
INFO ITCAM plugin gpex.bundle_manager_was.jar copied

Transform by wsc2n.sh in progress
Transform complete, log at
/wasv85config/wpcell/wpdmnode/ecam/runtime/was85.wpsnodea.wpsr01a/logs/transform.log

ITCAM 7.2 for WebSphere Application Server setupdc.sh configuration completed for /wasv85config/wpcell/wpdmnode/ecam/runtime/was85.wpsnodea.wpsr01a

4. After configuring the data collector, navigate to your heap definitions, and increase the maximum heap size field, adding 128 MB to the current value. If no value is set, assume it is 256 MB and set it to 384 MB. In the console, click **Servers** → **Server Types** → **WebSphere application servers** → **<your server>** → **Server Infrastructure** → **Java and process management** → **Process definition** → **Servant** → **Additional Properties** → **Java Virtual Machine**.
5. Restart the WebSphere Application Server.

Note: More details about the ITCAM configuration for WebSphere Application Server data collector is at the following website:

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfad.doc_7101/itcam_ecam_installation_72.pdf

- After the data collector configuration is complete, use the WebSphere administration console to navigate to the PMI window for the configured server (Figure 19-4). A new ITCAM for WebSphere Application Server link is now in the Additional Properties section of the window.

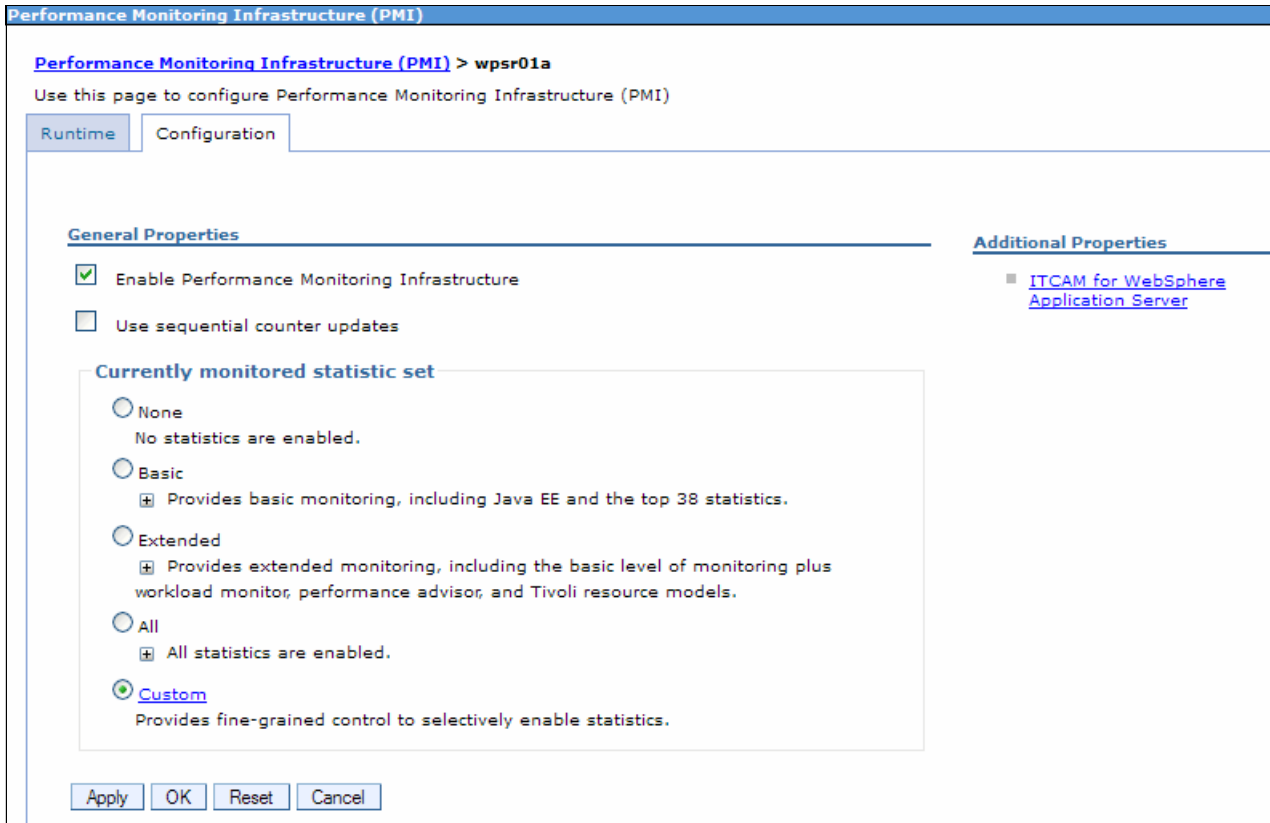


Figure 19-4 PMI configuration with ITCAM for WebSphere Application Server

- Click the **ITCAM for WebSphere Application Server** link to open the window shown in Figure 19-5. This window contains the settings that enable the data collector.

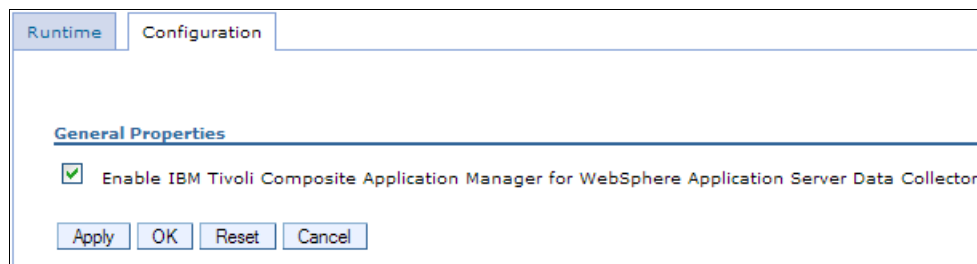


Figure 19-5 ITCAM for WebSphere Application Server window

Note: This setting is only enabled if you choose to apply the required PMI settings during data collector configuration. If not, enable the ITCAM for WebSphere Application Server data collector and restart the server.

The PMI settings must also be configured at the custom level.

19.3.3 Viewing IBM Tivoli Composite Application Manager for WebSphere data

For information about how to view the data monitored by ITCAM for WebSphere Application Server, refer to 16.6.3, “Viewing IBM Tivoli Composite Application Manager for WebSphere data” on page 593. The process is the same for z/OS and distributed platforms.

19.4 Additional resources for monitoring

In this section, we discuss additional resources to be used while monitoring your WebSphere environment.

19.4.1 IBM Support Assistant

IBM Support Assistant (ISA) is a software workbench provided by IBM to help you diagnose and solve questions about IBM software. It offers several tools as add-ons to help you analyze logs, heap dumps, Java dumps, and so on. For details about the tools and features available for z/OS and how to install and configure them, see *Introducing the IBM Support Assistant for WebSphere Application Server on z/OS* at the following website:

<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101575>

19.4.2 Verbose garbage collection

Verbose garbage collection was mentioned in 16.4.1, “JVM memory and CPU usage” on page 576. Again, assuming there is appropriate disk space, a monitoring strategy must include verbose garbage collection. To enable verbose garbage collection:

1. Click **Servers** → **Server Types** → **WebSphere application servers** → **<your server>** → **Server Infrastructure** → **Java and process management** → **Process definition**, as shown in Figure 19-6.

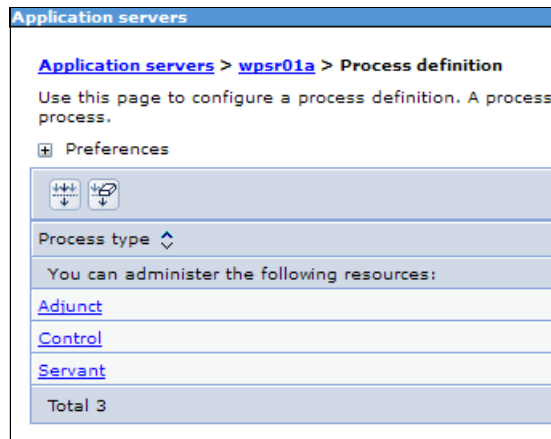


Figure 19-6 Process definition type

2. You can define the garbage collection for any of the above processes. However, it is more efficient to set it to run in a servant process, where the application runs and most of the tuning occurs automatically. Click **Servant** → **Additional Properties** → **Java Virtual Machine**, and select the **Verbose garbage collection** box, as shown on Figure 19-7 on page 721.

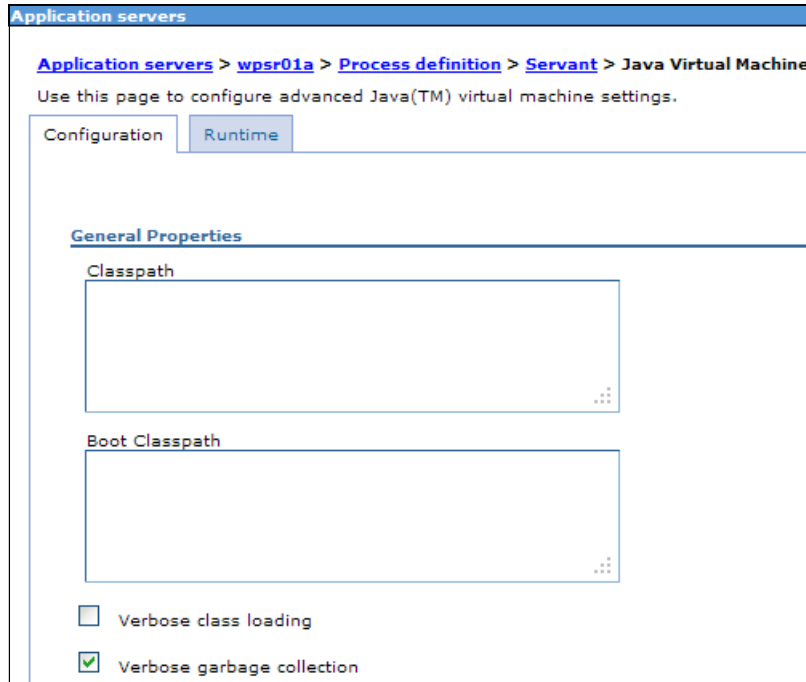


Figure 19-7 Enable verbose garbage collection

Verbose garbage collection can also be enabled using the Runtime tab, as shown in Figure 19-8.

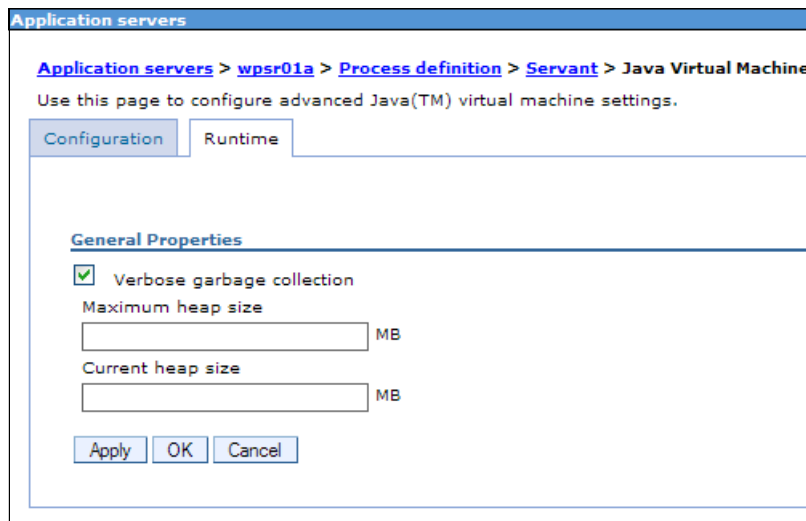


Figure 19-8 Enabling verbose garbage collection at run time

Note: After enabling verbose garbage collection, restart the server to validate the change.

When verbose garbage collector is enabled, a report is written to the output stream each time the garbage collector runs. The report is typically located in SYSOUT DD from servant output. In Version 7, WebSphere used an optimal throughput (**optthroughput**) algorithm for garbage collection. Currently, generational concurrent garbage (**gencon**) collection is used, allowing for performance improvements. Because **optthroughput** uses a large contiguous heap shared by all threads, when garbage collection is invoked, all of this area is scanned. This policy is

beneficial for applications that demand optimal throughput, but it has long pause times. Generational concurrent garbage collection divides the heap memory in two pieces:

- ▶ Nursery, for new objects
- ▶ Tenured, for aged objects

So, in this policy, the time spent to scan one of the areas is shorter.

Example 19-3 shows a sample garbage collection entry in SYSOUT.

Example 19-3 Garbage collection entry

```
<exclusive-start id="274" timestamp="2011-07-20T22:44:56.947"
intervalms="34510.096">
  <response-info timems="0.046" idlms="0.046" threads="0"
lastid="0000004808DCB900" lastname="WebSphere non-WLM Dispatch Thread t=007c7c68"
/>
</exclusive-start>
<af-start id="275" totalBytesRequested="40" timestamp="2011-07-20T22:44:56.948"
intervalms="34510.089" />
<cycle-start id="276" type="scavenge" contextid="0"
timestamp="2011-07-20T22:44:56.948" intervalms="34510.087" />
<gc-start id="277" type="scavenge" contextid="276"
timestamp="2011-07-20T22:44:56.948">
  <mem-info id="278" free="157892256" total="261947392" percent="60">
    <mem type="nursery" free="0" total="60620800" percent="0" />
    <mem type="tenure" free="157892256" total="201326592" percent="78">
      <mem type="soa" free="147826336" total="191260672" percent="77" />
      <mem type="loa" free="10065920" total="10065920" percent="100" />
    </mem>
    <remembered-set count="17359" />
  </mem-info>
</gc-start>
<allocation-stats totalBytes="42822544" >
  <allocated-bytes non-tlh="94144" tlh="42728400" />
  <largest-consumer threadName="WebSphere non-WLM Dispatch Thread t=007c7c68"
threadId="0000004808DCB900" bytes="40718768" />
</allocation-stats>
<gc-op id="279" type="scavenge" timems="105.492" contextid="276"
timestamp="2011-07-20T22:44:57.053">
  <scavenger-info tenureage="6" tilratio="70" />
  <memory-copied type="nursery" objects="226786" bytes="14839960"
bytesdiscarded="4904" />
  <memory-copied type="tenure" objects="65747" bytes="4425048"
bytesdiscarded="1512" />
  <finalization candidates="976" enqueued="572" />
  <references type="soft" candidates="4850" cleared="0" enqueued="0"
dynamicThreshold="29" maxThreshold="32" />
  <references type="weak" candidates="567" cleared="45" enqueued="16" />
  <references type="phantom" candidates="2" cleared="0" enqueued="0" />
</gc-op>
<gc-end id="280" type="scavenge" contextid="276" durationms="105.833"
timestamp="2011-07-20T22:44:57.053">
  <mem-info id="281" free="200176936" total="263258112" percent="76">
    <mem type="nursery" free="46784264" total="61931520" percent="75" />
    <mem type="tenure" free="153392672" total="201326592" percent="76">
      <mem type="soa" free="143326752" total="191260672" percent="74" />
```

```
    <mem type="10a" free="10065920" total="10065920" percent="100" />
  </mem>
  <pending-finalizers system="506" default="66" reference="16" classloader="0"
/>
  <remembered-set count="13724" />
  </mem-info>
</gc-end>
<cycle-end id="282" type="scavenge" contextid="276"
timestamp="2011-07-20T22:44:57.054" />
<allocation-satisfied id="283" threadId="0000004808DCB900" bytesRequested="40" />
<af-end id="284" timestamp="2011-07-20T22:44:57.054" />
<exclusive-end id="285" timestamp="2011-07-20T22:44:57.054" durationms="106.246"
/>
```

These and other tools that make garbage collection analysis easier are located on the IBM Support Assistant website at:

<http://www-01.ibm.com/software/support/isa/download.html>

For more information about Java memory management, refer to the Java information center at the following website:

http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Funderstanding%2Fmemory_management.html

19.4.3 Java dump and core files

Java core files present, in essence, a picture of what is occurring inside the Java Virtual Machine. These files are helpful for analyzing situations, such as when CPU utilization is nearing 100%, when threads are hanging, or when performance is slow.

Java dump and system dump files are a picture of the objects that were in Java Virtual Machine memory. These files are helpful in diagnosing memory-related problems, such as memory leaks.

Both kinds of files can be generated at the WebSphere console, as follows:

1. Click **Troubleshooting** → **Java dumps and cores**
2. Select the desired server.

3. Click one of the available buttons, as shown in Figure 19-9:

- **Heap dump**
- **Java core**
- **System dump**

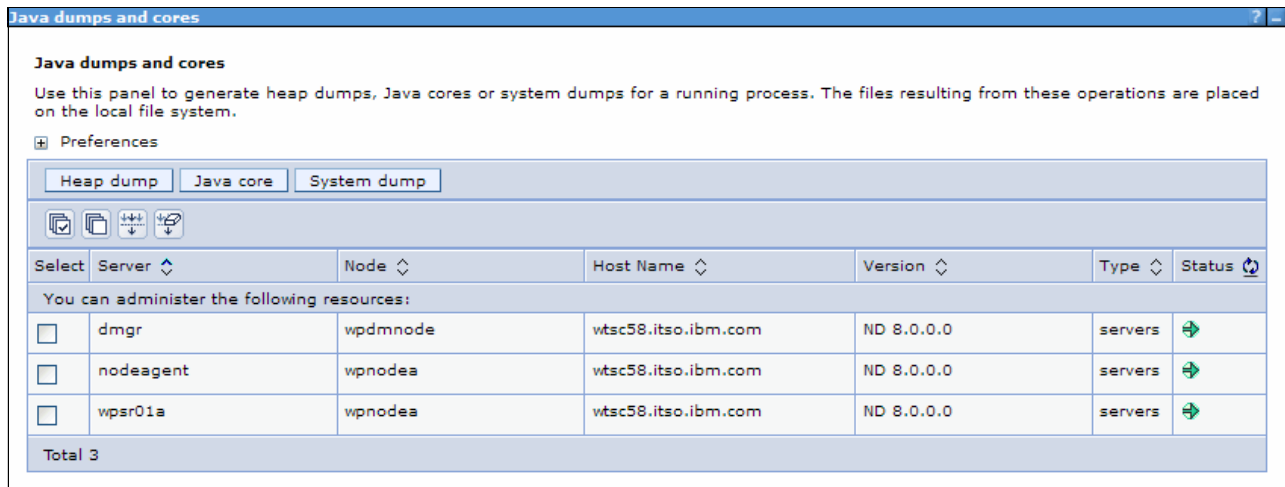


Figure 19-9 Core and dump generation

4. Check the job log for information about the files that were generated.

For more information about javadumps, refer to the Java information center at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fjavadump.html>

For more information about heapdumps, refer to the Java information center at the following website:

<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fheapdump.html>

For more information about system dumps, refer to the Java information center at the following website:

http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fdump_viewer.html

19.4.4 Basic logging

WebSphere Application Server V8.5 provides many logging options. The most significant environment incidents are logged automatically. Basic logging provides information in the Job Entry SubSystem (JES) spool, which can be seen in SYSOUT and SYSPRINT cards from process `sysout`. The IBM Tivoli Log Analyzer cannot be used to analyze basic logs in distributed platforms. Different from basic logs in distributed platforms, the IBM Tivoli Log Analyzer cannot be used to analyze them.

19.4.5 Advanced logging

In WebSphere Application Server V8.5, an alternative to the basic log and trace facility is offered, called High Performance Extensible Logging (HPEL). It provides three repositories:

- ▶ Log data repository: A storage facility for log records, typically information stored in `SystemOut.log`, `SystemErr.log`, or `java.util.logging` at level detail or higher.
- ▶ Trace data repository: A storage facility for trace records, typically information written to `java.util.logging` below level detail.
- ▶ Text log: Plain text file for log and trace records. Provided for convenience.

Note: All data that is written to the log and trace repositories is parsed and formatted to be stored in the text log file. For this reason, consider disabling the log file as soon as possible to enhance server performance.

Figure 19-10 depicts the three HPEL repositories.

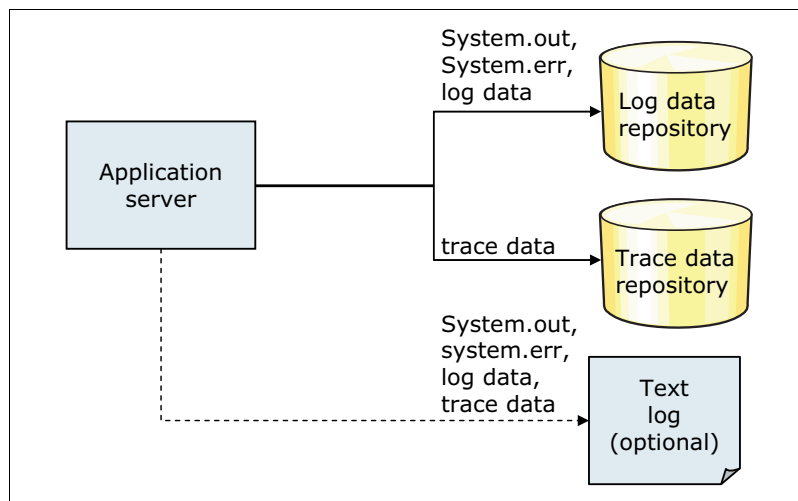


Figure 19-10 HPEL repositories

Now the data is stored in proprietary binary format, rather than text format as it is stored in basic logging. (The only exception is for the text log repository.) In this way, the following benefits are achieved:

- ▶ There is no more text parsing.
- ▶ More data is available due to the fact that truncation does not need to be done.
- ▶ Data is not formatted unless necessary.
- ▶ There is no need to clear log files before server start, for example, to diagnosis a problem.
- ▶ Trace speed is improved and more data can be available (it has half of the impact of basic tracing).
- ▶ It is a common solution between z/OS and distributed platform.
- ▶ Applications running with HPEL run faster than with basic logging.

To read the log and trace records in this new format, a new command called `logViewer.sh` was introduced. It reads the data from repositories, formats it, and displays it to the administrator, as shown in Figure 19-11.

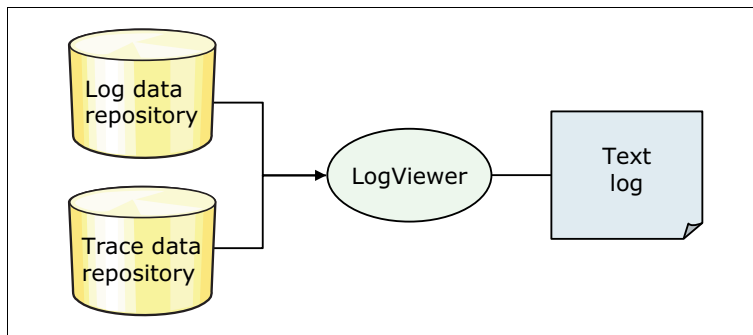


Figure 19-11 `logViewer.sh` command

To activate HPEL logging and tracing, in the administrative console, click **Troubleshooting** → **Logs and trace** → **<your_server>** → **Change log and trace mode** → **Switch to HPEL Mode** (button), as shown in Figure 19-12.

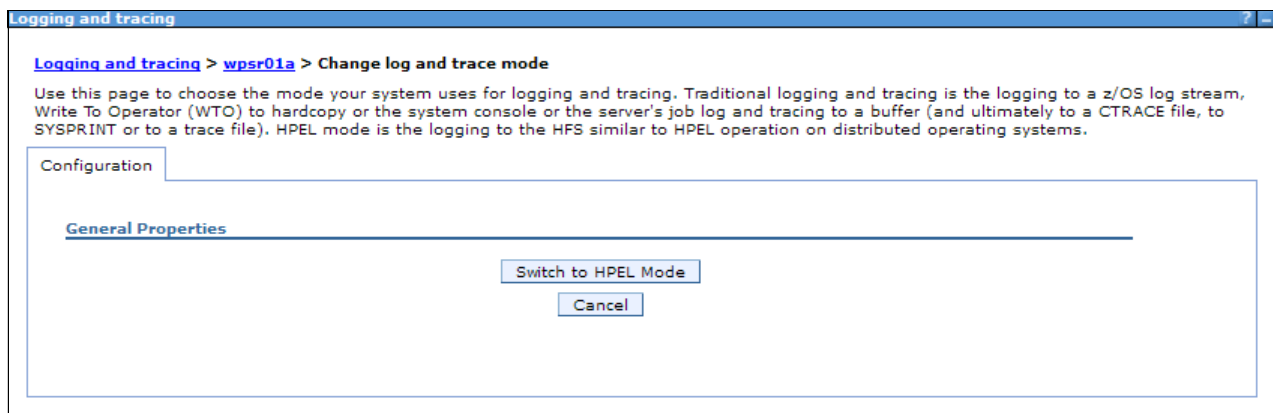


Figure 19-12 Enabling HPEL

Note: A server restart is needed after enabling HPEL logging and tracing.

Viewing data from HPEL repositories

After configuring HPEL, the data can be viewed using the `logViewer.sh` command from the `bin` subdirectory of the profile, as shown in Example 19-4.

Example 19-4 `logViewer.sh` output

Using `/wasv85config/wpcell/wpnodea/AppServer/profiles/default/logs/wpsr01a` as repository directory.

| Instance ID | Start Date |
|--|------------|
| 1311360474568 | 07/22/11 |
| 14:47:54.568 EDT | |
| 1311360474568/000001B400000003-WPSR01AS_STC05860 | 07/22/11 |
| 14:48:52.526 EDT | |

Because a single z/OS server is composed of several Java Virtual Machines (controller, servant, and adjunct), if the instance is not specified, use `logViewer.sh` to list the available instances instead of listing the logs and traces, as in distributed environments. This output is the same as that obtained from the `logViewer.sh -listInstances` command. In Example 19-4 on page 726, the first instance listed is the controller and the other is the servant.

After enabling HPEL, the log format for WebSphere z/OS is similar to that used for a distributed platform, making it easier for administrators from distributed platforms to analyze issues on the z/OS. In Example 19-5, we see the result of the `logViewer.sh -instance 1311360474568` command (control process).

Example 19-5 WebSphere control log

```
Using /wasv85config/wpccell/wpnodea/AppServer/profiles/default/logs/wpsr01a as repository
directory.
***** Start Display Current Environment *****
WebSphere Platform 8.5.0.0 [ND 8.5.0.0 n1118.03] running with process name WPSR01A and process
id 0000019800000005
Host Operating System is z/OS, version 01.12.00
Java version = 1.6.0, Java Compiler = j9jit26, Java VM name = IBM J9 VM
was.install.root = /wasv85config/wpccell/wpnodea/AppServer
user.install.root = /wasv85config/wpccell/wpnodea/AppServer/profiles/default
Java Home = /wasv85config/wpccell/wpnodea/AppServer/java64
ws.ext.dirs =
/wasv85config/wpccell/wpnodea/AppServer/java64/lib:/wasv85config/wpccell/wpnodea/AppServer/classes
:/wasv85config/wpccell/wpnodea/AppServer/lib:/wasv85config/wasv85config/wpccell/wpnodea/AppServer/i
nstalledChannels:/wasv85config/wpccell/wpnodea/AppServer/lib/ext:/wasv85config/wpccell/wpnodea/App
Server/web/help:/wasv85config/wpccell/wpnodea/AppServer/deploytool/itp/plugins/com.ibm.etools.ejb
deploy/runtime:/wasv85config/wpccell/wpnodea/AppServer/java64/jre/lib:/wasv85config/wpccell/wpdmno
de/ecam/itcamdc/lib:/wasv85config/wpccell/wpdmnode/ecam/itcamdc/lib/ext:/wasv85config/wpccell/wpdm
node/ecam/itcamdc/lib/ext/was:/wasv85config/wpccell/wpdmnode/ecam/toolkit/lib:/wasv85config/wpcel
l/wpdmnode/ecam/toolkit/lib/ext
Classpath =
/wasv85config/wpccell/wpnodea/AppServer/profiles/default/properties:/wasv85config/wpccell/wpnodea/
AppServer/properties:/wasv85config/wpccell/wpnodea/AppServer/lib/bootstrap.jar:/wasv85config/wpcel
l/wpnodea/AppServer/lib/bootstrapws390.jar:/wasv85config/wpccell/wpnodea/AppServer/lib/lmproxy.j
ar:/wasv85config/wpccell/wpnodea/AppServer/lib/startup.jar:/wasv85config/wpccell/wpnodea/AppServer
/java64/lib/tools.jar
Java Library path =
/wasv85config/wpccell/wpnodea/AppServer/patches:/wasv85config/wpccell/wpnodea/AppServer/lib/s390-c
ommon:/wasv85config/wpccell/wpnodea/AppServer/lib/s390x-64:/wasv85config/wpccell/wpnodea/AppServer
/java64/bin/classic:/wasv85config/wpccell/wpnodea/AppServer/lib/s390-common:/wasv85config/wpcell
/wpnodea/AppServer/java64/bin:/wasv85config/wpccell/wpnodea/AppServer/java64/bin/j9vm:/wasv85conf
ig/wpccell/wpnodea/AppServer/java64/lib/s390/j9vm:/wasv85config/wpccell/wpnodea/AppServer/java64/l
ib/s390:/wasv85config/wpccell/wpnodea/AppServer/java64/lib/s390x/j9vm:/wasv85config/wpccell/wpnode
a/AppServer/java64/lib/s390x:/wasv85config/wpccell/wpnodea/AppServer/lib:/wasv85config/wpccell/wpn
odea/AppServer/java64/jre/bin:/wasv85config/wpccell/wpnodea/AppServer/java64/jre/lib/s390/j9vm:/w
asv85config/wpccell/wpnodea/AppServer/java64/jre/lib/s390x/j9vm:/wasv85config/wpccell/wpnodea/AppS
erver/java64/jre/lib/s390x
Orb Version = IBM Java ORB build orb626fp1-20110419.00
***** End Display Current Environment *****
[7/22/11 18:47:54:568 GMT] 00000000 ManagerAdmin I   TRAS0017I: The startup trace state is
*=info.
[7/22/11 18:47:54:583 GMT] 00000000 ManagerAdmin I   TRAS0111I: The message IDs that are in use
are deprecated
```

```

[7/22/11 18:47:54:713 GMT] 00000000 ModelMgr      I   WSVR0800I: Initializing core configuration
models
[7/22/11 18:47:55:600 GMT] 00000000 ComponentMeta I   WSVR0179I: The runtime provisioning
feature is disabled. All components will be started.
[7/22/11 18:47:55:694 GMT] 00000000 CommonBridge A   BBOJ0011I: JVM Build is JRE 1.6.0 IBM J9
2.6 z/OS s390x-64 20110418_80450 (JIT enabled, AOT enabled)
J9VM - R26_Java626_GA_FP1_20110418_1915_B80450
JIT  - r11_20110215_18645ifx8
GC   - R26_Java626_GA_FP1_20110418_1915_B80450
J9CL - 20110418_80450.
[7/22/11 18:47:55:697 GMT] 00000000 CommonBridge A   BBOJ0051I: PROCESS INFORMATION:
STC05855/WPSR01A , ASID=102(0x66), PID=2497(0x9c1)
...
Lines removed
...
[7/22/11 18:50:03:770 GMT] 00000020 authz        I   CWWIM2000I Initialization of the
authorization component completed successfully.
[7/22/11 18:50:03:787 GMT] 00000020 UserManagemen I   CWWIM6003I Initialization of the dynamic
reload manager completed successfully.
[7/22/11 18:50:03:789 GMT] 00000019 WsServerImpl A   WSVR0001I: Server CONTROL PROCESS wpsr01a
open for e-business
Operation Complete
Processed 412 records in 0.382 seconds (1,078.534 records per second).

```

Example 19-5 on page 727 shows all of the log details available in the repository because **logViewer.sh** was invoked without any parameters. To see the messages from the most recent server, run **logViewer.sh -instance <your_instance> -latestInstance**.

There are several options to be used with the **logViewer.sh** command, making it a powerful tool to filter events from log and trace repositories. Some possibilities that can be explored are:

- ▶ To show messages starting at a specific level or higher, run:

```
logViewer.sh -instance <your_instance> -minlevel <message_level>
```
- ▶ To show messages from log and trace for a specific thread, run:

```
logViewer.sh -instance <your_instance> -Thread <thread_id>
```
- ▶ To show messages from log and trace in advanced format, run:

```
logViewer.sh -instance <your_instance> -format advanced
```
- ▶ To showing messages from log and trace from a specific time range, run:

```
logViewer.sh -instance <your_instance> -startDate <date/time/timezone>
-stopDate <date/time/timezone>
```

An example of the advanced format view is shown in Example 19-6.

Example 19-6 Advanced format view

```

...
[7/22/11 18:47:56:351 GMT] 00000000 A UOW= source=com.ibm.ws.management.AdminInitializer class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
ADMN0015I: The administration service is initialized.
[7/22/11 18:47:57:127 GMT] 00000000 I UOW=
source=com.ibm.ws.management.component.PluginConfigServiceImpl class= method= org=IBM
prod=WebSphere component=Application Server thread=[main]
PLGC0057I: The plug-in configuration service started successfully.

```



```
[7/22/11 18:47:57:177 GMT] 00000000 I UOW= source=com.ibm.ws.ssl.core.SSLComponentImpl class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0001I: SSL service is initializing the configuration
[7/22/11 18:47:57:202 GMT] 00000000 W UOW= source=com.ibm.ws.ssl.config.WSKeyStore class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0041W: One or more key stores are using the default password.
[7/22/11 18:47:57:220 GMT] 00000000 I UOW= source=com.ibm.ws.ssl.config.SSLConfigManager class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0027I: Disabling default hostname verification for HTTPS URL connections.
[7/22/11 18:47:57:229 GMT] 00000000 I UOW= source=com.ibm.ws.ssl.core.SSLDiagnosticModule
class= method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0014I: The SSL component's FFDC Diagnostic Module
com.ibm.ws.ssl.core.SSLDiagnosticModule registered successfully: true.
[7/22/11 18:47:57:230 GMT] 00000000 I UOW= source=com.ibm.ws.ssl.core.SSLComponentImpl class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPKI0002I: SSL service initialization completed successfully
[7/22/11 18:47:57:246 GMT] 00000000 I UOW= source=com.ibm.wsspi.rasdiag.DiagnosticConfigHome
class=com.ibm.wsspi.rasdiag.DiagnosticConfigHome method=setStateCollectionSpec org= prod=
component= thread=[main]
    RASD0012I: Updating State Collection Spec from Uninitialized Value to .*:.*=0
[7/22/11 18:47:57:257 GMT] 00000000 A UOW= source=com.ibm.ws.pmi.component.PMIImpl class=
method= org=IBM prod=WebSphere component=Application Server thread=[main]
    CWPMI1001I: PMI is enabled
```

...

More information about the **logviewer.sh** command is available at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Frtrb_logviewer.html

Another way to monitor logs and traces is using the WebSphere console. Click **Troubleshooting** → **Logs and trace** → **<your_server>** → **View HPEL logs and trace**, as shown in Figure 19-13.

| TimeStamp | Thread ID | Logger | Level | Message |
|----------------------|-----------|------------|-------|--|
| 7/22/11 18:47:54.568 | 00000000 | ragerAdmin | INFO | TRAS0017I : The startup trace state is *=info. |
| 7/22/11 18:47:54.583 | 00000000 | ragerAdmin | INFO | TRAS0111I : The message IDs that are in use are deprecated |
| 7/22/11 18:47:54.713 | 00000000 | g.ModelMgr | INFO | WSVR0800I : Initializing core configuration models |
| 7/22/11 18:47:55.600 | 00000000 | etaDataMgr | INFO | WSVR0179I : The runtime provisioning feature is disabled. All components will be started. |
| 7/22/11 18:47:55.694 | 00000000 | nmonBridge | AUDI | BBOJ0011I : JVM Build is JRE 1.6.0 IBM J9 2.6 z/OS s390x-64 20110418_80450 (JIT enabled, AOT enabled) J9VM - R26_Java626_GA_FP1_20110418_1915_B80450 JIT - r11_20110215_18645ifx8 GC - R26_Java626_GA_FP1_20110418_1915_B80450 J9CL - 20110418_80450. |
| 7/22/11 18:47:55.697 | 00000000 | nmonBridge | AUDI | BBOJ0051I : PROCESS INFORMATION: STC05855/WPSR01A , ASID=102(0x66), PID=2497(0x9c1) |
| 7/22/11 18:47:55.699 | 00000000 | nmonBridge | AUDI | com.ibm.ws390.orb.CommonBridge printProperties BBOJ0077I : org.osgi.framework.executionenvironment = OSGi/Minimum-1.1,OSGi/Minimum-1.2,JRE-1.1,J2SE-1.2,J2SE-1.3,J2SE-1.4,J2SE-1.5,JavaSE-1.6 |
| 7/22/11 18:47:55.700 | 00000000 | nmonBridge | AUDI | com.ibm.ws390.orb.CommonBridge printProperties BBOJ0077I : osgi.framework = file:/wasv8config/wpcell/wpnodea/AppSe/org.eclipse.osgi.jar |
| 7/22/11 18:47:55.701 | 00000000 | nmonBridge | AUDI | com.ibm.ws390.orb.CommonBridge printProperties BBOJ0077I : java.home = /wasv8config/wpcell/wpnodea/AppServer/jav |
| 7/22/11 18:47:55.702 | 00000000 | nmonBridge | AUDI | com.ibm.ws390.orb.CommonBridge printProperties BBOJ0077I : traceSettingsFile = /wasv8config/wpcell/wpnodea/AppSer/config/cells/wpcell/nodes/wpnodea/servers/wpsr01a/trace.dat |
| 7/22/11 18:47:55.703 | 00000000 | nmonBridge | AUDI | com.ibm.ws390.orb.CommonBridge printProperties BBOJ0077I : eclipse.application = com.ibm.ws.bootstrap.WSLauncher |
| | | | | com.ibm.ws390.orb.CommonBridge printProperties BBOJ0077I : org.osgi.framework.bootdelegation = com.ibm.jvm,com.ibm.lang.management,com.ibm.oti.reflect,com.ibm.oti.shared,com.ibm.oti.util,com.ibm.oti.vm,com.ibm.paas,com.ibm.paas.internal,com.ibm.tools.attach,com.ibm.tools.attach.javaSE,com.ibm.tools.attach.spi,org.apache.harmony.kernel.vm,org.apache.harmony.annotation.internal.nls,org.apache.harmony.beans,org.apache.harmony.beans.editors,org.apache.harmony.objectweb.asm,org.objectweb.asm.signature,com.ibm.j9ddrc,com.ibm.j9ddrc.corereaders,com.ibm.j9ddrc.corereaders.aix,com.ibm.j9ddrc.corereaders.debugger,com.ibm.j9ddrc.corereaders.elf,com.ibm.j9ddrc.corereaders.minidump,com.ibm.j9ddrc.corereaders.osthread,com.ibm.j9ddrc.corereaders.tdump,com.ibm.j9ddrc.corereaders.tdump.zebedee.le,com.ibm.j9ddrc.corereaders.tdump.zebedee.mvs,com.ibm.j9ddrc.corereaders.tdump,com.ibm.j9ddrlibraries,com.ibm.j9ddrutil,com.ibm.j9ddrlogging,com.ibm.j9ddrvm23.events,com.ibm.j9ddrvm23.j9,com.ibm.j9ddrvm23.j9.gc,com.ibm.j9ddrvm23.j9.stackmap,com.ibm.j9ddrvm23.j9.stackwalker,com.ibm.j9ddrvm23.j9.walkers,com.ibm.j9ddrvm23.pointer,com.ibm.j9ddrvm23.pointer.generated,com.ibm.j9ddrvm23.type,com.ibm.j9ddrvm24.events,com.ibm.j9ddrvm24.j9,com.ibm.j9ddrvm24.pointer,com.ibm.j9ddrvm24.pointer.generated,com.ibm.j9ddrvm24.types,com.ibm.j9ddrvm26.events,com.ibm.j9ddrvm26.j9,com.ibm.j9ddrvm26.j9.gc, |

Figure 19-13 HPEL log and trace viewed in the WebSphere console

Expanding the Content and Filtering Details link, you can see all of the possible filter options for viewing logs and traces, as shown in Figure 19-14.

The screenshot shows a web-based interface for HPEL console filtering. At the top, there is a section titled "Content and Filtering Details" with a minus sign icon. Below this is a "Server Instance" section with the text "Server instances grouped by server start date and time:". It contains a tree view with a folder for "July 22, 2011", which is expanded to show a folder for "18:48:52". Underneath, there are two server instance identifiers: "1311360532526/000001B400000003-WPSR01AS_STC05860" and "1311360532526", with the second one highlighted in green. Below the tree view are three sections: "View Contents" with checkboxes for "System out", "System err", and "Logs and trace", and dropdown menus for "Minimum level" and "Maximum level"; "Filtering" with text "Wild cards: *,?,% are allowed" and "Separate multiple entries by a ':'", and input fields for "Include loggers:", "Exclude loggers:", and "Message contents:"; and "Event Timing" with input fields for "From:", "On:", "Until:", and "On:". At the bottom left, there are "Apply" and "Reset" buttons.

Figure 19-14 HPEL console filtering options

More filtering options are available using the buttons at the top of the console messages, as shown in Figure 19-13 on page 730.

More information about HPEL is at the information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Fctrb_HPELOverview.html

19.4.6 z/OS monitoring

To monitor information about z/OS, several operating systems tools and facilities can be configured and used. We show how to monitor WebSphere from the operating system perspective.

SMF

Information about WebSphere Application Server can be obtained from SMF records. It uses record type 120 to record information about its behavior, so when you start to configure the WebSphere environment, it is important to configure your SMFPRMxx member from PARMLIB to record SMF record type 120. You can change the SYS or SUBSYS statement, as shown in Example 19-7.

Example 19-7 SMFPRMxx configuration for record type 120

```
SUBSYS(STC,EXITS(IEFU29,IEFACTRT),INTERVAL(SMF,SYNC),  
TYPE(0,30,70:79,88,89,120,245))
```

Note: After configuring SMFPRMxx, use the SET SMF=xx command to refresh the configuration.

More WebSphere configuration is needed. In the WebSphere console, click **Servers** → **Server Types** → **WebSphere application servers** → **<your server name>** → **Server Infrastructure** → **Java and Process Management** → **Process definition** → **Control** → **Additional Properties** → **Environment Entries**, and define one or more entries shown in Figure 19-15.

The screenshot shows the 'Environment Entries' configuration page in the WebSphere console. The breadcrumb trail is: Application servers > wpsr01a > Process definition > Control > Environment Entries. Below the breadcrumb, there is a description: 'Use this page to specify an arbitrary name and value pair. The value that is specified system configuration properties.' There are 'New...' and 'Delete' buttons. Below these are icons for selection, copy, paste, and refresh. A table with columns 'Select', 'Name', and 'Value' is shown. The table contains 9 rows of SMF properties, each with a checkbox in the 'Select' column and a value in the 'Value' column. At the bottom, it says 'Total 9'.

| Select | Name | Value |
|--------------------------|--|-------|
| <input type="checkbox"/> | server SMF container activity enabled | 1 |
| <input type="checkbox"/> | server SMF container interval enabled | 1 |
| <input type="checkbox"/> | server SMF interval length | 0 |
| <input type="checkbox"/> | server SMF request activity CPU detail | 1 |
| <input type="checkbox"/> | server SMF request activity enabled | 1 |
| <input type="checkbox"/> | server SMF request activity security | 1 |
| <input type="checkbox"/> | server SMF request activity timestamps | 1 |
| <input type="checkbox"/> | server SMF server activity enabled | 1 |
| <input type="checkbox"/> | server SMF server interval enabled | 1 |

Figure 19-15 SMF properties

Note: A server restart is needed to validate the SMF configuration.

An overview of SMF record type 120 is at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftrrb_SMFrt120overview.html

To gain a better understanding of SMF record type 120, subtype 9, go to the following website:

<http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101342>

After collecting SMF records, you can use the SMF browser to look at the data. The tool is available at the following website:

<https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=zosos390>

You must download the `bbomsmfv.jar` file and upload it to your z/OS file system. It is a UNIX System Services application. It depends on Java to run, so configure your `PATH` variable to point to a Java installation directory.

Note: SMF browser testing was done with Java 5 and later. The browser might work with earlier versions of Java, but this was not tested.

To process the SMF data, first dump the records from the SMF VSAM file to a sequential data set, as shown in Example 19-8.

Example 19-8 Sample job to dump SMF records to sequential data set

```
//SMFDMP JOB (999,POK),'SMFDUMP',MSGLEVEL=(1,1),
// CLASS=A,MSGCLASS=T,NOTIFY=&SYSUID
/*
//STEP EXEC PGM=IFASMFDP
//INDD DD DSN=<HLQ>.<your_dataset>,DISP=SHR
//OUTDD DD DSN=LIST.SMFOUT,DISP=(NEW,CATLG,KEEP),UNIT=SYSDA,
// VOL=SER=TARTS3,SPACE=(CYL,(50,10),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
        INDD(INDD,OPTIONS(DUMP))
        OUTDD(OUTDD,TYPE(120))
/*
```

After generating the sequential data set, go to OMVS in the directory where the `bbomsmfv.jar` file was previously uploaded. Run the command shown in Example 19-9, where:

- ▶ “`INFILE(<dataset name>)`” is the data set generated by the IFASMFDP utility.
- ▶ “`PLUGIN(PERFSUM,<output file>)`” is the file that will be generated by SMF browser analysis.

Example 19-9 SMF browser invocation

```
java -cp bbomsmfv.jar com.ibm.ws390.sm.smfview.SMF "INFILE(LIST.SMFOUT)"
"PLUGIN(PERFSUM,/tmp/smf.out)"
```

Another option for analyzing the data is to run an RMF post processor job to analyze the report produced on dumped SMF records.

RMF

RMF is the strategic IBM product for performance analysis, capacity planning, and problem determination in a z/OS host environment. It has three monitors:

- ▶ Monitor I: Provides long-term collection for system workload and resource utilization
- ▶ Monitor II: Provides on-demand measurements for use in solving immediate problems
- ▶ Monitor III: Provides short-term data collection and online reports for continuous monitoring of system status and solving performance problems

For WebSphere Application Server V8 monitoring, you can use Monitor III reports and combine information with Monitor I post processor to generate workload activity reports. Details are available at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Ftprf_capwar.html

Dispatch progress monitor

Dispatch progress monitor (DPM) provides information, at specified time intervals, about a dispatched request if it is still being processed after the time interval elapses. You can monitor the following protocols:

- ▶ IIOF
- ▶ HTTP
- ▶ HTTPS
- ▶ MDB
- ▶ SIP
- ▶ SIPS

Using this tool, you define a time interval (not too short) to capture data about long running requests. When the time elapses, you can instruct DPM to generate one of the following:

- ▶ SVC dump
- ▶ Java core dump
- ▶ Heap dump
- ▶ Java transaction dump
- ▶ Traceback data

The default time and action are obtained from the WLM classification file (for details about WLM classification, see 17.10, “Tuning workload management on z/OS systems” on page 624”). To enable DPM, use the MODIFY command. Enabling DPM for HTTP protocol shows how to enable DPM for a server, setting the protocol HTTP to five seconds.

Example 19-10 shows the syntax for enabling DPM for HTTP protocol.

Example 19-10 Enabling DPM for HTTP protocol

```
F <server>,DPM,HTTP=5
```

Note: To disable DPM for a specific protocol, set the time to 0.

More details about DPM are located at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Fae%2Ftprf_monitor_dispatch_requests.html

19.4.7 Summary of monitoring tips

The following summary lists a simple set of best practices about how to establish a useful monitoring environment:


- ▶ Take the time to understand the applications deployed into the environment. Use this knowledge to plan for the kinds of metrics that are beneficial to understand application performance.
- ▶ Activate monitoring at the planned level in all testing environments, especially when benchmarking. Although it is vital to your ability to understand an environment, monitoring is not free and uses CPU, memory, and other resources. Monitoring can impact capacity planning.
- ▶ Use monitoring to understand normal operations and gain an appreciation of what is normal for your systems.
- ▶ Check for differences in your systems after all release changes, especially if full performance testing and benchmarks were not re-established.
- ▶ Use the basic metrics of PMI as a good starting set of metrics, but customize them to your needs.

Monitoring WebSphere Application Server V8.5 alone is not enough. Application server monitoring needs to be part of an overall monitoring strategy



Part 5

Working with applications



Features for application development and deployment

This chapter provides an overview of the features that are available for application development and deployment in WebSphere Application Server. WebSphere Application Server V8.5 supports Java Platform, Enterprise Edition (Java EE) V6, and it has backwards compatibility with older versions. Programming models that were available as features packs in versions 7.0 and 8.0 are now part of V8.5. In addition, this chapter briefly describes some deployment and development tools for WebSphere Application ServerV8.5.

This chapter covers the following topics:

- ▶ Java Enterprise Edition 6 support
- ▶ Integrated standards-base programming models and extensions
- ▶ Monitored directory support
- ▶ Development and deployment tools
- ▶ Development and deployment tools

20.1 Java Enterprise Edition 6 support

Java EE 6 expands the developer value that was introduced in Java EE 5 and continues to focus on developer productivity and ease-of-use enhancements. Version 6 includes the following new features:

- ▶ Support for the EJB 3.1 specification
- ▶ Support for the Context and Dependency Injection (CDI) 1.0 specification at a runtime level that uses the Apache OpenWebBeans 1.x implementation
- ▶ Java Persistence API (JPA) 2.0
- ▶ Java Servlet 3.0
- ▶ Java API for RESTful Web Services (JAX-RS) 1.1
- ▶ JavaServer Faces (JSF) 2.0
- ▶ JavaServer Pages (JSP) 2.2
- ▶ Bean Validation 1.0
- ▶ Java Architecture for XML Binding (JAXB) 2.2
- ▶ Enterprise Web Services 1.3
- ▶ Java API for XML-Based Web Services (JAX-WS) 2.2
- ▶ Java EE Connector Architecture 1.6

Table 20-1 includes Java EE and JAVA SE versions supported by WebSphere Application Server from Version 6.1 to Version 8.5.

Table 20-1 Java EE support

| Specification or API | WebSphere Application Server version | | | |
|--|--|--|-----------------------------------|----------------------------------|
| | Version 8.5 | Version 8.0 | Version 7.0 | Version 6.1 |
| Java Platform, Enterprise Edition (Java EE) specification Prior to Java EE 5, the specification name was Java 2 Platform, Enterprise Edition (J2EE) | Java EE 6 (JSR 316), Java EE 5, J2EE 1.4, and J2EE 1.3 | Java EE 6 (JSR 316), Java EE 5, J2EE 1.4, and J2EE 1.3 | Java EE 5, J2EE 1.4, and J2EE 1.3 | J2EE 1.4, J2EE 1.3, and J2EE 1.2 |
| Java Platform, Standard Edition (Java SE) specification Prior to Java SE 6, the specification name was Java 2 Platform, Standard Edition (J2SE) | Java SE 6 Java SE 7 (Optional) | Java SE 6 | Java SE 6 | J2SE 5 |

For more information about JEE 6.0 support, the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=rovr_specs

20.2 Integrated standards-base programming models and extensions

Many of the core programming models in WebSphere Application Server V8.5 were available through feature packs in Versions 7.0 and 8.0. Now, these programming models are built into WebSphere Application Server V8.5. This section describes the available programming models:

- ▶ Session Initiation Protocol applications
- ▶ WebSphere Batch programming model
- ▶ OSGi applications programming model
- ▶ Communications enabled applications
- ▶ Service Component Architecture programming model
- ▶ Extensible Markup Language programming model
- ▶ Integrated Web Services support
- ▶ Support for integrated IBM WebSphere Application Server Web 2.0 and Mobile Toolkit
- ▶ Simplified development of server-side REST applications using Java API for RESTful Web Services
- ▶ IBM WebSphere SDK Java Technology Edition Version 7.0

20.2.1 Session Initiation Protocol applications

Session Initiation Protocol (SIP) applications are Java programs that use at least one SIP servlet written to the JSR 116 specification. WebSphere Application Server V8.5 also supports SIP Servlet Specification 1.1, also referred to as JSR 289. SIP is used to establish, modify, and terminate multimedia IP sessions. SIP negotiates the medium, the transport, and the encoding for the call. After the SIP call is established, the communication takes place over the specified transport mechanism, independent of SIP. Examples of application types that use SIP are voice over IP (VOIP), click-to-call, and instant messaging.

Session Initiation Protocol (SIP) applications are packaged as SIP archive (SAR) files, and are deployed to the application server using the standard WebSphere Application Server administrative tools. SAR files can also be bundled in a Java EE application archive (EAR file), similar to other Java EE components.

In the application server, the web container and SIP container are converged and are able to share session management, security, and other attributes. High availability of these converged applications is made possible because of the integration of HTTP and SIP in the base application server. For more information about SIP applications, see the following resources:

- ▶ JSR 289 SIP Servlet API 1.1 Specification, found at the following website:
<http://www.jcp.org/aboutJava/communityprocess/final/jsr289/index.html>
- ▶ JSR 116, found at the following website:
<http://jcp.org/en/jsr/detail?id=116>
- ▶ RFC 3261, found at the following website:
<http://www.ietf.org/rfc/rfc3261.txt>

20.2.2 WebSphere Batch programming model

WebSphere Batch provides a transactional batch programming model (large number of small and repetitive operations) and a compute-intensive programming model (small number of CPU/Memory intensive operations). Both the transactional batch and compute-intensive programming models are implemented as Java objects. They run as background jobs, described by a job control language and are supported by infrastructure components that aim to support batch workloads.

The control language for batch jobs is called XML job control language (xJCL). The xJCL allows users to describe the job steps involved in a batch job. The application runs in batch containers that run in designated WebSphere Application Server environments. The batch container ultimately processes a job definition and carries out the lifecycle of a job.

Figure 20-1 shows a typical batch container.

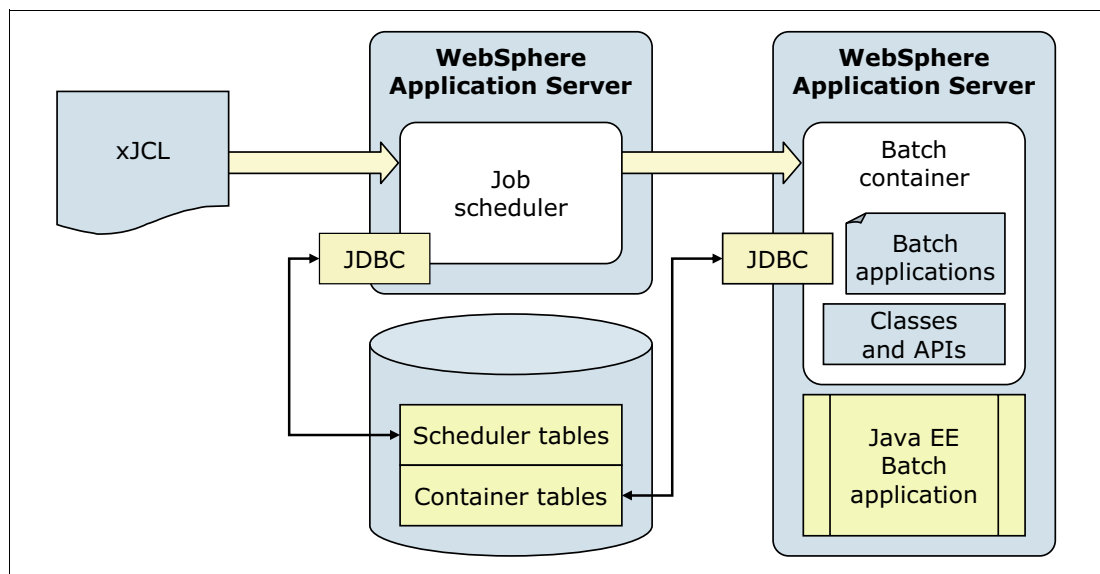


Figure 20-1 Batch container

The batch container provides the following services:

- ▶ Checkpointing, which involves resuming batch work from a selected position.
- ▶ Result processing, which involves intercepting and processing step and job return codes.
- ▶ Batch data stream management, which involves reading, positioning, and repositioning data streams to files, relational databases, native z/OS data sets, and many other different types of input and output resources.

Figure 20-2 depicts a typical batch workflow.

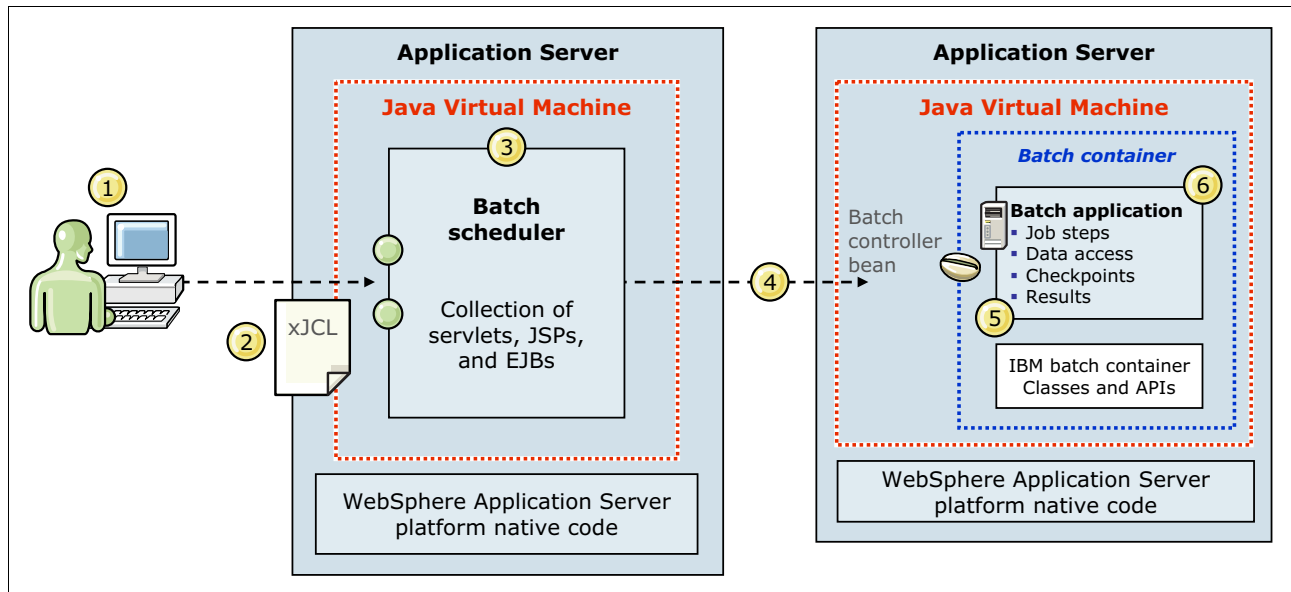


Figure 20-2 Batch workflow

The batch workload, shown in Figure 20-2 proceeds as follows:

1. Batch jobs are submitted to the system using the Job Management Console or programmatically by Enterprise Java Beans (EJB), Java Message Service (JMS), or web services.
2. Each job is submitted in the form of an XML Job Control Language (xJCL) document.
3. The batch scheduler analyzes the request.
4. The job is dispatched to the best endpoint for job execution based on several different metrics.
5. The endpoint sets up the jobs in the batch container and begins executing the batch steps based on the definitions in the xJCL.
6. The batch application is invoked.

The Job Dispatcher aggregates job logs and provides lifecycle management functions, such as start, stop, cancel, and so on.

Features in WebSphere Application Server for WebSphere Batch

In addition to fully incorporating WebSphere batch functions into WebSphere Application Server V8.5, the following new enhancements are included:

- ▶ *Parallel batch*, which has the ability to split the work and process jobs as multiple subordinate jobs concurrently.
- ▶ *Enterprise integration*, which allows for management of Batch Jobs from an external workload scheduler tool, such as Tivoli Workload Scheduler.
- ▶ *Cobol support*, which allows the usage of Cobol modules in WebSphere applications.
- ▶ *CommandRunner* utility job step, which allows shell command lines (including scripts and compiled programs) to be run as job steps.

More information about WebSphere Batch is in Chapter 21, “WebSphere Batch” on page 751 and at the following websites:

- ▶ Chapter 6 of *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022:

<http://www.redbooks.ibm.com/redpieces/pdfs/sg248022.pdf>

- ▶ More information about COBOL support is in Chapter 16 of *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022:

<http://www.redbooks.ibm.com/redpieces/pdfs/sg248022.pdf>

20.2.3 OSGi applications programming model

OSGi applications are modular applications that use both Java EE and OSGi technologies. You can design and build applications and suites of applications from coherent, versioned, and reusable OSGi modules that are accessed only through well-defined interfaces. This enables the same, or different, applications to use different versions of the same third-party libraries without interference.

OSGi applications allow the composition of isolated enterprise applications using multiple, multi-version bundles that have dynamic life cycles. Application maintenance and upgrades can be simplified using standard OSGi mechanisms to simultaneously load multiple versions of classes in the same application.

The OSGi applications support in WebSphere Application Server includes the following major capabilities:

- ▶ Use the OSGi Service Platform Release 4 Version 4.2 Enterprise Specification Blueprint Container for declarative assembly of components. The container simplifies unit tests outside of the application server.
- ▶ Use extensions to the Blueprint component model for declarative transactions and container-managed Java Persistence API (JPA).
- ▶ Develop OSGi application projects using IBM Rational Application Developer, which enforces OSGi visibility rules. These rules allow projects to only access packages from other projects that explicitly declare them as part of the project externals. This provides environmental support to development best practices.
- ▶ Compose isolated enterprise applications using multiple, versioned bundles with a dynamic lifecycle.
- ▶ Deploy applications in archive files that contain only application-specific content and metadata that points to shared bundles. This means that application archive files can be smaller. It also means that, when a library is shared by several OSGi applications, only one copy of the library is loaded into memory.
- ▶ Extend and scale running applications as business needs change without changing the underlying application.
- ▶ Update a running application only impacting those bundles affected by the update.
- ▶ Use an integrated bundle repository, and configure the locations of external repositories, to support the provisioning of bundles to applications.
- ▶ Deploy existing web application archive (WAR) files as web application bundles (WABs). This allows web applications to use the OSGi module system.
- ▶ Deploy web applications that use Version 3.0 of the Java Servlet Specification.
- ▶ Deployment of EJB in OSGi bundles (*New in V8.5*).

- ▶ Enhanced security provided by bean security in the Blueprint XML file of OSGi Applications (*New in V8.5*).
- ▶ Simultaneously load multiple versions of classes in the same application using standard OSGi mechanisms.
- ▶ Administratively update deployed applications in a modular fashion, at the bundle-level.
- ▶ Deploy applications that use their own versions of common utility classes, distinct from the versions that are used by the server runtime environment. This can be done without configuring application Java EE class loader policies, such as PARENT_LAST mode.
- ▶ Use federated lookup mechanisms between the local Java Naming and Directory Interface (JNDI) and the OSGi service registry.

For more information about OSGi applications, review Chapter 26, “Working with OSGi applications” on page 921 and the following resources:

WebSphere Application Server V8.5 Concepts, Planning, and Design Guide, section 11.2.2:

<http://www.redbooks.ibm.com/redpieces/pdfs/sg248022.pdf>

Overview of the purpose of OSGi:

<http://www.youtube.com/watch?v=J2wq0Y603-Q>

The OSGi home page:

<http://www.osgi.org/About/HomePage>

20.2.4 Communications enabled applications

The Communications Enabled Applications (CEA) support in WebSphere Application Server V8.5 allows you to add dynamic web communications to any application or business process. CEA provides Representational State Transfer (REST) and web service interfaces to enable existing applications to take advantage of communication features involving phone calls and web collaboration.

With the CEA capability, enterprise solution architects and developers can use a single core application to enable multiple modes of communication. CEA applications do not require developers to have extensive knowledge of telephony or SIP. CEA capabilities deliver call control, notifications, and interactivity, providing the platform for more complex communications.

Using this simplified programming model for adding web-based communications, enterprise developers can perform the following tasks:

- ▶ Quickly add communications support to any application, for example, click-to-call integration.
- ▶ Enable shared sessions between users and company representatives.
- ▶ Push relevant session data for application use, for example, customer phone numbers.
- ▶ Deliver automated notifications and instant messaging support.
- ▶ Provide enterprise-grade security, scalability, and high availability.
- ▶ Integrate with customer private branch exchange (PBX) systems.

Deprecated feature: CEA Dojo widgets encapsulate various user interfaces that are based on the CEA REST API. These widgets are deprecated. You can also obtain the source for these widgets from IBM developerWorks®.

For more information about Administering communications enabled application, visit the Information Center at:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=welc6tech_cea

20.2.5 Service Component Architecture programming model

Service Component Architecture (SCA) is a set of specifications that constitute a programming model for building applications using an SOA. SCA extends other SOA technologies, such as web services, and provides a platform and language-neutral component model that is based on open standards specified by the Open SOA Collaboration. SCA allows the creation of complex composite applications based on previously existing service components.

The features of SCA support include:

- ▶ POJO (Java Object) service-component implementations, including support for annotations
- ▶ Asynchronous capability
- ▶ Recursive composition model support
- ▶ Support for SCA services developed from existing WSDL files or Java code
- ▶ Deployment of SCA composites in business-level applications
- ▶ SCA authorization and security identity policies
- ▶ PassByReference optimization for SCA applications
- ▶ Several binding types, including web services binding, SCA default binding, Enterprise JavaBeans (EJB), Java Message Service (JMS), Atom, and HTTP bindings
- ▶ Support for Java Architecture for XML Binding (JAXB) data bindings in SCA applications
- ▶ SCA annotations for Java EE web modules, session beans, and message-driven beans
- ▶ Preview of native SCA deployment
- ▶ Spring 2.5.5 containers in SCA applications
- ▶ OSGi applications as SCA implementations
- ▶ Service Data Objects 2.1.1
- ▶ Support for SCA OASIS programming model implementation
- ▶ Sample SCA composites compiled specifically for use with the product

More information about SCA is in chapter Chapter 25, “Working with SCA applications” on page 901 and at the following resource:

WebSphere Application Server V8.5 Concepts, Planning, and Design Guide, section 11.2.1:

<http://www.redbooks.ibm.com/redpieces/pdfs/sg248022.pdf>

20.2.6 Extensible Markup Language programming model

Extensible Markup Language (XML) structured data has become the predominant format for data interchange. XML data is navigated, queried, or transformed in almost every existing application that runs on WebSphere Application Server V8.5. This release delivers critical technology that provides application developers with support for the following key World Wide Web Consortium (W3C) XML standards:

- ▶ Extensible Stylesheet Language Transformations (XSLT) 2.0
- ▶ XML Path Language (XPath) 2.0
- ▶ XML Query Language (XQuery) 1.0

These W3C XML standards offer application developers numerous advanced capabilities for building XML applications. WebSphere Application Server V8.5 support for XML has the following key features and capabilities:

- ▶ An XML application development environment tuned for developer productivity
- ▶ An XML runtime API that offers consistent execution and data navigation API that allows access to existing Java logic
- ▶ The ability to query large amounts of data stored in XML outside of a database with XQuery 1.0
- ▶ Optimum XML-application performance
- ▶ XML-application reliability with support for XML schema-aware processing and validation
- ▶ 40+ preconfigured samples including four end-to-end scenarios

20.2.7 Integrated Web Services support

WebSphere Application Server V8.5 supports web services that are developed and implemented based on the Web Services for Java Platform, Enterprise Edition (Java EE) specification, V1.3. This specification supports WSDL Version 1.1, SOAP Version 1.1 and SOAP V1.2. The application server supports the Java API for the XML Web Services (JAX-WS) programming model and the Java API for XML-based RPC (JAX-RPC) programming model.

Java Architecture for XML Binding (JAXB) 2.2 provides a convenient way to map Java classes and XML schema for simplified development of web services. Version 2.2 provides minor enhancements to its annotations for improved schema generation and better integration with Java API for XML-based web services.

JAX-WS 2.2 simplifies the development of web services with more platform independence for Java applications by the use of proxies and Java annotations. JAX-WS 2.2 requires JAXB 2.2.

20.2.8 Support for integrated IBM WebSphere Application Server Web 2.0 and Mobile Toolkit

The WebSphere Application Server Web 2.0 and Mobile Toolkit simplifies the addition of Asynchronous JavaScript and XML (AJAX) rich desktop and mobile user interfaces and REST web services to Java web applications. Web 2.0 capabilities, such as AJAX and REST, help application developers to create more connected, interactive applications, that result in higher customer satisfaction, user productivity, and enhanced decision making. New mobile AJAX components enable developers to create mobile web applications that run on devices, such as smart phones and tablets.

20.2.9 Simplified development of server-side REST applications using Java API for RESTful Web Services

Java API for RESTful Web Services (JAX-RS) offers a simpler way to develop, consume, and scale REST applications. It is composed of a collection of interfaces and Java annotations that simplifies the development process. With the annotations, you can declare resource classes and the data types they support. It also allows developers to gain access to the runtime context. Through its extensible framework, it is also possible to integrate custom content handlers.

20.2.10 IBM WebSphere SDK Java Technology Edition Version 7.0

WebSphere Application Server V8.5 supports IBM WebSphere Software Development Kit (SDK) Java Technology Edition Version 7.0 as a pluggable JDK. Java 6 is installed with the product and used by default. Java 7 can be optionally installed and enabled using the *managesdk* tool. This IBM SDK provides a full-function SDK for Java that is compliant with the Java Platform, Standard Edition (Java SE) 7 application programming interfaces (APIs). The SDK contains the Java application *Runtime Environment* and other tools that enable developers to create Java applications.

For more information, refer to the IBM SDK Java Technology Edition V7 Information Center:
<http://publib.boulder.ibm.com/infocenter/java7sdk/v7r0/index.jsp>

20.3 Monitored directory support

Simply by dragging and dropping applications into a defined and monitored directory, you can speed the process of editing, compiling, deploying, debugging, updating, and uninstalling applications. When an application is moved into the directory, after a defined interval, it is automatically installed and started. Likewise, if the application is removed from the directory, it is stopped and uninstalled. If the application or module is moved into the directory again, it is updated. The supported file types are:

- ▶ Enterprise Archive (EAR)
- ▶ Web Application Archive (WAR)
- ▶ Java Archive (JAR)
- ▶ SIP Application Resource (SAR)

More information about Monitored directories is in 23.7.2, “Deploying using the monitored directory support feature” on page 860.

20.4 Development and deployment tools

This section discusses the development and deployment tools that are available for WebSphere Application Server V8.5.

20.4.1 IBM Assembly and Deploy Tools for WebSphere Administration

The IBM Assembly and Deploy Tools for WebSphere Administration is targeted to help in the assembly and deployment of applications only. It does not provide development capabilities. The IBM Assembly and Deploy Tools have the following key components:

- ▶ Import and validate applications
- ▶ Edit deployment descriptors and binding files
- ▶ Edit EAR-level configuration (Enhanced EAR)
- ▶ Create and debug Jython and `wsadmin` scripts
- ▶ Deploy EJB and web services
- ▶ Deploy applications to local or remote WebSphere Application Server V8.5 servers
- ▶ Debug applications on WebSphere Application Server V8.5

20.4.2 WebSphere Application Server Developer Tools for Eclipse

The IBM WebSphere Application Server Developer Tools for Eclipse is a lightweight set of tools for developing, assembling, and deploying Java EE, OSGi, Web 2.0 and mobile applications. The tool supports WebSphere Application Server V8.5 (including the Liberty profile), WebSphere Application Server V8.0, and WebSphere Application Server V7.0. In combination with the WebSphere Application Server V8.5 Liberty profile, this tool provides a fast and lightweight environment for the rapid development and unit testing of web, Web 2.0, mobile, and OSGi applications.

For more information, refer to the Information Center at:

http://publib.boulder.ibm.com/infocenter/radhelp/v9/topic/com.ibm.rad.install.doc/topics/wdt_overview.html

20.4.3 Rational Application Developer for WebSphere Software

Rational Application Developer for WebSphere Software offers a more extensive set of tools that support enterprise development. IBM Rational Application Developer for WebSphere Software can be used to design, develop, analyze, test, profile, and deploy high-quality web, SOA, Java, Java EE, and portal applications.

This product includes the following features:

- ▶ Fully-integrated tools and support for IBM WebSphere Application Server V6.1 through V8.5
- ▶ Tools, including many simple wizards and visual editors, that fully support the Java EE programming model, including web, Java, web services, and EJB applications
- ▶ Code quality, testing, and deployment tools, such as the enhanced runtime analysis to detect memory leaks or thread locks
- ▶ Web 2.0, OSGi, Java Persistence API 2.0, SCA, XML, CEA, portal and web services development features
- ▶ IBM Workload Deployer (cloud) support
- ▶ Support for Java 7
- ▶ Ant scripting and JUnit testing framework
- ▶ WebSphere performance profiling and logging
- ▶ Agile development support with tools for refactoring code and unit testing
- ▶ Automated tools to manage server instances and server configurations, including automated creation and submission of `wsadmin` scripts
- ▶ Integration with IBM Rational Team Concert™ and IBM Rational ClearCase® so that management operations can be performed within the development environment and increase collaboration and team productivity

- ▶ WebSphere Adapter Support for third-party products, such as SAP, PeopleSoft Enterprise, Siebel, Oracle E-Business Suite, and JD Edwards
- ▶ Unified Modeling Language (UML) modeling function
- ▶ Support for batch and Java Persistence API development
- ▶ Support for development for the Liberty profile



WebSphere Batch

This chapter discusses the concepts and features of WebSphere Batch for WebSphere Application Server V8.5 and includes a scenario that illustrates how to work with WebSphere Batch.

WebSphere Batch provides a comprehensive execution environment for efficient Java batch processing, including new key aspects, such as the ability to run batch jobs in parallel by running Java batch inside WebSphere Application Server, support for COBOL, and a unified batch architecture for the enterprise.

This chapter provides an overview of WebSphere Batch and includes the following topics:

- ▶ Overview of WebSphere Batch
- ▶ Batch programming models
- ▶ Configuring the batch environment
- ▶ Example: Working with batch applications

21.1 Overview of WebSphere Batch

Batch applications are designed to execute long and complex transaction processing that typically executes computationally intensive work. This type of processing requires more resources than traditional online transactional processing (OLTP) systems. Batch applications run as background jobs described by a job control language and use a processing model based on submit, work, and result actions. The execution of batch processes can take hours and the tasks are typically transactional, involving multi-step processes.

WebSphere Application Server V8.5 with WebSphere Batch supplies a unified batch architecture. Using XML job control language (xJCL), WebSphere Batch provides consistent programming and operational models. WebSphere Batch makes use of a batch technology that is optimized for Java and supports long-running applications. Ensuring agility, scalability, and cost efficiency for enterprises.

21.1.1 Batch jobs

A batch job consists of a series of definitions that direct the execution of one or more batch applications and specifies their input and output. A batch job performs a specific set of tasks in a predefined sequence to accomplish specific business functionalities.

Batch job workloads are executed in a batch container in WebSphere Application Server environments. This batch container is the main engine responsible for the execution of batch applications. It runs batch jobs under the control of an asynchronous bean, which can be thought of as a container-managed thread. The batch container ultimately processes job definitions and carries out the lifecycle of jobs.

WebSphere runs batch applications that are written in Java and implements a WebSphere batch programming model. They are packed as EAR files and are deployed to the batch container hosted in an application server or cluster. Batch applications are executed non-interactively in the background.

Batch applications implement one of two programming models:

- ▶ Transactional batch

These applications handle large amounts of work based on repetitive tasks, such as processing a large number of records.

- ▶ Compute-intensive applications

Compute-intensive applications perform work that requires large amounts of system resources, in particular CPU and memory. The application is responsible for providing all of the logic for performing the necessary work.

Batch jobs can perform a series of tasks that are a combination of transactional and compute-intensive tasks to complete the execution of a batch application.

21.1.2 Batch applications

Batch applications are programs designed to execute non-interactive tasks in the background. Input and output is generally accessed as logical constructs by the batch application and are mapped to concrete data resources by the batch job definition.

Batch applications are Java EE applications consisting of Plain Old Java Objects (POJOs). These applications conform to a few well-defined interfaces that allow the batch runtime to manage the start of batch jobs designed for the application.

Batch work is expressed as jobs, which are made up of steps that are processed sequentially.

All jobs contain the following information:

- ▶ The identity of the batch application that performs the work
- ▶ One or more job steps that must be performed to complete work
- ▶ The identity of an artifact within the application that provides the logic for each job step
- ▶ Key and value pairs for each job step that provide additional context to the application artifacts

Jobs for batch applications contain additional information specific to the batch programming mode:

- ▶ Definitions of sources and destinations of data
- ▶ Definitions of checkpoint algorithms

21.1.3 Elements of the batch environment

A typical batch environment consists of a job scheduler, batch container, batch applications, jobs, interfaces for management functions, and database tables, as shown in Figure 21-1.

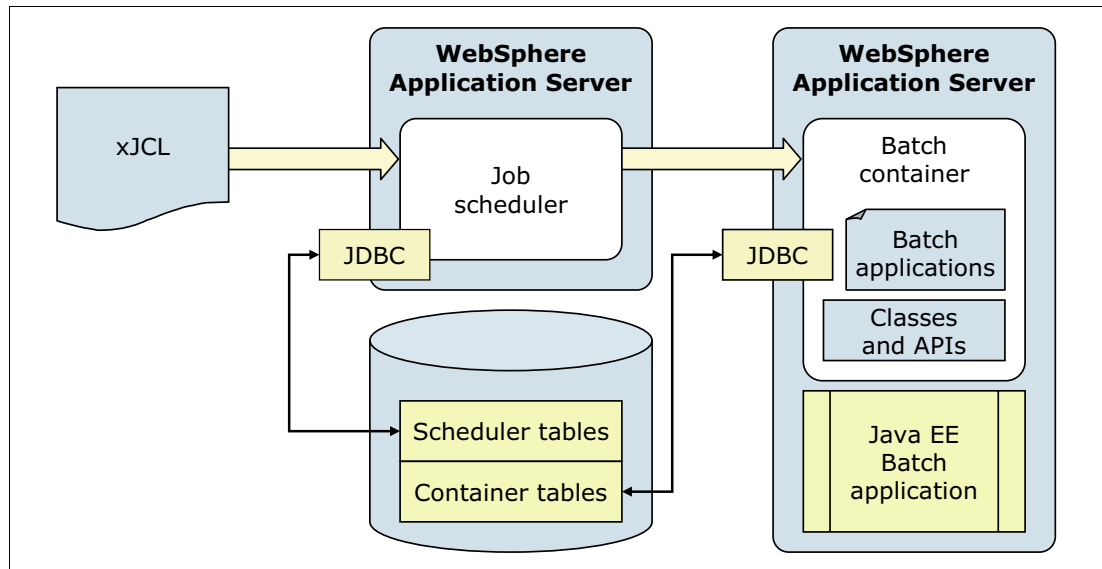


Figure 21-1 The batch elements

The following list describes the elements of a batch environment:

- ▶ Job scheduler

The job scheduler is the batch component that is hosted in an application server or in an application server cluster. It provides all job management functions, such as submit, cancel, and restart. It also maintains a history of all jobs, including those waiting to run, those running, and those having already run. It accepts and schedules the execution of batch jobs, manages the job database, assigns job IDs, and selects jobs to run.

The job scheduler exposes three API types to access its management functions:

- A web interface called the job management console

The job management console provides a graphical user interface (GUI) with which batch administrators and submitters can perform job management functions that include submitting a job, viewing jobs, canceling or suspending a job, and resuming a suspended job.

- A shell command called `lrcmd`

With the command-line interface, you can submit and control the batch jobs in the system.

- APIs, available as either web services or EJBs.

High availability: Each cell holds exactly one job scheduler. You can eliminate a single point of failure for the job scheduler by enabling it in an application server cluster within the cell.

- ▶ **Batch container**

The batch container processes job definitions and carries out the lifecycle of a job. It provides the execution environment for batch jobs, under the control of asynchronous beans, similar to container-managed threads and provides application services, such as checkpoint or restart and job-logging. Java EE-based batch applications run inside the batch container, which is hosted in an application server.

The batch container is hosted in an application server or cluster. A cell can include multiple batch containers.

The batch container provides the support to run jobs with transactions and the ability to checkpoint and restart these applications, as required. For that reason, the batch container uses a relational database to store checkpoint information for transactional batch applications.

- ▶ **Java EE batch application**

Java EE batch applications are regular Java EE applications that are deployed as Enterprise Archive (EAR) files, which contain implementations of one or more Java batch applications. These Java batch applications follow either the transactional batch or compute-intensive programming models.

- ▶ **xJCL**

Jobs are described using an XML dialect called XML Job Control Language (xJCL). This dialect has constructs for expressing all of the information needed for both compute-intensive and batch jobs, although some elements of xJCL are only applicable to compute-intensive or batch jobs. The job description identifies which application to run and its input and output.

The xJCL definition of a job is not part of the batch application. This definition is constructed separately and submitted to the job scheduler to run. The job scheduler uses information in the xJCL to determine where and when the job runs.

- ▶ **Scheduler tables**

The job scheduler uses a relational database to store job information. It can be any relational database supported by WebSphere Application Server. If the job scheduler is clustered, the database must be a network database, such as DB2.

- ▶ Container tables

The batch container uses a relational database to store checkpoint information for transactional batch applications. The database can be any relational database supported by WebSphere Application Server and is accessed using JDBC. If the batch container is clustered, the database must be a network database, such as DB2.

- ▶ Grid endpoints

The grid endpoints are application servers that are augmented to provide a special runtime environment needed by batch applications. This runtime environment is provided by a product-provided Java EE application, the batch execution environment. This application is deployed automatically by the system when a batch application is installed, and it serves as an interface between the job scheduler and batch applications. It provides the runtime environment for both compute-intensive and transactional batch applications.

21.2 Batch programming models

The transactional batch and compute-intensive programming models are both implemented as Java objects. They are packaged into an enterprise archive (EAR) file for deployment into the application server environment. The individual programming models provide details about how the lifecycle of the application and jobs submitted to it are managed by the grid endpoints. Central to all batch applications is the concept of a job to represent an individual unit of work to be run.

You can mix transactional batch, compute intensive, and native execution job steps. The run time uses a controller that is the same for every job, regardless of the type of steps that the job contains. The controller runs the appropriate logic for the step. These different job step types can also be run in parallel.

For more information about WebSphere Batch programming models, see chapter 6 in *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022.

21.2.1 Transactional batch programming model

Batch applications are Enterprise JavaBeans (EJB) based Java EE applications. These applications conform to a few well-defined interfaces that allow the batch runtime environment to manage the start of batch jobs destined for the application.

Figure 21-2 on page 756 shows the batch programming model principal components, followed by a brief overview of each of these components.

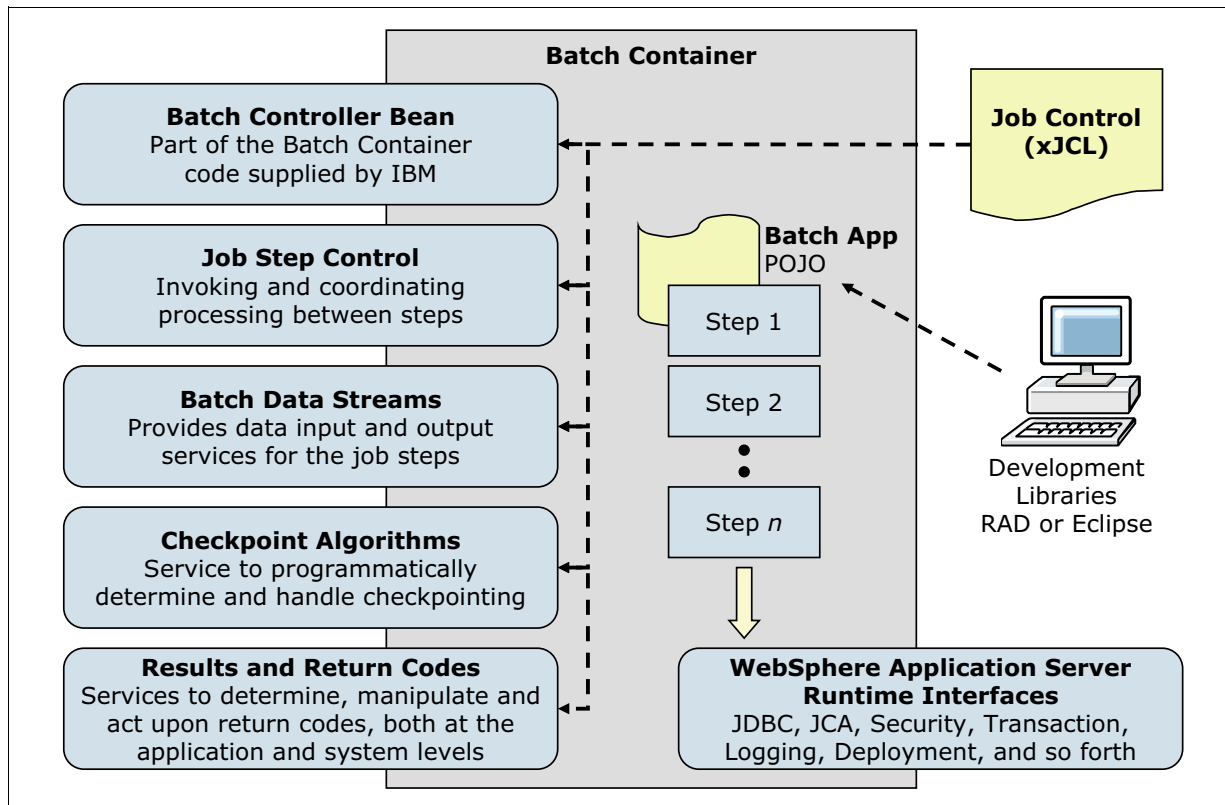


Figure 21-2 Batch programming model

► Batch controller bean

A batch application includes a stateless session bean that the product run time provides. This stateless session bean acts as a job step controller and each application can include only a single controller bean.

The implementation of this bean is provided by the WebSphere Application Server product and not by the batch application. The bean must be declared in the batch application deployment descriptor. The resource references and EJB references declared on the controller bean are available to batch data streams of the batch application in which the controller bean is declared. For example, if a batch data stream in the application needs access to a data source, a resource reference to that data source can be declared on the controller bean, and the batch data stream can look up the data source at run time.

For more information about the batch controller bean, visit the WebSphere Application Server V8.5 Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/cgrid_xdbatchcb.html

► Batch job steps

A batch job can be composed of one or more batch steps. All steps in a job are processed sequentially. Dividing a batch application into steps allows for separation of distinct tasks in a batch application. Batch steps are implemented as POJO classes that implement the interface `com.ibm.websphere.batch.BatchJobStepInterface`. This interface provides the business logic of the batch step. Typically, a batch step contains code to read a record from a batch data stream, perform business logic with that record, and then continue to read the next record. The `processJobStep` method of a batch step class is called by the

grid endpoints in a batch loop. This method contains all of the logic that can be batched to perform on data.

The grid endpoints invoke batch step class methods in a global transaction. This global transaction is managed by the grid endpoints. The behavior of the transaction, such as transaction timeout or transaction commit interval, is controlled by the checkpoint algorithm associated with the batch job to which the step belongs.

For more information about batch job steps, visit the WebSphere Application Server V8.5 Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/cgrid_xdbatchstp.html

► **Batch data streams**

Methods on the BatchDataStream interface allow the batch runtime environment to manage the data stream being used by a batch step. For example, one of the methods retrieves current cursor information from the stream to track how much data was processed by the batch step.

Batch data streams (BDS) are Java objects that provide an abstraction for the data stream processed by a batch step. A batch step can have one or more BDS objects associated with it. The grid endpoints make the BDS associated with the batch step available at run time. The grid endpoints also manage the lifecycle of a BDS by invoking batch-specific callbacks.

A BDS object implements the `com.ibm.websphere.batch.BatchDataStream` interface. This interface is server agnostic. The implementing object can retrieve data from any type of data source, for example, files and databases. Call back methods on the BatchDataStream interface allow the grid endpoints to manage the BDS at run time. One of the key features of a BDS is its capability to convey its current position in the stream to the grid endpoints and the capability to move itself to a given location in the data stream. This feature allows the grid endpoints to record (in the grid endpoints database) how much data a batch step processed. This information is recorded on every checkpoint. Therefore, the grid endpoints can restart a batch job from a recorded position in the data stream if the job is canceled or fails in a recoverable manner.

For more information about batch data streams, visit the WebSphere Application Server V8.5 Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/cgrid_xdbatchds.html

► **Checkpoint algorithms**

The batch runtime environment uses checkpoint algorithms to decide how often to commit global transactions under which batch steps are started. The xJCL definition of a batch job defines the checkpoint algorithms to be used and the properties that define the checkpoint behavior, such as transaction timeouts and checkpoint intervals. You can apply a different algorithm to each batch step but can apply no more than one checkpoint algorithm to a batch step.

On each batch step iteration, the common batch container consults the checkpoint algorithm applied to that step to determine if it must commit the global transaction. Call back methods on the checkpoint algorithms allow the common batch container to inform the algorithm when a global transaction is committed or started. This behavior enables the algorithm to track the global transaction lifecycle.

WebSphere Application Server V8.5 supports two checkpoint algorithms:

- Time-based algorithm

The time-based checkpoint algorithm commits global transactions at a specified time interval.

- Record-based algorithm

The record-based checkpoint algorithm commits global transactions at a specified number of iterations of the processJobStep method of batch step. Each call to the processJobStep method is treated as iterating through one record.

A checkpoint algorithm service provider interface (SPI) is also provided for building additional custom checkpoint algorithms.

For more information about checkpoint algorithms, visit the WebSphere Application Server V8.5 Information Center:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/cgrid_xdbatchalg.html

- ▶ Results algorithm

Results algorithms are an optional feature of the batch programming model. Results algorithms are applied to batch steps through xJCL. The algorithms manipulate the return codes of batch jobs. Additionally, these algorithms are place holders for triggers based on step return codes. Results algorithms allows for two types of actions to occur at the end of a batch step:

- To influence the return code of the batch job based on the return code of the batch step that just ended. There are two types of return codes: The return code of an individual batch step and the return code of the batch job to which the step belongs.
- To provide a place holder for triggers or actions to take based on various step return codes.

At the end of a batch step, the grid endpoints check the xJCL of the batch job to determine which results algorithm to invoke. For each results algorithm specified, the grid endpoints pass to the algorithm the return code of the batch step. The results algorithm can then act based on the return codes passed in. The algorithm then passes a return code for the batch job back to the grid endpoints, which is persisted to the grid endpoints database as the current return code of the batch job. This return code can be the same as the return code that the grid endpoints passed to the results algorithm initially, or the return code can be different, depending on logic coded into the results algorithm. If a results algorithm is not specified on a batch step, the job return code is that of the results algorithm from the previous step. If no results algorithms are specified, the job return code is zero (0).

For more information about results algorithms, visit the WebSphere Application Server V8.5 Information Center:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/cgrid_xdbatchres.html

- ▶ Batch job return codes

Batch job return codes fall into two groups, system and user. System return codes are defined as negative integers. User application return codes are defined as positive integers. Both system and user ranges include the return code of zero (0). If a user application return code is specified in the system return code range, a warning message is posted in the job and system logs.

Table 21-1 on page 759 lists the system batch job return codes that the batch environment uses.

Table 21-1 Batch job return codes

| Return code | Explanation |
|-------------|---|
| 0 | Job ended normally |
| -1 | Internal protocol error - WSGrid utility |
| -2 | Input parameter error - WSGrid utility |
| -4 | Job was suspended |
| -8 | Job was canceled |
| -10 | Job was forcibly canceled (z/OS only) |
| -12 | Job failed and is in restartable state |
| -14 | Job failed and is in execution failed state |
| -16 | Catastrophic failure - WSGrid utility |

There are two options that are used to report an error in a batch application. The first option is for the application to produce an exception when an error is encountered. This results in termination of the job with a batch job return code of -12 and a batch job status of restartable. The second option is for the application to return a `BatchConstants.STEP_COMPLETE_EXECUTION_FAILED` return code from the `processJobStep` method and return an application-specific error return code from the `destroyJobStep` method. This results in termination of the job and a batch job status of execution failed.

For more information about results algorithms, visit the WebSphere Application Server V8.5 Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/cgrid_cgreturncodes.html

► xJCL

The WebSphere Batch uses xJCL to describe jobs that identify the batch application to run. The xJCL language definition is similar to traditional JCL, and it has constructs for expressing all of the information needed for both compute-intensive and batch jobs. The xJCL definition of a job is not part of the batch application.

The job definition identifies which batch application to run and its inputs and outputs. It also identifies which checkpoint algorithms and results algorithms to use. The job scheduler uses information in the xJCL to determine where and when the job runs. Example 21-1 shows a sample xJCL.

Example 21-1 xJCL sample

```
<job name="PostingsSampleEar"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <jndi-name>ejb/com/ibm/websphere/samples/PostingsJob</jndi-name>
    <step-scheduling-criteria>
        <scheduling-mode>sequential</scheduling-mode>
    </step-scheduling-criteria>
    <checkpoint-algorithm name="{checkpoint}">

<classname>com.ibm.wsspi.batch.checkpointalgorithms.{checkpoint}</classname>
    <props>
        <prop name="interval" value="{checkpointInterval}" />
    </props>
```

```

</checkpoint-algorithm>
<results-algorithms>
  <results-algorithm name="jobsum">
    <classname>com.ibm.wsspi.batch.resultsalgorithms.jobsum</classname>
  </results-algorithm>
</results-algorithms>
<substitution-props>
  <prop name="wsbatch.count" value="5" />
  <prop name="checkpoint" value="timebased" />
  <prop name="checkpointInterval" value="15" />
  <prop name="postingsDataStream"
value="\${was.install.root}\${file.separator}temp\${file.separator}postings" />
</substitution-props>
<job-step name="Step1">
  <jndi-name>ejb/DataCreationBean</jndi-name>
  <checkpoint-algorithm-ref name="\${checkpoint}" />
  <results-ref name="jobsum"/>
  <batch-data-streams>
    <bds>
      <logical-name>myoutput</logical-name>
</bds>
</batch-data-streams>
<impl-class>com.ibm.websphere.samples.PostingOutputStream</impl-class>
  <props>
    <prop name="FILENAME" value="\${postingsDataStream}" />
  </props>
</bds>
</batch-data-streams>
  <props>
    <prop name="wsbatch.count" value="\${wsbatch.count}" />
  </props>
</job-step>
<job-step name="Step2">
  <step-scheduling condition="OR">
    <returncode-expression step="Step1" operator="eq" value="0" />
    <returncode-expression step="Step1" operator="eq" value="4" />
  </step-scheduling>
  <jndi-name>ejb/PostingAccountData</jndi-name>
  <checkpoint-algorithm-ref name="\${checkpoint}" />
  <results-ref name="jobsum"/>
  <batch-data-streams>
    <bds>
      <logical-name>myinput</logical-name>
      <impl-class>com.ibm.websphere.samples.PostingStream</impl-class>
      <props>
        <prop name="FILENAME" value="\${postingsDataStream}" />
      </props>
    </bds>
  </batch-data-streams>
</job-step>
  <job-step name="Step3">
    <step-scheduling>
      <returncode-expression step="Step2" operator="eq" value="4" />
    </step-scheduling>
    <jndi-name>ejb/OverdraftAccountPosting</jndi-name>
    <checkpoint-algorithm-ref name="\${checkpoint}" />
  </job-step>

```



```
<results-ref name="jobsum" />
<batch-data-streams>
  <bds>
    <logical-name>dbread</logical-name>

<impl-class>com.ibm.websphere.samples.OverdraftInputStream</impl-class>
  </bds>
</batch-data-streams>
</job-step>
</job>
```

For more information about xJCL, including sample source codes, visit the WebSphere Application Server V8.5 Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/rgrid_samples.html

21.2.2 Compute-intensive programming model

Compute-intensive applications are applications that perform intensive computational work that does not fit into the conventional Java EE request and response paradigm due to the following characteristics:

- ▶ The need for asynchronous submission and start of work
- ▶ The need for work to run for extended periods of time
- ▶ The need for individual units of work to be visible to and manageable by operators and administrators

The compute-intensive programming model provides an environment that addresses these needs, centered around two basic concepts:

- ▶ The use of jobs to submit and manage work asynchronously
- ▶ A minor extension to the asynchronous beans programming model to support work that runs for an extended period

A compute-intensive application is packaged in an enterprise bean module in a Java EE EAR file. The deployment descriptor for the enterprise bean module must contain the definition of the controller bean. The implementation of the controller bean is provided in the application server runtime. The controller bean allows the runtime environment to control jobs for the application. When a job arrives for the application to run, the compute-intensive execution environment invokes the controller bean. The Java Naming and Directory Interface (JNDI) name of this stateless session bean is specified in the xJCL for the job.

A compute-intensive application is started by the application server in the same way as other Java EE applications are started. If the application defines any start-up beans, those beans are run when the application server starts.

You can use Java EE development tools, such as IBM Rational Application Developer, to develop and package compute-intensive applications in the same way that they are used to construct Java EE applications containing enterprise bean modules and asynchronous beans.

For more information about compute-intensive applications, visit section 6.2.2 Compute-intensive programming model of the *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022 IBM Redbooks publication.

21.2.3 Parallel batch

A transactional batch application can be built as a job and divided into subordinate jobs so that the subordinate jobs can run independently and in parallel. You use the parallel job manager to submit and manage the transactional batch jobs.

The parallel job manager (PJM) provides a facility and framework for submitting and managing transactional batch jobs that run as a coordinated collection of independent parallel subordinate jobs. The PJM basic features are:

- ▶ The PJM is in the batch container. You do not need to install and configure the PJM.
- ▶ Only a single xJCL file is required. The file combines the contents of the top-level job xJCL with the contents of the subordinate job xJCLs.
- ▶ You do not need to create a separate database.
- ▶ You package the PJM APIs in the batch application as a utility Java archive (JAR). No shared library is required.
- ▶ The contents of the `xd.spi.properties` file are part of the xJCL.

Figure 21-3 summarizes the PJM architecture, which shows where the APIs are called.

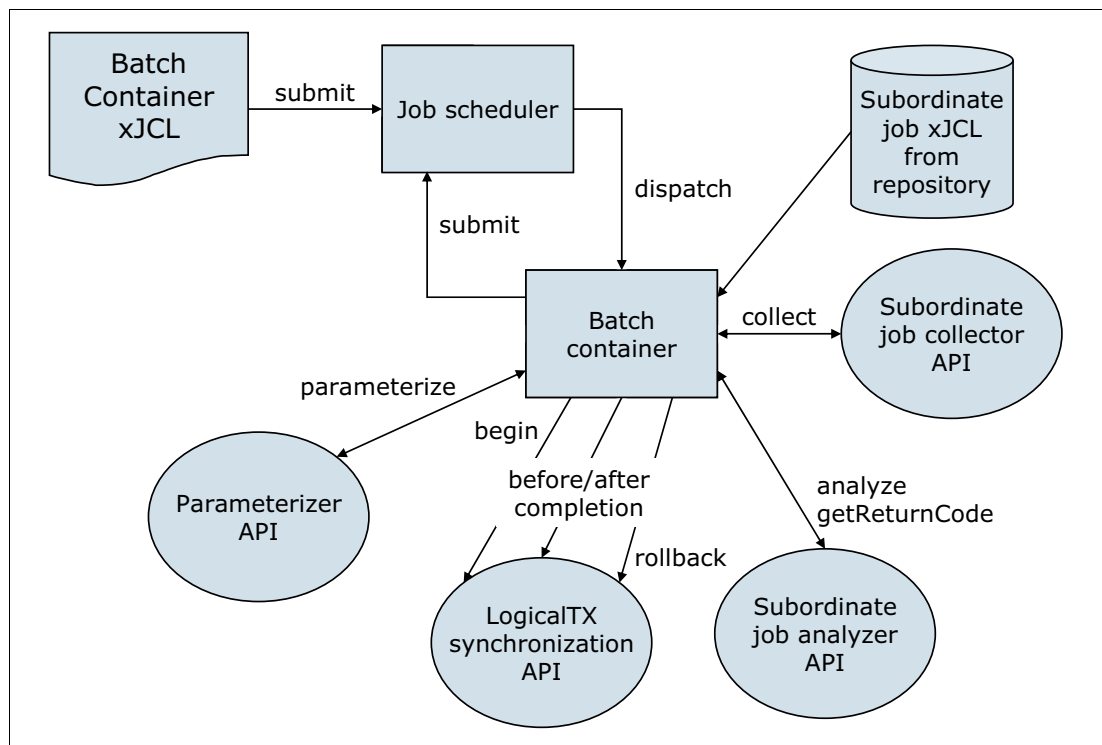


Figure 21-3 PJM architecture

With PJM job management, the top-level job submits the subordinate jobs and monitors their completion. The top-level job end state is influenced by the outcome of the subordinate jobs, as follows:

1. If all subordinate jobs complete in the ended state, that is, in a successful completion, the top-level job completes in the ended state.
2. If any subordinate job completes in the restartable state and no subordinate job ended in the failed state, the top-level job completes in the restartable state.

3. If any subordinate job completes in the failed state, the top-level job completes in the failed state.
4. If the top-level job and subordinate jobs are in the restartable state, restart only the top-level job. If any subordinate jobs are restarted manually, the top-level job does not process the logical transaction properly.

The steps in the execution of parallel batch jobs can be executed in different application server instances that are part of the same cluster. The steps are:

1. First, the xJCL is submitted to the job scheduler, which dispatches the xJCL to an endpoint that runs the application that the xJCL references.
2. The batch container determines that the job is to have subordinate jobs running in parallel from inspecting the run property of the job in the xJCL and then delegates the running to the PJM subcomponent.
3. The PJM invokes the parameterizer API and uses the information in the xJCL to help divide the job into subordinate jobs. The PJM then invokes the LogicalTX synchronization API to indicate the beginning of the logical transaction. The PJM generates the subordinate job xJCL and submits the subordinate jobs to the job scheduler.
4. The job scheduler dispatches the subordinate jobs to the batch container endpoints so that they can run.
5. The batch container runs the subordinate job. When a checkpoint is taken, the subordinate job collector API is invoked.
6. This API collects relevant state information about the subordinate job. This data is sent to the subordinate job analyzer API for interpretation.
7. After all subordinate jobs reach a final state, the beforeCompletion and afterCompletion synchronization APIs are invoked. The analyzer API is also invoked to calculate the return code of the job.

Other aspects to be taken into account to help you understand how to optimally use the parallel job manager are:

► Transaction timeouts

You can keep the default value for the TransactionTimeout property in the top-level job xJCL. You can alternatively adjust it depending on the transaction timeout of the subordinate job.

► Job logs

You can view job logs for a subordinate job from the job management console. The PJM retrieves subordinate job logs for its logical job and aggregates them into its top-level job log.

► Disaster recovery

You can use the job recovery script when a primary site fails to allow a secondary site to take over.

21.2.4 COBOL support

COBOL has been a part of batch processing since the early days of computers and there is significant investment in mission-critical COBOL assets, especially on mainframes. With WebSphere Application Server V8.5, COBOL support includes the following key features:

- In z/OS, you can call standard COBOL modules from Java on the same thread in same process.

- ▶ Java and COBOL run in same transaction scope.
- ▶ WebSphere-managed DB2 connections are shareable with COBOL.
- ▶ You can use COBOL working storage isolation per job step or per remote call.
- ▶ IBM Rational Application Developer tooling is available for Java call stub generation.

The new COBOL container allows COBOL modules to be loaded into the WebSphere Application Server for z/OS address space and invoked directly. It provides the means of direct integration of COBOL resources into WebSphere Java processing. The container itself is implemented as a handful of DLLs and JAR files, as shown in Figure 21-4.

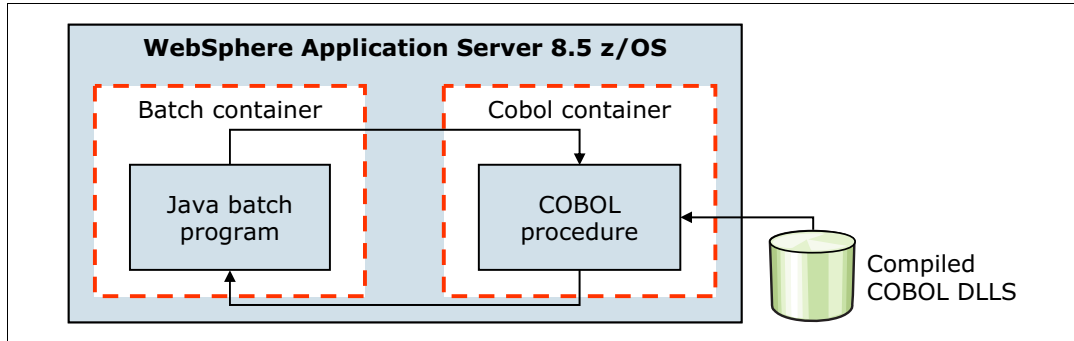


Figure 21-4 WebSphere Batch COBOL container

The COBOL container enables COBOL modules to be loaded into the batch container where they are invoked directly by the batch application. The COBOL container itself can be created and destroyed multiple times within the lifecycle of a server. Each container is created with Language Environment enclave separate from that of a server. The container is assured of a clean Language Environment each time it is created.

Java programs can pass parameters into COBOL and retrieve the results. The COBOL call stub generator tool is provided to create the Java call stubs and data bindings based on the data and linkage definitions in the COBOL source.

You can dynamically update a COBOL module without having to restart the application server. Further, JDBC Type 2 connections created by the Java program can be shared with the COBOL program under the same transactional context. The COBOL container supports a wide variety of data types beyond integers, including primitive and national data types.

For more information about COBOL support on WebSphere Application Server V8.5 Batch, visit the following information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/a/tgrid_cobol_container_developing.html

More information is also in section 6.5.3 *COBOL support in WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022.

21.2.5 Batch toolkit

IBM provides two features to help with the development of batch applications:

- ▶ Rational Application Developer includes a complete set of tools to facilitate batch application development
- ▶ The batch toolkit supplied with WebSphere Application Server includes tools to facilitate batch application development. It combines batch development tools into a ready-to-use

environment and includes simple command-line utilities that deal with packaging applications and other tasks. However, it does not include the full utilities that come with Rational Application Developer.

The batch toolkit contains the following components:

- Batch framework
- Lightweight batch container
- Packaging tool
- xJCL generator
- Unit test server

21.3 Configuring the batch environment

Configuration tasks for the batch environment include configuring the job scheduler and grid endpoints.

To set up an environment to host transactional batch or compute-intensive job types, you must deploy the job scheduler and the batch container to at least one WebSphere application server or cluster. The transactional batch, compute-intensive applications, or both are installed on the same WebSphere application server or cluster.

The job scheduler and batch container both require access to a relational database. Access to the relational database is through the underlying WebSphere Application Server connection management facilities. The relational databases supported are the same as those relational databases that are supported by WebSphere Application Server, including DB2, Oracle, and others. The simple file-based Apache Derby database is automatically configured for you by default so that you can quickly get a functioning environment up and running. However, do not use the Derby database for production use. Moreover, the default Derby database does not support a clustered job scheduler, nor a clustered batch container.

A highly-available environment includes both a clustered job scheduler and one or more clustered batch containers. Clustering requires a network database. Use production grade databases, such as DB2 for this purpose.

21.3.1 Configuring the job scheduler

The job scheduler accepts job submissions and determines where to run them. As part of managing jobs, the job scheduler stores job information in an external job database. Configuration for the job scheduler includes the selection of the deployment target, data source JNDI name, database schema name, and endpoint job log location to be configured for the scheduler.

You can use the command-line interface, the Enterprise JavaBeans (EJB) interface, the web services interface, and the job management console to communicate with the job scheduler.

Stand-alone application servers or clusters can host the job scheduler. The first time a server or cluster is selected to host the grid scheduler, an embedded Apache Derby database is automatically created and configured to serve as the scheduler database if the default data source JNDI name `jdbc/1rsched` is selected.

Although Derby is used as the default job scheduler database, you might want to use your own database. See the following information center website for more details about creating a job scheduler and grid endpoint database:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/tgrid_bgsched.html

21.3.2 Securing the job scheduler

WebSphere authentication determines the users, from the active WebSphere security registry, that can authenticate and gain access to the web, command line, and programmatic interfaces of the job scheduler. Therefore, you can secure the job scheduler application by simply enabling global security and application security.

Application security secures the job management console. The job scheduler application uses a combination of both declarative and instance-based security approaches to secure jobs and commands.

Finally, security for the batch environment is based on two basic principles of WebSphere security:

- ▶ Authentication

Allows users defined to the WebSphere active security repository to authenticate and gain access to the web, command line, and programmatic interfaces of the job scheduler.

- ▶ Authorization

Using roles assignment, WebSphere determines if the authenticated users have the proper security rights to perform actions against jobs. There are three roles:

- **Irsubmitter:** Users in the Irsubmitter role can view, submit, and operate on their own jobs, but on no others.
- **Iradmin:** Users assigned the Iradmin role have authority to perform all job scheduler application actions on all jobs regardless of job ownership.
- **Irmonitor:** Users assigned the Irmonitor role only can view jobs and job logs of all users.

For more details about the authoritative roles and the capabilities that each can perform, refer to the following WebSphere Application Server V8.5 Information Center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/ae/cgrid_bgroles.html

The actions that a user can take against a job depend on the security model that is being enforced. User actions against a job can be role based, group based, or a combination of the two.

- ▶ Role based security

When role-based security is enabled, you must be granted the Irsubmitter role, the Iradmin role, or the Irmonitor role to act on a job.

- ▶ Job group security

In the group security model, group-affiliation alone is the basis for all job-related security decisions. The administrator does not assign job roles to specific users. A user can complete an action for a job only if the user and job are members of the same group. For example, if two users are members of the same group and each submits a job that is assigned to that same group, both users can view and take actions against either of the two jobs.

- ▶ Job group and role security

In the group and role security model, both group-affiliation and role-based security governs job-related security decisions. This means a user can take a job-related action only if the user and job are members of the same group and the user's role permits the job action.

21.3.3 Job scheduler integration with external schedulers

Many customers already use an external workload scheduler to manage batch workloads on the z/OS operating system. While a Java batch running inside a WebSphere Application Server environment is attractive, a way to control batch jobs through an external workload scheduler is important.

You can integrate the job scheduler with an external workload scheduler by configuring and securing the job scheduler, enabling the interface, and running batch jobs with the WSGRID utility.

External scheduler integration

Because an external scheduler does not know how to directly manage batch jobs, a proxy model is used. The proxy model uses a regular JCL job to submit and monitor the batch job. The JCL job step invokes a special program provided by batch, named WSGRID. The WSGRID application submits and monitors a specified batch job, writing intermediary results of the job into the JCL job log. WSGRID does not return until the underlying job is complete, consequently providing a synchronous execution model. Because the external scheduler can manage JCL jobs, it can manage a JCL job that invokes WSGRID. Using this pattern, the external scheduler can indirectly manage a job.

An optional plug-in interface in the job scheduler enables a user to add code that updates the external scheduler operation plan to reflect the unique state of the underlying job, such as job started, step started, step ended, job ended. The WSGRID program is written with special recovery processing so that if the JCL job is canceled, the underlying job is canceled also, thus ensuring synchronized lifecycle of the two jobs.

Figure 21-5 on page 768 shows the job control by an external workload scheduler but without a z/OS Job Entry Subsystem (JES) being required.

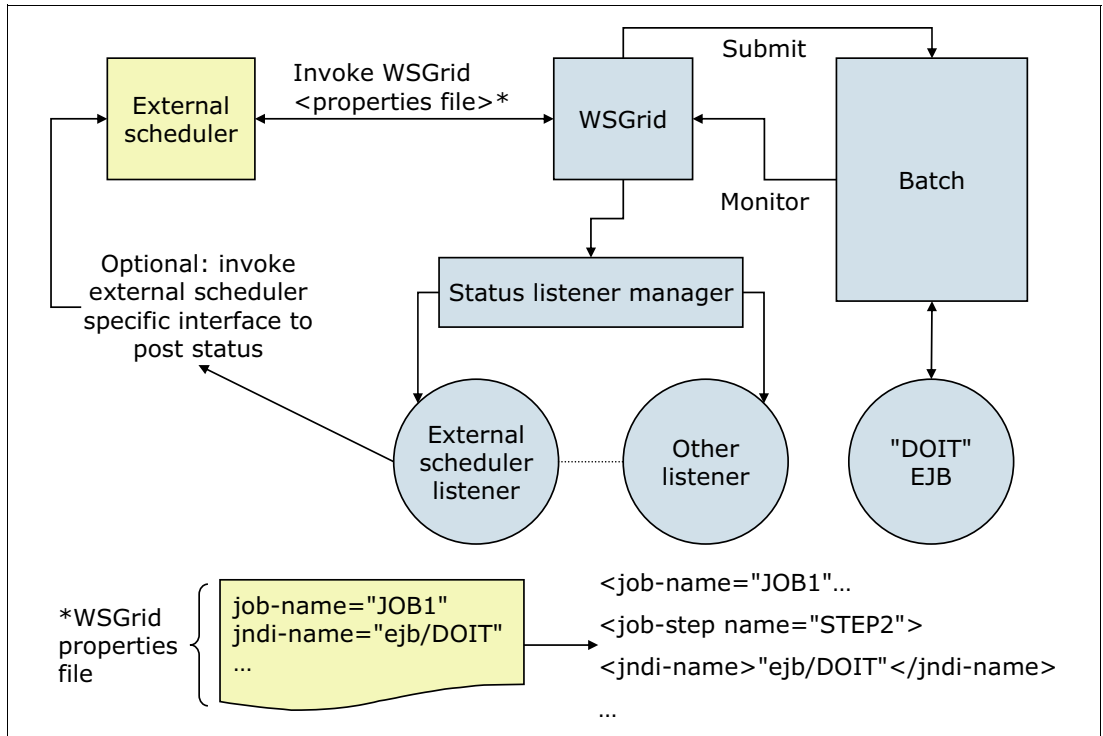


Figure 21-5 Job control by an external workload scheduler

Figure 21-6 on page 769 shows the job control by an external workload scheduler for the z/OS platform environment. In this diagram, the Tivoli Workload Scheduler is shown as an example workload scheduler, communicating with the z/OS Job Entry Subsystem (JES).

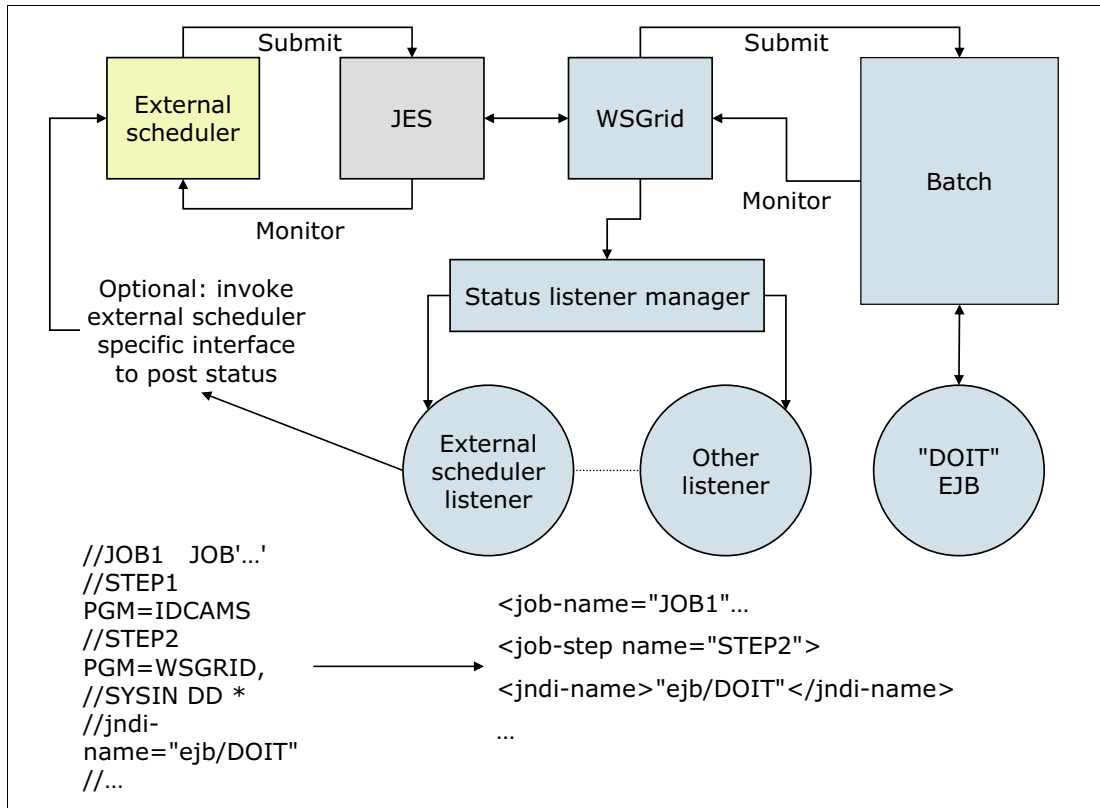


Figure 21-6 Job control by an external workload scheduler on z/OS

Configuring the external scheduler interface

You can configure an external scheduler interface to control the workload for batch jobs. To communicate with the external scheduler interface, you can use:

- ▶ The default messaging provider as a Java Message Service (JMS) provider.

The external scheduler interface uses Java Message Service (JMS) as its default messaging provider. JMS is a bidirectional communication mechanism between an external client and the job scheduler.

For more information about how to set up the external scheduler interface using the default messaging provider, refer to the following WebSphere Application Server V8.5 Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgas.doc/ae/tgrid_cgexternalconfig.html

- ▶ On z/OS you also have the option of setting up the external scheduler interface by using WebSphere MQ as a messaging provider.

For information about how to set up the external scheduler interface using the WebSphere MQ for z/OS platforms, refer to the following WebSphere Application Server V8.5 Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgas.doc/ae/tgrid_zoscgnative.html

21.3.4 Configuring grid endpoints

To set up a WebSphere grid endpoint:

1. Install a batch application on a server or cluster using the administrative console, wsadmin commands, or another supported method for deploying applications.
2. If the application is the first batch application installed on the server or cluster, restart the server or cluster.

The WebSphere grid endpoints are automatically set up. By installing the application on the deployment target, the common batch container is automatically deployed on the server or cluster selected using the default Apache Derby data source.

Note: The default file-based Derby data source can be used only when using the batch function on a stand-alone application server. If you have a WebSphere Application Server Network Deployment environment, you must use a network database.

Check the following WebSphere Application Server V8.5 Information Center website for more information about grid endpoints variables:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/a_e/rgrid_endpointvw.html

21.3.5 Configuring the job scheduler and job management console

The job management console is a stand-alone web interface for managing jobs. It runs on a target application server or cluster within a WebSphere cell, more specifically, in the cluster or application server where the job scheduler is enabled. Section 21.4.1, “Enabling the job scheduler” on page 774 shows how to configure the job scheduler and therefore enable the job management console. Section 21.4.5, “Using the job management console” on page 780 shows usage examples of the job management console.

With the job management console you can:

- ▶ Submit jobs
- ▶ Monitor job execution
- ▶ Perform operational actions against jobs
- ▶ View job logs
- ▶ Manage the job repository
- ▶ Manage job schedules

Some of the specific actions that you can execute through the job management console are:

- ▶ Submitting job schedules with a preferred processing time
- ▶ Configuring job schedules so that they can, for example, occur or recur at a specific time of day or week
- ▶ Choosing to delay the submission of a job by specifying the start date and time of when you want to run the job

When role-based security is enabled, you must be granted the Jrsubmitter role, the Jradmin role, or the Jrmonitor role through the administrative console to access the job management console.

When the security enabled is based on the group and the role, you must be in the appropriate group and the appropriate role to access the job management console. You must be in the

user group of the job or the administrative group. You must also be in the Irsubmitter role, the Iradmin role, or the Irmonitor role.

Figure 21-7 shows the job management console.

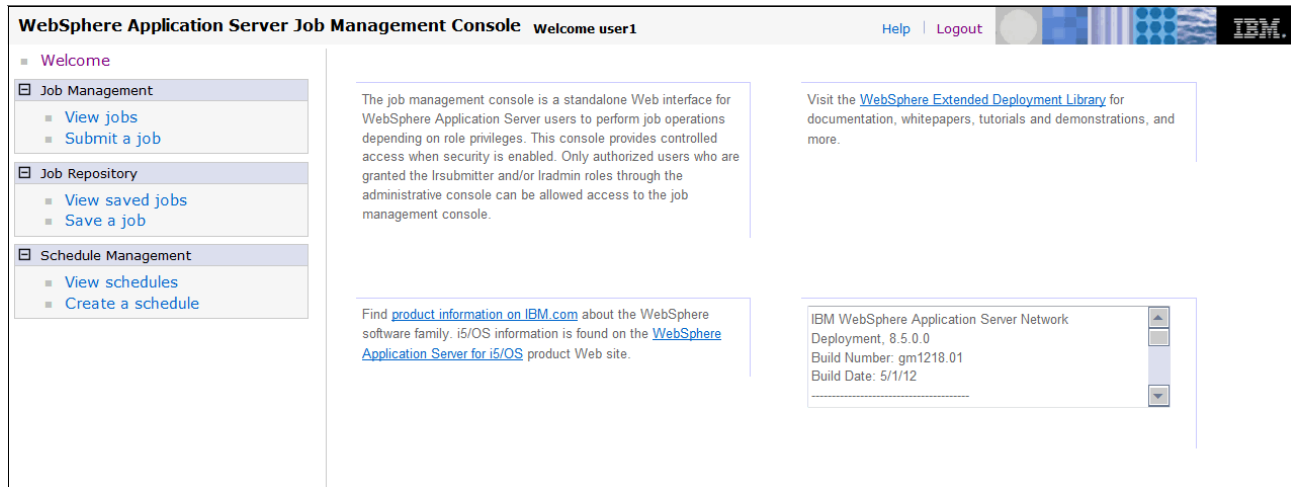


Figure 21-7 Job management console

To access the job scheduler from the job management console:

1. Configure the job scheduler.
2. Ensure that the job scheduler is running.

If the application server or cluster members on which the job scheduler is installed have the started icon in the status field, the job scheduler is usually running. You can verify whether the job scheduler started by checking the log files.

3. In a browser, type the web address: `http://<job scheduler server host>:<port>/jmc`.

If an on-demand router (ODR) is defined in the cell, type the web address:

`http://<odr host>:80/jmc`.

4. If you cannot access the job management console, check the appropriate log. If you specified a server in the web address, check the server log. If you specified a cluster member in the web address, check the cluster member log.

21.3.6 Command-line interface for batch jobs

The command-line interface interacts with the job scheduler to submit and manipulate a batch job. It is located in the `app_server_root/bin` directory as the `lrcmd.sh` or `lrcmd.bat` script and can be started from any location in the WebSphere cell.

The following examples illustrate the use of the `lrcmd` script on a UNIX system:

- ▶ Submit a job to the job scheduler.

```
./lrcmd.sh -cmd=submit -xJCL=myxjclfile.xml -host=myHost -port=80
```

- ▶ Cancel a previously submitted job.

```
./lrcmd.sh -cmd=cancel -jobid=myJob:2 -host=myHost -port=80
```

- ▶ Restart a job.

```
./lrcmd.sh -cmd=restart -jobid=myJob:2 -host=myHost -port=80
```

- ▶ Purge job information.
./lrcmd.sh -cmd=purge -jobid=myJob:2
- ▶ Show the status of a batch job.
./lrcmd.sh -cmd=status host=myHost -port=80
- ▶ Save an xJCL to the job repository.
./lrcmd.sh -cmd=save -xJCL=myxjclfile.xml -job=myJob -host=myHost -port=80
- ▶ Remove a job from the job repository.
./lrcmd.sh -cmd=remove -jobid=myJob:2 -host=myHost -port=80
- ▶ Show the status of a batch job.
./lrcmd.sh -cmd=status host=myHost -port=80
- ▶ Suspend a job.
./lrcmd.sh -cmd=suspend -jobid=myJob:2 -seconds=300 -host=myHost -port=80
- ▶ Resume start of a previously suspended job.
./lrcmd.sh -cmd=resume -jobid=myJob:2
- ▶ Display the output for a job.
./lrcmd.sh -cmd=output -jobid=myJob:2 -host=myHost -port=80
- ▶ Display the return code of a batch job.
./lrcmd.sh -cmd=getBatchJobRC -jobid=myJob:2 -host=myHost -port=80
- ▶ Submit a recurring job request to the job scheduler.
./lrcmd.sh -cmd=submitRecurringRequest -job=WeeklyJob -request=MyWeeklyReport -interval=weekly -startDate=2012-07-01 -startTime=23:00:00
- ▶ Modify an existing recurring job request.
./lrcmd.sh -cmd=modifyRecurringRequest -request=MyWeeklyReport -startDate=2012-07-01 -startTime=22:30:00 -xJCL=/tmp/myJulXJCL -port=80\

Check the following WebSphere Application Server V8.5 Information Center website for more information about the command line interface for batch jobs:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgwas.doc/a/cgrid_bgcommnd.html

21.3.7 Job logs

A job log is a file that contains a detailed record of the execution details of a job. System messages from the batch container and output from the job executables are collected. By examining job logs, you can see the lifecycle of a batch job, including output from the batch applications themselves.

A job log is composed of the following three types of information:

- ▶ xJCL
A job log contains a copy of the xJCL used to run the job, including xJCL substitution values.
- ▶ System messages

A set of system messages that communicate the major lifecycle events corresponding to the job. The following system events are recorded in a job log:

- Begin and end of a job
- Begin and end of a step
- Begin and end of a checkpoint
- Open, close, and checkpoint of a batch data stream
- Checkpoint algorithm invocation / results
- Results algorithm invocation / results

► Application messages

A set of messages written to standard out and standard error by a job step program.

Job log output is collected on the job scheduler node and on the grid execution endpoint node. The logs are viewable through the job management console. Information for the logs is dynamically updated, so you can refresh the job log view to collect the latest information of the job log while the job is running. Example 21-2 is a sample of this type of log.

Example 21-2 Job log output

```
System.out: [03/13/07 08:25:32:708 EDT] Tue Mar 13 08:25:32 EDT 2007: SimpleCI
application starting...
System.out: [03/13/07 08:25:32:708 EDT] -->Will loop processing a variety of math
functions for approximately 30.0 seconds!
System.out: [03/13/07 08:26:02:752 EDT] Tue Mar 13 08:26:02 EDT 2007: SimpleCI
application complete!
System.out: [03/13/07 08:26:02:753 EDT] -->Actual Processing time = 30.043
seconds!
CWLRB5764I: [03/13/07 08:26:03:069 EDT] Job SimpleCIEar:44 ended
```

21.3.8 Job classes

Each job is assigned to a job class, which defines policies to limit resource consumption by batch jobs. Job classes can be configured using the administrative console.

Job classes establish policies for:

► Execution time

Maximum time a job can run before being automatically canceled by the system.

► Concurrent jobs

Maximum number of concurrently dispatched jobs of a given job class.

► Job log retention

Specifies the rule for deleting aged job logs. Retention can be specified by either space or time:

– Space

Specified in megabytes. Job logs of the specified class are deleted from oldest to newest on an endpoint if the sum of space used by job logs exceeds the specified maximum.

– Time

Specified as an integral number of days. Job logs of the specified class older than N days old are automatically deleted by the system.

- ▶ Job output queue

Specifies the rule for deleting jobs on the job output queue. A job is on the output queue after it has either completed, stopped, or canceled. Output queue policy allows for automatic purging of the output queue by:

- Number

Specified as an integral number of jobs. When jobs on the output queue of the specified class exceed this number, they are deleted oldest to newest until the total is less than the specified number.

- Time

Specified as an integral number of days. Job logs of the specified class older than N days old are automatically deleted by the system.

21.4 Example: Working with batch applications

This section shows, by example, how to work with batch applications. Because this is a simple example to illustrate batch concepts, the test environment is a stand-alone server and the default Derby database. For a production environment, it is more common to use application server clusters and a database, such as DB2.

The conditions for this scenario are:

- ▶ A stand-alone application server called *itsoBatch* that hosts the job scheduler and batch application.
- ▶ The WebSphere Application Server install path (<WAS_INSTALL_ROOT>) is */opt/IBM/WebSphere/AppServer*.
- ▶ The WebSphere profile for the server (<PROFILE_NAME>) is *AppSrv01*.
- ▶ The *Java Batch IVT Sample* is used. The sample is stored in */tmp/sample_ivt* (referred to as <unzipped_sample_dir>). To download the sample batch application, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.samples.doc/ae/sample_mb_ivt.html

21.4.1 Enabling the job scheduler

After the *itsoBatch* application server is created, the job scheduler can be configured using the administrative console, as shown in Figure 21-8 on page 775. The following instructions provide an example of how to configure the job scheduler using the administrative console:

1. Log on to the administrative console.
2. Click **System administration** → **Job scheduler** to view the Job scheduler page.

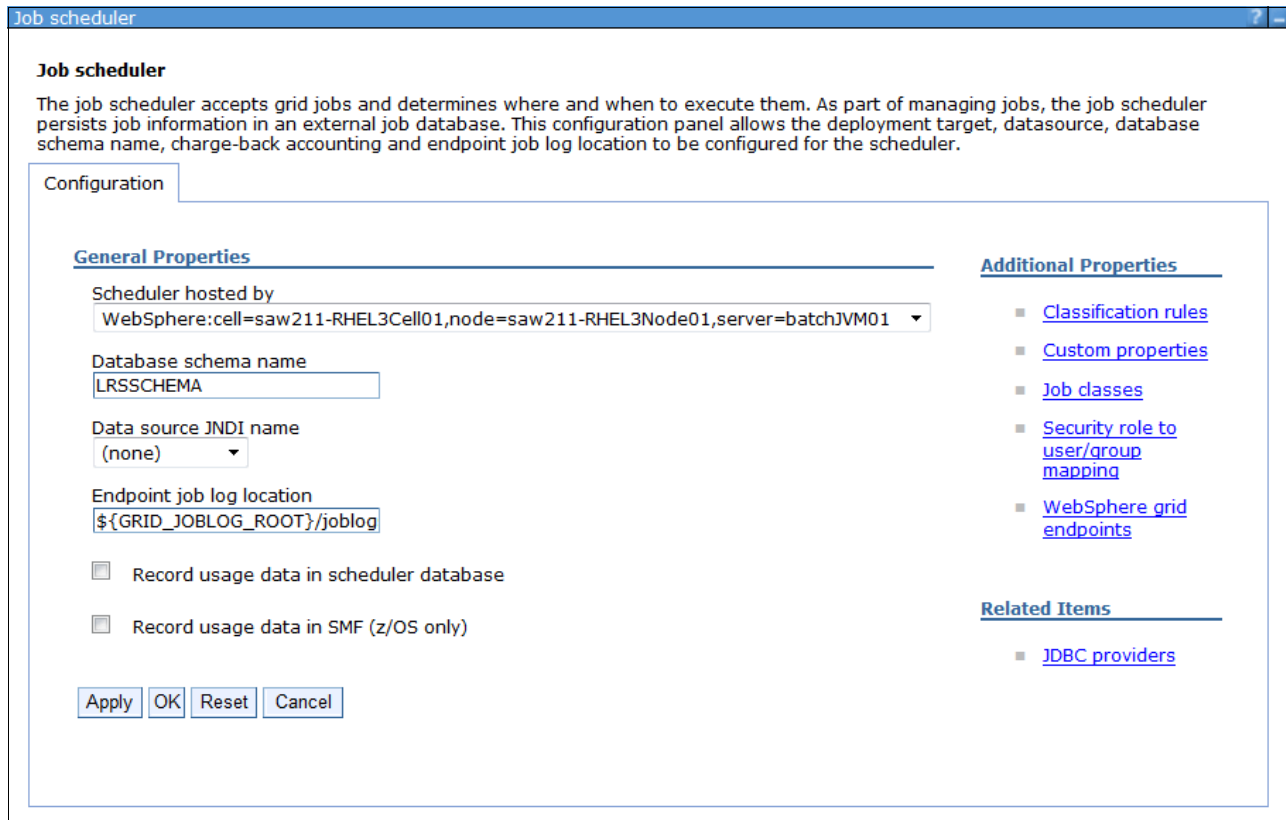


Figure 21-8 Job scheduler configuration

- a. In the **Scheduler hosted by** list, select the deployment target.
 - b. Type the database schema name. The default is LRSSCHEMA.
 - c. Select the data source JNDI name from the list. If the default of (none) is selected, a default embedded Derby job scheduler database is created with a value of jdbc/Irsched.
 - d. Type the directory where the job scheduler and the batch execution environment write the job logs. The default is \${GRID_JOBLOG_ROOT}/joblogs.
 - e. Optional: Select the **record usage data in scheduler database** option to specify if the scheduler records job usage data for charge-back purposes in the scheduler database.
 - f. Click **OK** and save the configuration.
3. If administrative security is enabled, enable application security and secure the job scheduler, described in 21.4.4, “Securing the job scheduler using Job groups” on page 777.

21.4.2 Verifying the job scheduler installation

To verify that the job scheduler is installed correctly:

1. Restart the application server (or cluster members) where the job scheduler is configured.

If the application server (or cluster members) on which the job scheduler is installed has the started icon in the status field, the job scheduler is active. You can verify whether the job scheduler started by checking the log files.

2. Access the job management console through a web browser by typing:

```
http://job_scheduler_server_host:grid_host/jmc
```

The *grid_host* port is the WC_defaulthost port for the server running the job scheduler. To find this port, go to your server in the administrative console, expand ports, and look for WC_defaulthost. In the case of our test environment, the URL is

```
http://saw211-RHEL3:9080/jmc.
```

To ensure that the job management console is working correctly, check the SystemOut.log file on the target application server configured to host job scheduler. Example 21-3 shows the message in the log that the application as started.

Example 21-3 SystemOut.log message

```
[6/20/12 12:04:48:739 EDT] 0000006c JobSchedulerS I    CWLRB3220I: Long Running  
Job Scheduler is initialized
```

21.4.3 Installing the sample batch application

Use the following steps to install the IVT sample batch application into WebSphere Application Server:

1. Download the sample IVT application compressed file from the WebSphere Application Server V8.5 Information Center website:

```
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.samples.doc/ae/sample\_mb\_ivt.html
```

2. Unzip the file on your target server.
3. Configure the *JAVA_HOME* and add it to the *PATH* environment variables in your system to be able to run java and create the IVT database on Derby.
4. Run the command **java -version** to make sure that Java Version 1.6 or later is installed and defined in the system's path.
5. Create the IVTDB database in the server where the target application server is running.

Use the appropriate CreateIVTTables DDL file (CreateIVTTablesDerby.ddl) that is located in the *<unzipped_sample_dir>/IVT/scripts* directory of the uncompressed IVT sample application: From a command prompt, issue the following commands:

```
cd <WAS_INSTALL_ROOT>/derby/databases
```

```
java -Djava.ext.dirs=<WAS_INSTALL_ROOT>/derby/lib -Dij.protocol=jdbc:derby:  
org.apache.derby.tools.ij<unzipped_sample_dir>/scripts/CreateIVTTablesDerby.ddl
```

6. Create the JDBC resources:
 - a. In the administrative console, click **Resources** → **JDBC** → **JDBC providers**.
 - b. Create a JDBC XA provider at the server scope with the following properties:
 - Database type: Derby
 - Provider type: Derby JDBC Provider
 - Implementation type: XA data source
 - Name: XDCGIVT Derby JDBC Provider (XA)
 - Description: Accept the default value
 - c. Click through the remaining panels. On the last panel, click **Finish**.
 - d. In the administrative console, click **Resources** → **JDBC** → **Data source**.

- e. Create a data source with the following properties:
 - Data source name: XDCGIVT data source (XA)
 - Java Naming and Directory Interface (JNDI) name: jdbc/IVTdbxa
 - JDBC provider: XDCGIVT Derby JDBC Provider (XA)
 - Database name: <WAS_INSTALL_ROOT>/derby/databases/IVTDB
 - Select the option **Use this data source in container managed persistence (CMP)**.
 - Security aliases: Accept the default values.
 - f. Click through the remaining panels. On the last panel, click **Finish**.
 - g. Save the configuration to the master repository and synchronize the nodes.
7. Select the new data source, and click **Test Connection** to test the connection to the database.
 8. Install the XDCGIVT sample using the administrative console:
 - a. In the administrative console, click **Applications** → **New application** → **New Enterprise Application**.
 - b. Specify the full path to the sample XDCGIVT.ear file(<unzipped_sample_dir>/installableApps/XDCGIVT.ear).
 - c. In the wizard, select **Fast Path - Prompt only when additional information is required**, accept default settings, apply the proper modules mapping, and continue through the steps.

When mapping modules of the batch application to servers, select the server (or cluster) to run the batch job (*itsoBatch*).

Click **Finish** when you are done.
 - d. Restart the application server.
 9. After the application server is restarted, verify that the application installed successfully:
 - a. Go to the Enterprise applications administrative console page by clicking **Applications** → **Application Types** → **WebSphere enterprise applications**.
 - b. If the application is not running, select the application, and click **Start**.

Note: For more information about how to install batch applications using the administrative console, wsadmin scripting, or using another supported way to deploy JAVA EE applications, see Chapter 23, “Packaging and deploying Java EE applications” on page 813.

21.4.4 Securing the job scheduler using Job groups

You can secure the job scheduler using groups. A user can then act on a job only if the user and job are members of the same group.

This example assumes that the job scheduler is configured and that WebSphere security is enabled. It also assumes that a group was created and a user that belongs to the group. For this example, the user ID is *user1* and the group is *BATCHGROUP*.

Group security is enabled for the job scheduler by mapping authenticated users to the Iradm admin administrative security role. The next step is to assign a group to a job.

Refer to the section on assigning users and groups to roles in the WebSphere Application Server V8.5 Information Center documentation, and follow the directions:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/tsec_tasroles.html

You can use the WebSphere Application Server administrative console to enable job group security for the job scheduler with the following procedure:

1. Enable job group security for the job scheduler:

For the purpose of this scenario, we mapped the wasadmin, our primary WebSphere administrative ID to the lradmin role, and a user group, BATCHGROUP, to the lsubmitter role. This allows you to access the job scheduler console with different roles and understand the difference of permissions and possibilities each role provides. There is also the lmonitor role that can be used for ID and group mapping:

- a. Click **System administration** → **Job scheduler** → **Security role to user/group mapping**.
- b. Select **lradmin** for the role, and click **Map Users**:
 - i. In the **Job scheduler** → **Security role to user/group mapping** → **Map users/groups** window, click **Search** to list all users.
 - ii. Select your primary WebSphere administrative ID from the Available listview, which in this example is wasadmin, and click the **right arrow** button to add the selected user to the **Selected** list, as shown in Figure 21-9 on page 779.

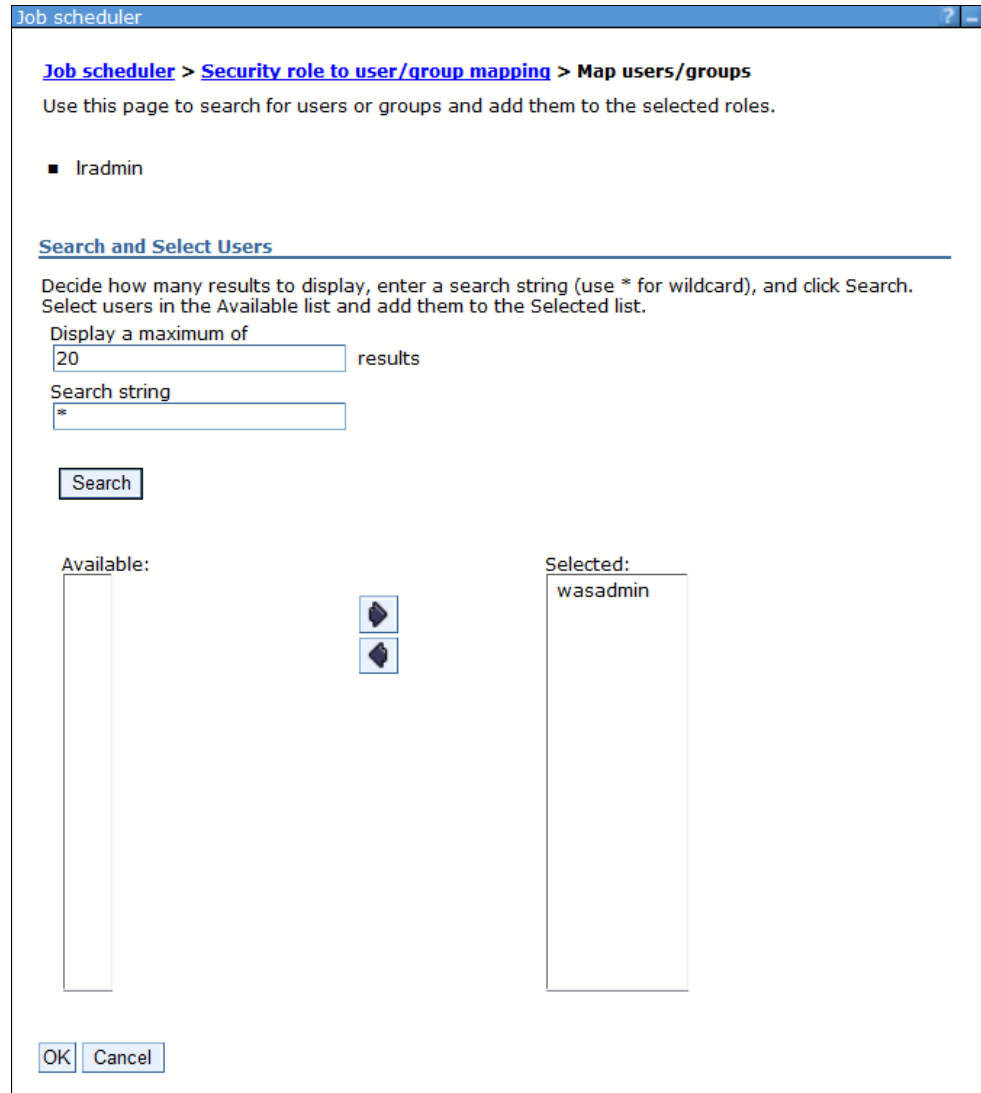


Figure 21-9 Mapping users to the lradmin role

- iii. Click **OK**.
- c. Select the **lsubmitter** role, and click **Map Groups**.
 - i. In the **Job scheduler** → **Security role to user/group mapping** → **Map users/groups** window, click **Search** to list all groups.
 - i. Select the **BATCHGROUP** from the Available listview, and click the **right arrow** button to add the selected user to the Selected listview.
 - ii. Click **OK**.

Figure 21-10 on page 780 shows the finalized security roles mapping for this scenario.

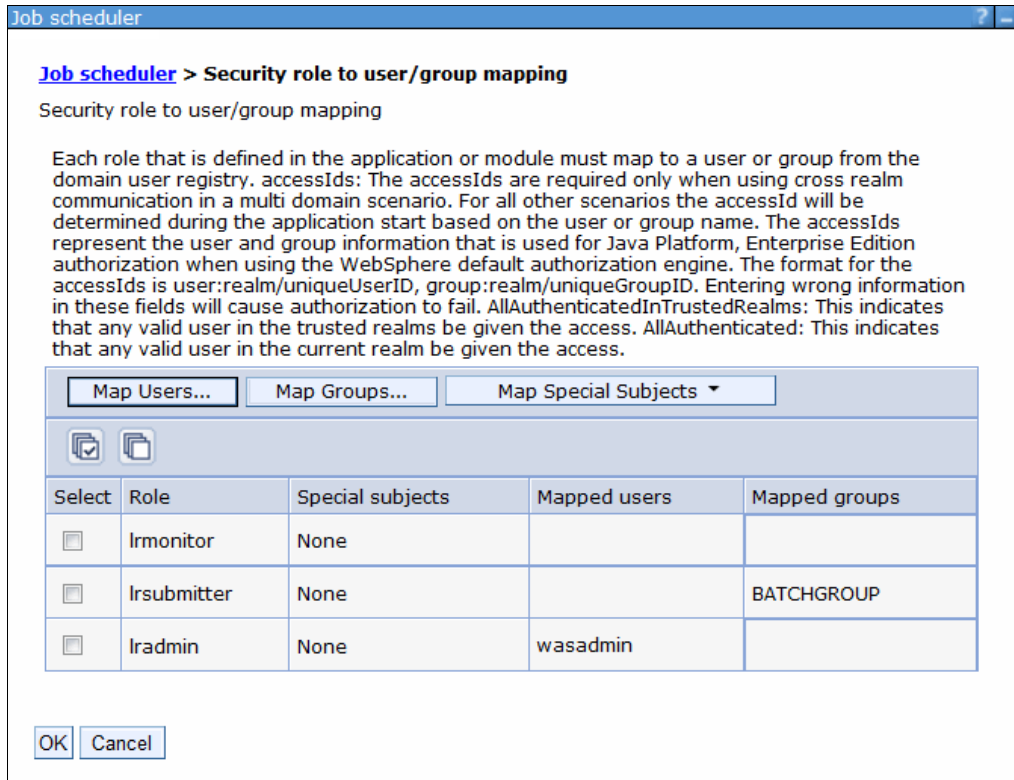


Figure 21-10 Security role to user/group mapping

- d. Save the updates.
- e. Restart the server.
- f. Verify that group security is enabled. A message in the SystemOut.log file (shown in Example 21-4) of your application server (in our case the batchJVM01 application server) indicates that group security is enabled.

Example 21-4 SystemOut.og message

CWLRB5837I: The WebSphere Application Server Batch Feature is running under GROUP security policy

For information about how to configure the job scheduler security based on groups and roles and groups techniques, refer to the following WebSphere Application Server V8.5 Information Center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.cgas.doc/a/e/tgrid_bgsecure_top.html

21.4.5 Using the job management console

After the job scheduler is enabled with the proper security settings and you completed the deployment of the sample IVT batch application, you can use the job management console to perform administrative tasks for the batch job.

The following URL is used to access the job management console:

<http://saw211-RHEL3:9080/jmc>

For comparison, access the job management console with the user ID that was mapped to the *lradmin* role, which in our case is the *wasadmin* ID (Figure 21-11). Note the full range of functionality that is available.

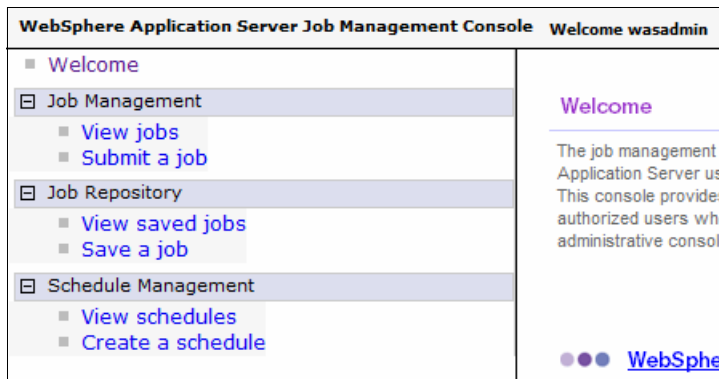


Figure 21-11 Job management console view for a user with *lradmin* role

Next, access the console with the user ID mapped to the *lrsubmitter* role, which in this test case is *user1* (Figure 21-12). Note the noticeable difference between the *lradmin* and the *lrsubmitter* permissions. The *user1* ID has restricted access in the console.

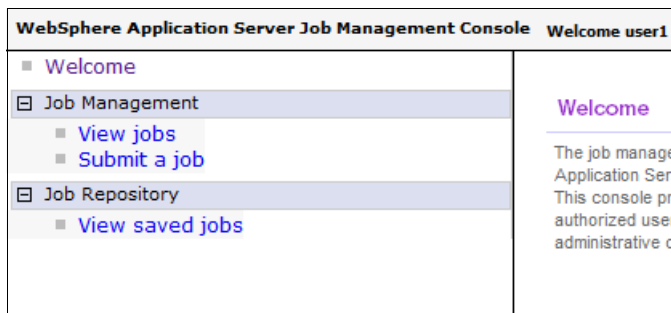


Figure 21-12 Job management console view for a user with *lrsubmitter* role

Submitting a job

The sample IVT application contains a few batch job xJCL files:

- ▶ XDCGIVTtxt2txtJCL.xml
Copies one text file to another text file and compares the two files.
- ▶ XDCGIVTbyte2byteJCL.xml
Copies a byte file to another byte file and compares the two files.
- ▶ XDCGIVTtxt2db2txtJCL.xml
Copies a text file to a database, copies the entries in the database to another text file, and compares the two files.

To test these batch jobs, you must first edit these files and set the following parameters to a valid location on your test server:

- ▶ `inputDataStream`, with a valid input file path, as shown in the following example:

```
<prop name="inputDataStream" value="/tmp/ivtJobs/input-text.txt" />
```
- ▶ `outputDataStream`, with a valid output file path, as shown in the following example:

```
<prop name="outputDataStream" value="/tmp/ivtJobs/output-text.txt" />
```

After you modify the files, follow this procedure to submit the XDCGIVTtxt2txtJCL.xml job:

1. Connect to the Job management console as user1 (Irsubmitter role).
2. Navigate to **Job Management** → **Submit a job**.
3. Under the Job Definition section, apply the following options:
 - a. Select **Local file system**.
 - b. Click **Browse** next to Specify path to xJCL to specify the file containing the job definition to submit as a new job:

```
<unzipped_sample_dir>/IVT/scripts/XDCGIVTtxt2txtJCL.xml
```
 - c. Select **Update substitution properties** to update the values of the substitution properties for the job. If a job has substitution properties without values, you must specify them. For the testing of this scenario, this option is disabled.
 - d. Enable **Delay submission** to delay the start date and time of when to run the job, as shown in Figure 21-13.

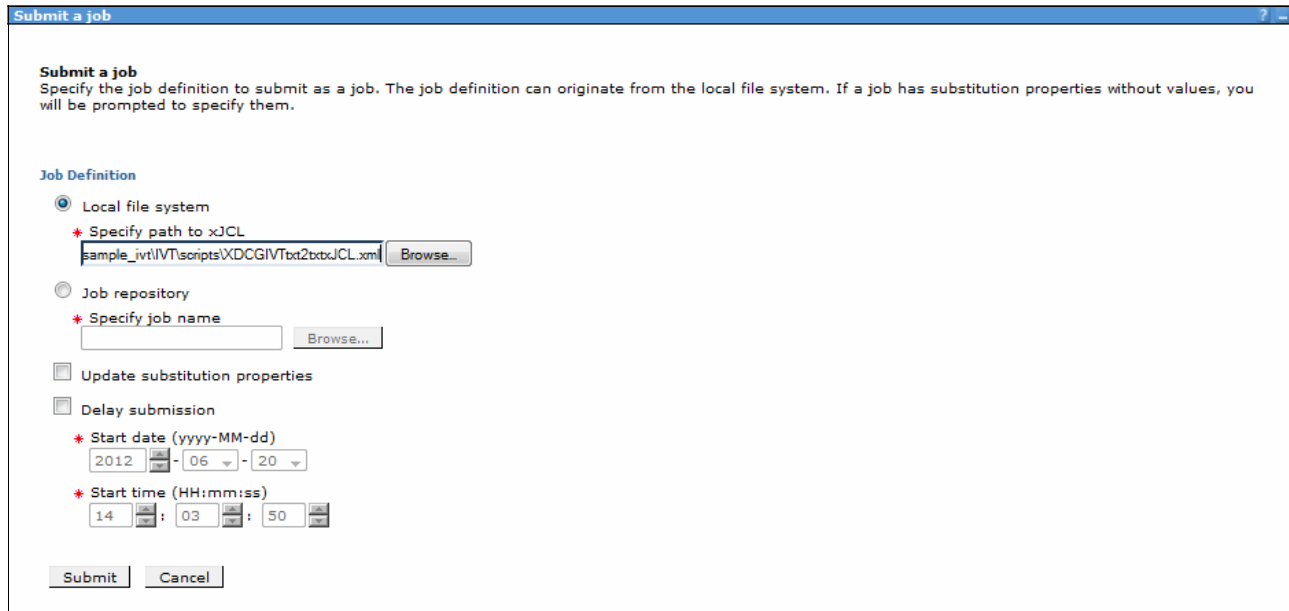


Figure 21-13 Submitting a job from the Job management console

4. Click **Submit**

Saving a job

Saved batch jobs can be stored in a job repository and can be re-used when you are submitting jobs or creating job schedules. To save a batch job using the job management console:

1. Navigate to **Job Repository** → **Save a job**.
2. Specify a Job name.
3. Specify the xJCL path of the batch job.
4. Click **Save**.

Creating a job schedule

The job management console allows you to create job schedules so that job processing can occur or recur on a specific time of day or week.

To create schedules:

1. Navigate to **Schedule Management** → **Create a schedule** (Figure 21-14).

Specify information for creating a schedule.

Step 1: Create schedule

Step 2: Specify job

Step 3: Confirm create schedule

Create schedule

Specify the name of the schedule to create. Specify the start date and time for the job to first run.

* Name: weeklySchedule

* Start date (yyyy-MM-dd): 2012-07-01

* Start time (HH:mm:ss): 22:00:00

* Interval: Weekly

< Back Next > Finish Cancel

Figure 21-14 Creating a job scheduler

Perform the following actions:

- a. Specify the schedule name.
 - b. Set the start date of the new schedule.
 - c. Set the repetition interval for the schedule and then click **Next**.
2. In the Specify job window, specify the local path of the xJCL file, or select a previously saved job from the job repository and then click **Next**.
 3. You can specify values for substitution properties for your job. In the case of the IVT sample application, Figure 21-15 shows the available parameters.

Specify information for creating a schedule.

Step 1: Create schedule

Step 2: Specify job

Step 2.1: Specify substitution properties

Step 3: Confirm create schedule

Specify substitution properties

Specify values for substitution properties for this job.

| Property | Value |
|------------------|---|
| checkPoint | 10 |
| debugEnabled | false |
| fileEncoding | 8859_1 |
| inputDataStream | /tmp/ivtJobs/input-text.txt |
| numberRecords | 100 |
| outputDataStream | /tmp/ivtJobs/output-text.txt |
| perfEnabled | true |
| supportclassIn | com.ibm.websphere.batch.devframework.datast |
| supportclassOut | com.ibm.websphere.batch.devframework.datast |

< Back Next > Finish Cancel

Figure 21-15 Values for substitution properties

Click **Next**.

4. Review the scheduler configuration settings, and click **Finish**.

For more information and examples about using the Job management console, visit the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/welc6tech_computegrid_adm.html

21.4.6 Using the command-line interface for batch jobs

The command-line interface interacts with the job scheduler to submit and manipulate a batch job. In this section, we describe a sample scenario to verify the status of the jobs and schedules created in 21.4.5, “Using the job management console” on page 780.

Showing the status of batch jobs

To list the status of previously submitted jobs:

1. Connect to the server where the job scheduler is active.
2. Open a command prompt, and execute: `cd <WAS_INSTALL_ROOT>/bin.`
3. Execute the following command:

- UNIX platforms:

```
./lrcmd.sh -cmd=status -host=<job_scheduler_host> -port=<job_scheduler_port>  
-userid=<userid_job_scheduler> -password=<password_job_scheduler>
```

Example:

```
./lrcmd.sh -cmd=status -host=saw211-RHEL3 -port=9080 -userid=user1  
-password=batch
```

- Windows:

```
lrcmd.bat -cmd=status -host=<job_scheduler_host> -port=<job_scheduler_port>  
-userid=<userid_job_scheduler> -password=<password_job_scheduler>
```

Example 21-5 shows the expected output with the status of the batch jobs.

Example 21-5 Output from the lrcmd command for status of batch jobs

```
CWLRB4940I: com.ibm.ws.batch.wsbatch : -cmd=status -host=saw211-RHEL3 -port=9080  
-userid=user1 -password=*****
```

```
CWLRB5000I: Wed Jun 27 18:07:10 EDT 2012 : com.ibm.ws.batch.wsbatch : response to  
status
```

```
CWLRB3060I: [2012-06-27 15:07:16.459] [XDCGIVT:00000] [pending submit] [Batch]  
[user1] [] []
```

```
CWLRB3060I: [2012-06-27 17:42:50.050] [XDCGIVT:00004] [ended] [Batch] [user1]  
[saw211-RHEL3Node01] [itsoBatch]
```

```
CWLRB3060I: [2012-06-27 17:42:55.314] [XDCGIVT:00005] [pending submit] [Batch]  
[user1] [] []
```

```
CWLRB3060I: [2012-06-27 17:44:05.537] [XDCGIVT:00006] [pending submit] [Batch]  
[user1] [] []
```

```
CWLRB3060I: [2012-06-27 17:52:39.120] [XDCGIVT:00007] [pending submit] [Batch]  
[user1] [] []
```


CWLRB3060I: [2012-06-27 18:06:30.249] [XDCGIVT:00008] [pending submit] [Batch]
[user1] [] []

Viewing details of job schedules

To view the details of previously created job schedules:

1. Connect to the server where the job scheduler is active.
2. Open a command prompt, and execute: `cd <WAS_INSTALL_ROOT>/bin.`
3. Execute the following command:

- UNIX platforms:

```
./lrcmd.sh -cmd=getRecurringRequestDetails -request=<request_name>  
-host=<job_scheduler_host> -port=<job_scheduler_port>  
-userid=<userid_job_scheduler> -password=<password_job_scheduler>
```

For example:

```
./lrcmd.sh -cmd=getRecurringRequestDetails -request=weeklySchedule  
-host=saw211-RHEL3 -port=9080 -userid=wasadmin -password=need2reset
```

- Windows platforms:

```
lrcmd.bat -cmd=getRecurringRequestDetails -request=<request_name>  
-host=<job_scheduler_host> -port=<job_scheduler_port>  
-userid=<userid_job_scheduler> -password=<password_job_scheduler>
```

Example 21-6 shows a sample of the expected output, showing the details of a batch job schedule.

Example 21-6 Output from the lrcmd command showing details of batch job schedule

```
[root@saw211-RHEL3 bin]# ./lrcmd.sh -cmd=getRecurringRequestDetails  
-request=weeklySchedule -host=saw211-RHEL3 -port=9080 -userid=wasadmin  
-password=need2reset
```

```
CWLRB4940I: com.ibm.ws.batch.wsbatch : -cmd=getRecurringRequestDetails  
-request=weeklySchedule -host=saw211-RHEL3 -port=9080 -userid=wasadmin  
-password=*****
```

```
CWLRB5000I: Thu Jun 28 09:27:13 EDT 2012 : com.ibm.ws.batch.wsbatch : response to  
getRecurringRequestDetails
```

```
CWLRB5430I: [weeklySchedule] [2012-07-01 22:00:00] [weekly]  
[inputDataStream="/tmp/ivtJobs/input-text.txt"  
supportClassOut="com.ibm.websphere.batch.devframework.datastreams.patterns.TextFile  
eWriter" checkPoint="10" perfEnabled="true" numberRecords="100"  
outputDataStream="/tmp/ivtJobs/output-text.txt"  
supportClassIn="com.ibm.websphere.batch.devframework.datastreams.patterns.TextFile  
Reader" debugEnabled="false" fileEncoding="8859_1"] []
```

21.4.7 Checking the batch job logs

After the job is submitted from the Job management console, you can verify the logs to determine if the execution was successful.

The job logs are available in the following folder:

<WAS_INSTALL_ROOT>/profiles/<PROFILE_NAME>/joblogs

There are two logs to be verified:

► *part.0.log*

This log shows the initial load and dispatch information for the job, including the dispatch to the grid endpoint that will execute the process. Example 21-7 shows a snippet of the log file.

Example 21-7 part.0.log file

```
CWLRB5684I: [06/20/12 14:22:23:411 EDT] Job XDCGIVT:00000 is queued for
execution
CWLRB5586I: [06/20/12 14:22:23:484 EDT] CWLRS6006I: Job class Default,
Importance 8, Service Class null, Service Goal Type 0, Application Type j2ee,
Submitter user1.
CWLRB5586I: [06/20/12 14:22:23:484 EDT] CWLRS6007I: Job Arrival Time 6/20/12
2:22 PM, Goal Max Completion Time 0, Goal Max Queue Time 0, Breach Time 6/21/12
2:22 PM.
CWLRB5586I: [06/20/12 14:22:23:485 EDT] CWLRS6021I: List of eligible endpoints
to execute the job: saw211-RHEL3Node01/batchJVM01.
CWLRB5586I: [06/20/12 14:22:23:486 EDT] CWLRS6011I: APC is not active. GAP will
make the endpoint selection.
CWLRB5586I: [06/20/12 14:22:24:863 EDT] CWLRS6013I: GAP is dispatching job
XDCGIVT:00000. Job queue time 1.399 seconds.
CWLRB3090I: [06/20/12 14:22:25:440 EDT] Job XDCGIVT:00000 is dispatched to
endpoint saw211-RHEL3Node01/batchJVM01: result: 0
```

► *part.2.log*

This output includes any application generated output directed to the System.out and System.err output streams. Example 21-8 shows a snippet of the log file.

Example 21-8 part.2.log file

```
CWLRB5610I: [06/20/12 14:22:30:423 EDT] Firing IVTStep3 results algorithm
com.ibm.wsspi.batch.resultsalgorithms.jobsum: [RC 0] [jobRC 0]
CWLRB5624I: [06/20/12 14:22:30:503 EDT] Stopping step IVTStep3 chkpt
checkpoint. User transaction status: STATUS_ACTIVE
CWLRB5602I: [06/20/12 14:22:30:607 EDT] Closing IVTStep3 batch data stream:
inputStream
CWLRB5602I: [06/20/12 14:22:30:608 EDT] Closing IVTStep3 batch data stream:
generatedOutputInputStream
CWLRB5604I: [06/20/12 14:22:30:609 EDT] Freeing IVTStep3 batch data stream:
inputStream
CWLRB5604I: [06/20/12 14:22:30:609 EDT] Freeing IVTStep3 batch data stream:
generatedOutputInputStream
CWLRB5854I: [06/20/12 14:22:30:610 EDT] Job Step [XDCGIVT:00000,IVTStep3]:
Metric = clock Value = 00:00:00:005
CWLRB5854I: [06/20/12 14:22:30:611 EDT] Job Step [XDCGIVT:00000,IVTStep3]:
Metric = retry Value = 0
CWLRB5844I: [06/20/12 14:22:30:611 EDT] Job Step Batch Data Stream
[XDCGIVT:00000,IVTStep3,generatedOutputInputStream]: Metric = skip Value = 0
CWLRB5844I: [06/20/12 14:22:30:612 EDT] Job Step Batch Data Stream
[XDCGIVT:00000,IVTStep3,generatedOutputInputStream]: Metric = rps Value =
484,027
```

CWLRB5844I: [06/20/12 14:22:30:613 EDT] Job Step Batch Data Stream
[XDCGIVT:00000,IVTStep3,inputStream]: Metric = skip Value = 0
CWLRB5844I: [06/20/12 14:22:30:614 EDT] Job Step Batch Data Stream
[XDCGIVT:00000,IVTStep3,inputStream]: Metric = rps Value = 428,816
CWLRB2600I: [06/20/12 14:22:30:614 EDT] [06/20/12 14:22:30:614 EDT] Job
[XDCGIVT:00000] Step [IVTStep3] completed normally rc=0.
CWLRB3800I: [06/20/12 14:22:30:621 EDT] Job [XDCGIVT:00000] ended normally.



Understanding class loaders

Understanding how class loaders work is critical to packaging and deploying applications. Failure to configure class loaders properly results in a cascade of class loading exceptions (such as `ClassNotFoundException`) when trying to start your application or even at run time.

In this chapter, we explain class loaders and how to customize the behavior of the WebSphere class loaders to suit your particular application's requirements. The chapter concludes with an example designed to illustrate these concepts.

We cover the following topics:

- ▶ JVM class loaders
- ▶ WebSphere Application Server and Java EE application class loaders
- ▶ Configuring class loaders for Java EE applications
- ▶ Learning class loaders for Java EE by example
- ▶ OSGi class loaders

22.1 JVM class loaders

Class loaders enable the Java virtual machine (JVM) to load classes. Given the name of a class, the class loader locates the definition of this class. Each Java class must be loaded by a class loader.

When you start a JVM, it uses the following class loaders:

- ▶ The *bootstrap* class loader loads only the core Java libraries in the `Java_home/jre/lib` directory. This class loader, which is part of the core JVM, is written in native code.
- ▶ The *extensions* class loader loads the code in the extensions directories (`Java_home/jre/lib/ext` or any other directory that is specified by the `java.ext.dirs` system property). This class loader is implemented by the `sun.misc.Launcher$ExtClassLoader` class.
- ▶ The *application class loader* loads code that is found on `java.class.path`, which ultimately maps to the system `CLASSPATH` variable. This class loader is implemented by the `sun.misc.Launcher$AppClassLoader` class.

Note: Keep in mind that each JVM has its own set of class loaders. In an environment that is hosting multiple application servers (JVMs), the class loaders for the JVMs are completely separate even if they are running on the same physical machine.

The *parent-delegation model* is a key concept to understand when dealing with class loaders. It states that a class loader delegates class loading to its parent before trying to load the class itself. The parent class loader can be either another custom class loader or the bootstrap class loader. However, a class loader can delegate requests only to its parent class loader and never to its child class loaders. A class loader can go up the hierarchy but never down.

The extensions class loader is the parent for the application class loader. The bootstrap class loader is the parent for the extensions class loader. Figure 22-1 shows the class loaders hierarchy.

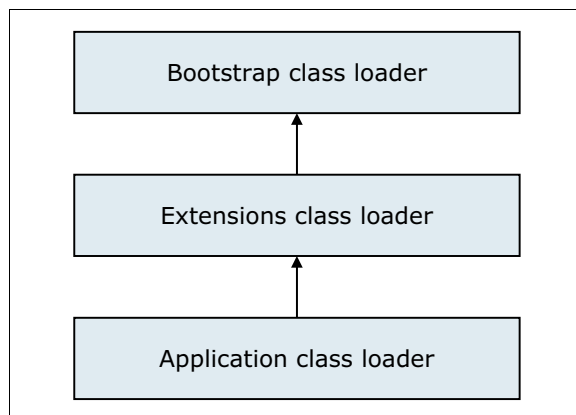


Figure 22-1 Java class loaders hierarchy

If the application class loader needs to load a class, it first delegates to the extensions class loader, which in turn delegates to the bootstrap class loader. If the parent class loader cannot load the class, the child class loader tries to find the class in its own repository. In this manner, a class loader loads only classes that its ancestors cannot load.

22.2 WebSphere Application Server and Java EE application class loaders

When working with Java Platform, Enterprise Edition (Java EE) applications, two additional types of class loaders are involved:

- ▶ The application server class loaders, which loads all of the classes that are needed for the application server in which the enterprise applications are running.
- ▶ The application class loaders, which loads the application classes as defined in the web.xml file.

Figure 22-2 shows a typical Java EE class loader.

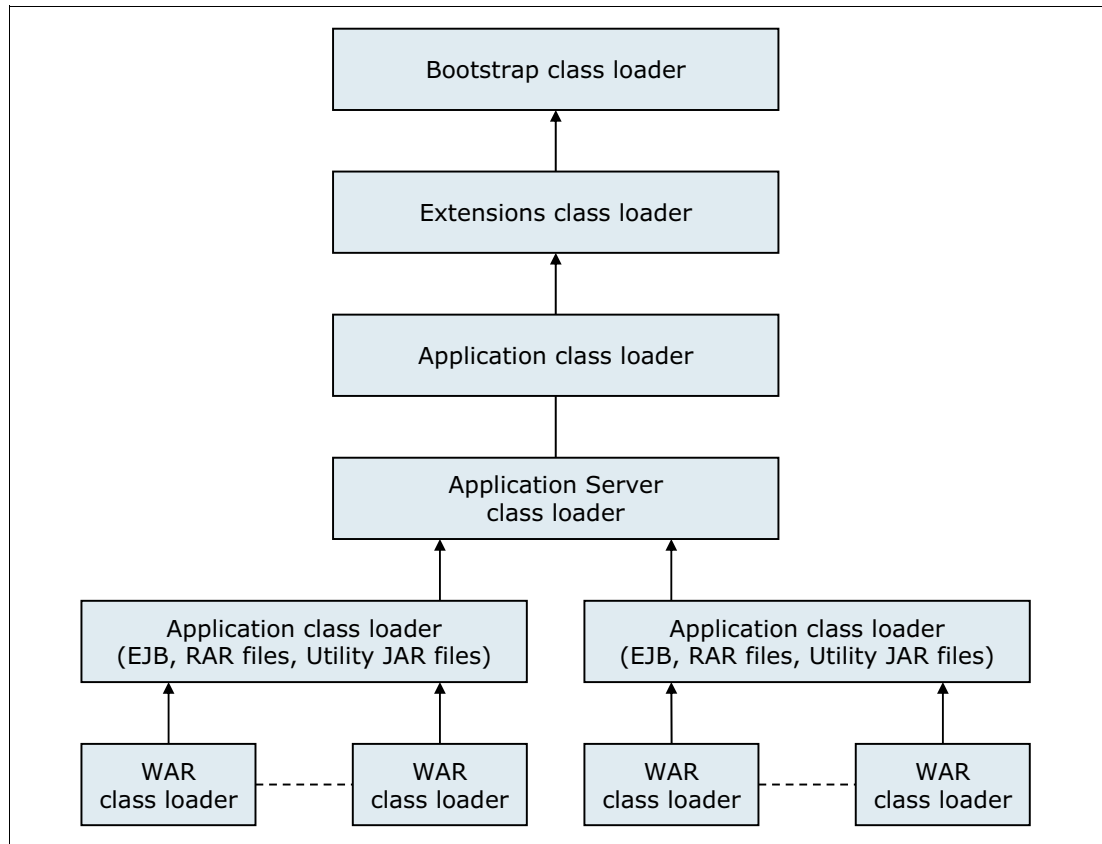


Figure 22-2 Java EE class loading

WebSphere Application Server provides several custom delegated class loaders, similar to those class loaders shown in 22.1, “JVM class loaders” on page 790, but it implements the extensions as OSGi packages, as shown in Figure 22-3 on page 792.

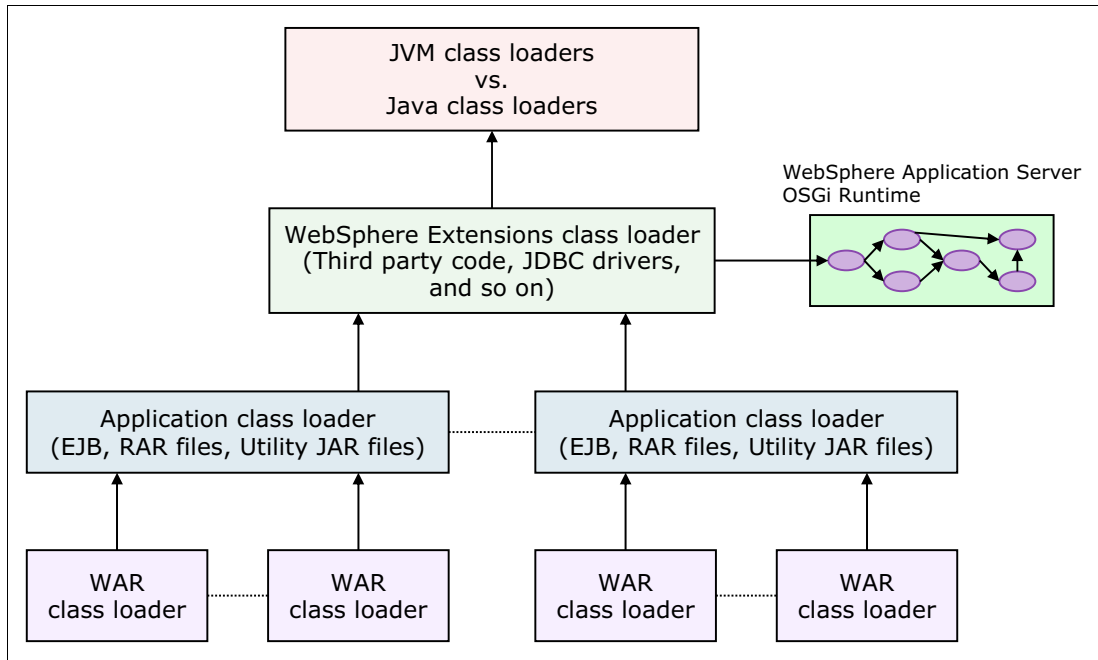


Figure 22-3 WebSphere class loaders hierarchy

The top box represents the Java class loaders (bootstrap, extensions, and application). WebSphere loads just enough here to get itself bootstrapped and to initialize the WebSphere extensions class loader.

We provide information about the other components of this hierarchy in the sections that follow.

22.2.1 WebSphere extensions class loader

The WebSphere extensions class loader is where WebSphere Application Server itself is loaded. WebSphere is packaged as a set of OSGi bundles with each OSGi bundle loaded separately by its own class loader. This network of OSGi class loaders is then connected to the extensions class loader, and the remainder of the class loader hierarchy through an OSGi gateway class loader.

Beginning with WebSphere V6.1, extensions began using OSGi packaging and the runtime classes are stored in the *install_root/plugins* directory.

The class path that is used by the extensions class loader is retrieved from the `ws.ext.dirs` system property, which is initially derived from the `WAS_EXT_DIRS` environment variable set in the `setupCmdLine` script file.

Each directory listed in the `ws.ext.dirs` environment variable is added to the WebSphere extensions class loaders class path, and every `.jar` file and `.zip` file in the directory is added to the class path.

Although the classes and `installedChannels` directories no longer exist in the *install_root* directory, the `setupCmdLine` script still adds them to the extensions class path. Thus, if you added your own JAR files to one of these directories in previous releases, you can create this directory and add your JAR files to it, and the JAR files are still loaded by the extensions class loader. However, you must avoid this situation by migrating away from such a setup.

Alternatively, if you developed Java applications that rely on the WebSphere JAR files that were in the *install_root/lib* directory prior to V6.1, you need to modify your application to retain compatibility. WebSphere Application Server provides the following thin client libraries that are designed specifically for such applications:

- ▶ The administrative client library
- ▶ The web services client library

You can find these thin client libraries in the *install_root/runtimes* directory:

- ▶ `com.ibm.ws.admin.client_8.5.0.jar`
- ▶ `com.ibm.ws.webservices.thinclient_8.5.0.jar`

These libraries provide everything your application might need to connect to and work with WebSphere. WebSphere Application Server V8.5 provides the ability to restrict access to internal WebSphere classes so that your applications do not make unsupported calls to WebSphere classes that are not published in the official WebSphere Application Server API. To restrict access to internal WebSphere classes, select the **Access to internal server classes** option in the main Application Server configuration page under **Servers** → **Server Types** → **WebSphere Application servers** → **YourAppSrvName**.

The default setting for this option is *Allow*, meaning that your applications can make unrestricted calls to non-public internal WebSphere classes. This function might be prohibited in future releases. Therefore, as an administrator, consider switching this setting to *Restrict* to see whether applications still work as expected. If applications depend on non-public WebSphere internal classes, you will receive a `ClassNotFoundException`. In this case, you can switch back to the *Allow* setting. To retain compatibility with future WebSphere Application Server releases, developers can migrate applications so that the applications do not make unsupported calls to the WebSphere internal classes.

22.2.2 Application and web module class loaders

Java EE 6 applications consist of the following primary elements:

- ▶ Web modules
- ▶ EJB modules
- ▶ Application client modules
- ▶ Resource adapter archives (RAR files)
- ▶ Utility JAR files

Utility JAR files contain code that is used by both EJB and servlets. Utility frameworks, such as `log4j`, are good examples of a utility JAR file.

EJB modules, utility JAR files, resource adapter files, and shared libraries that are associated with an application are always grouped together into the same class loader. This class loader is called the *application class loader*. Depending on the class loader policy, this class loader can be shared by multiple applications (EAR files) or can be unique for each application, which is the default.

By default, web modules receive their own class loader, a WAR class loader, to load the contents of the WEB-INF/classes and WEB-INF/lib directories. You can modify the default behavior by changing the application's WAR class loader policy. You can find this policy setting in the administrative console by clicking **Applications** → **WebSphere enterprise applications** → *application_name* → **Class loading and update detection** → **WAR class loader policy**, which opens the window shown in Figure 22-4.

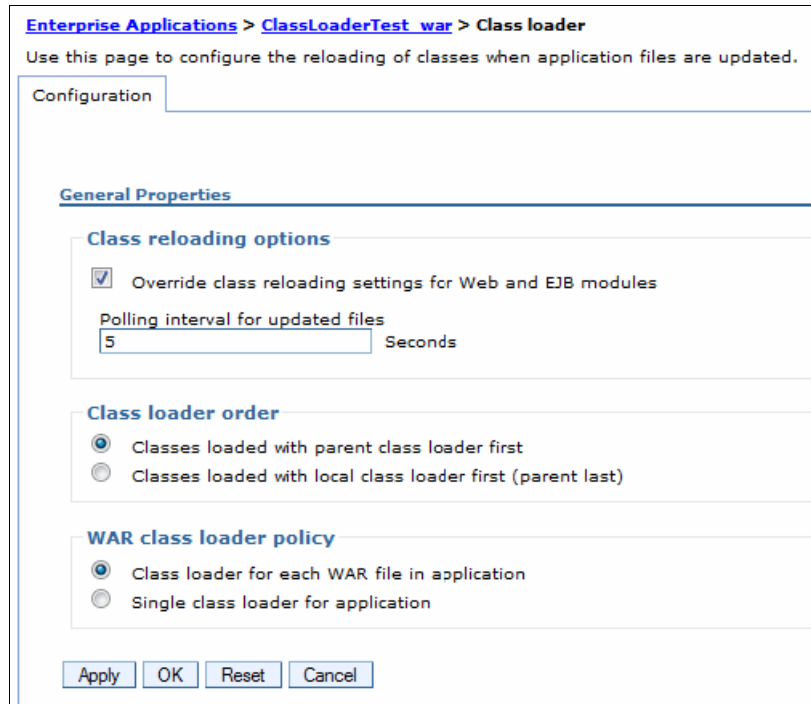


Figure 22-4 WAR class loader policy settings

The default WAR class loader policy setting is the Class loader for each WAR file in application option. This setting is called *Module* in previous releases and in the application deployment descriptor, as viewed in Rational Application Developer.

If the WAR class loader policy is set to the Single class loader for application option, the web module contents are loaded by the application class loader in addition to the EJB, RAR files, utility JAR files, and shared libraries. The application class loader is the parent of the WAR class loader.

The application and the WAR class loaders are reloadable class loaders. They monitor changes in the application code to reload modified classes automatically. You can modify this behavior at deployment time.

22.2.3 Handling Java Native Interface code

Because a JVM has only a single address space and because native code can be loaded only once per address space, the JVM specification states that native code can be loaded only by one class loader in a JVM. This design might cause a problem if, for example, you have an application (EAR file) with two web modules that both need to load the same native code through a Java Native Interface (JNI). Only the web module that first loads the library will succeed.

To solve this problem, you can break out just the few lines of Java code that load the native code into a class on its own and place this file on WebSphere's application class loader (in a utility JAR file). However, if you deploy multiple such applications (EAR files) to the same application server, you have to place the class file on the WebSphere extensions class loader instead to ensure that the native code is loaded only once per JVM.

If the native code is placed on a reloadable class loader (such as the application class loader or the WAR class loader), it is important that the native code can unload itself properly if the Java code needs to reload. WebSphere has no control over the native code, and if the native code does not unload and load properly, the application might fail.

If one native library depends on another library, the handling of JNI code can become even more complicated. For more details and troubleshooting, go to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=trun_classload

22.3 Configuring class loaders for Java EE applications

We provided an overview of WebSphere class loaders and how they work together to load classes. WebSphere Application Server includes settings that allow you to influence the class loader behavior. We provide information about these options in this section.

22.3.1 Application server class loader policies

For each application server in the system, you can set the class loader policy to either *Single* or *Multiple*. From the administrative console, click **Servers** → **Server Types** → **WebSphere application servers** → **server_name**. Next, on the Configuration tab under the Server-specific Application Settings section, select the appropriate class loader policy.

When the application server class loader policy is set to Single, a single application class loader is used to load all EJB, utility JAR files, and shared libraries within the application server (JVM). If the WAR class loader policy is set to use the Single class loader for application option, the web module contents for this particular application are also loaded by this single class loader.

When the application server class loader policy is set to Multiple, which is the default, each application receives its own class loader for loading EJB, utility JAR files, and shared libraries. Depending on whether the WAR class loader policy is set to use the Class loader for each WAR file in application option or the Single class loader for application option, the web module might or might not receive its own class loader.

Here is an example to illustrate. Suppose that you have two applications, *Application1* and *Application2*, running in the same application server. Each application has one EJB module, one utility JAR file, and two web modules. If the application server has its class loader policy set to Multiple and the class loader policy for all the web modules are set to use the Class loader for each WAR file in application option, the result is as shown in Figure 22-5 on page 796.

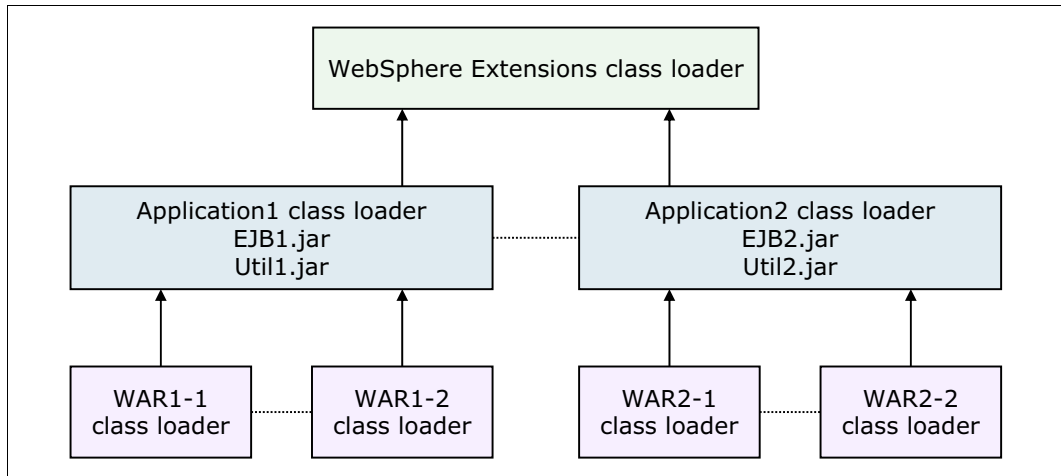


Figure 22-5 Class loader policies - Example 1

Each application is completely separated from the other application, and each web module is completely separated from the other web module in the same application. WebSphere's default class loader policies results in total isolation between the applications and the modules.

Now, if we change the class loader policy for the WAR2-2 module to use the Single class loader for application option, the result is shown in Figure 22-6.

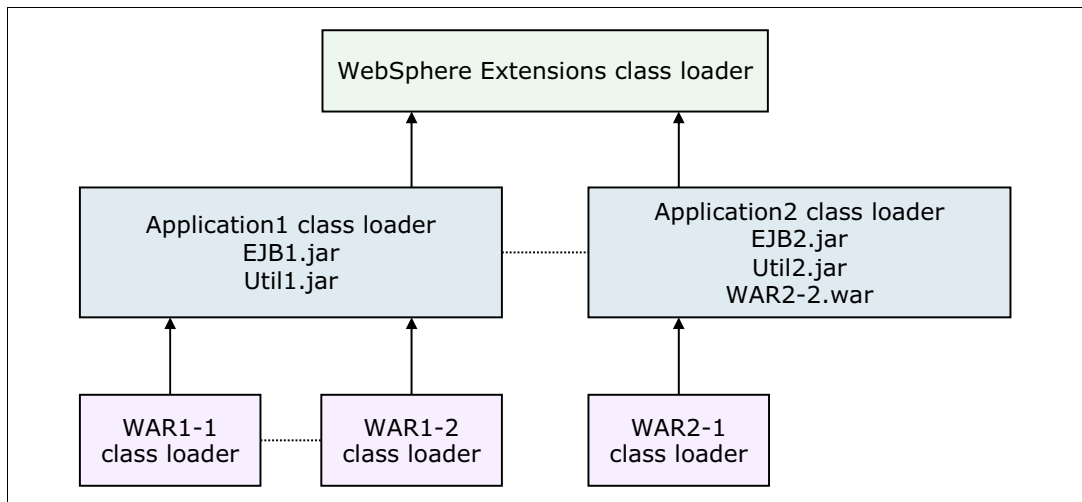


Figure 22-6 Class loader policies - Example 2

Web module WAR2-2 is loaded by Application2's class loader and classes, and for example, classes in the Util2.jar file can see classes in WAR2-2's /WEB-INF/classes and /WEB-INF/lib directories.

As a last example, if we change the class loader policy for the application server to Single and also change the class loader policy for WAR2-1 to use the Single class loader for application option, the result is as shown in Figure 22-7 on page 797.

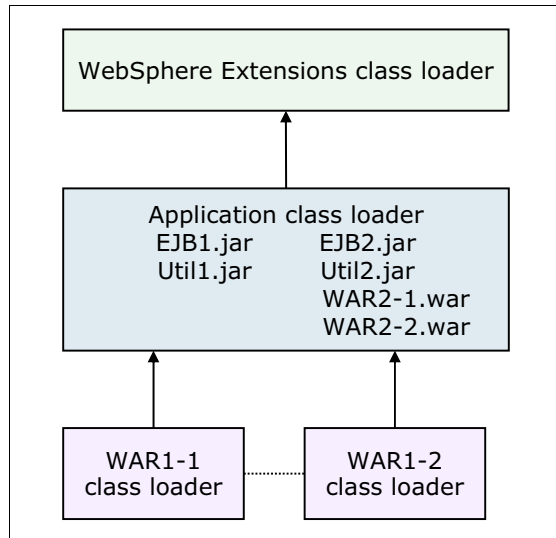


Figure 22-7 Class loader policies - Example 3

There is now only a single application class loader loading classes for both Application1 and Application2. Classes in the Util1.jar file can see classes in the EJB2.jar, Util2.jar, WAR2-1.war, and WAR2-2.war files. However, the classes that are loaded by the application class loader still cannot see the classes in the WAR1-1 and WAR1-2 modules because a class loader can find only classes by going up the hierarchy, never down.

22.3.2 Class loading and delegation mode

The WebSphere application class loader and WAR class loader both have a setting called the *class loader order*. This setting determines whether the class loader order follows the normal parent delegation model or whether the class loader overrides it.

The class loading mode uses one of the following options:

- ▶ Classes loaded with parent class loader first (default)
- ▶ Classes loaded with local class loader first (parent last)

In previous WebSphere releases, these settings were called PARENT_FIRST and PARENT_LAST, respectively.

If the class loading policy is set to classes loaded with local class loader first (parent last), the class loader attempts to load classes from its local class path before delegating the class loading to its parent. This policy allows an application class loader to override and provide its own version of a class that exists in the parent class loader.

Terminology note: The settings page for a web module includes the following options for class loader order:

- ▶ Classes loaded with parent class loader first
- ▶ Classes loaded with local class loader first (parent last)

However, in this context, the *local class loader* really refers to the WAR class loader, so the Classes loaded with local class loader first option actually refers to classes loaded with *WAR class loader* first.

Assume that you have an application, similar to Application1 in the previous examples, and it uses the popular log4j package to perform logging from both the EJB module and the two web modules. Also, assume that each module has its own, unique log4j.properties file that is packaged into the module. You can configure log4j as a utility JAR file so that you have only a single copy of it in the EAR file.

However, if you use this configuration, you might be surprised to see that all modules, including the web modules, load the log4j.properties file from the EJB module. The reason is that when a web module initializes the log4j package, the log4j classes are loaded by the application class loader. The log4j package is configured as a utility JAR file. Thus, it looks for a log4j.properties file on its class path and finds it in the EJB module.

Even if you do not use log4j for logging from the EJB module and the EJB module does not, therefore, contain a log4j.properties file, log4j does not find the log4j.properties file in any of the web modules. Remember that a class loader can find classes only by going up the hierarchy, never down.

To solve this problem, use one of the following approaches:

- ▶ Create a separate file, for example, a Resource.jar file. Configure the file as a utility JAR file, move all the log4j.properties files from the modules into this file, and make their names unique (such as war1-1_log4j.properties, war1-2_log4j.properties, and ejb1_log4j.properties). When initializing log4j from each module, tell it to load the proper configuration file for the module instead of the default file (log4j.properties).
- ▶ Keep the log4j.properties file for the web modules in its original directory (/WEB-INF/classes), add the log4j.jar file to both web modules (in the /WEB-INF/lib directory), and set the class loading mode for the web modules to use the Classes loaded with local class loader first (parent last) option. When initializing log4j from a web module, it loads the log4j.jar file from the module itself, and log4j finds the log4j.properties file on its local class path, which is the web module itself. When the EJB module initializes log4j, it loads from the application class loader, and it finds the log4j.properties file on the same class path, which is the one in the EJB1.jar file.
- ▶ If possible, merge all log4j.properties files into one file and place this file on the application class loader (for example, in a Resource.jar file).

Singletons: The singleton pattern is used to ensure that a class is instantiated only once. However, *once only* means *once for each class loader*. If you have a singleton instantiated in two separate web modules, two separate instances of this class are created, that is, one for each WAR class loader. Thus, in a multi-class loader environment, take care when implementing singletons.

22.3.3 Shared libraries

Shared libraries are files used by multiple applications. Examples of shared libraries are commonly used frameworks, such as Apache Struts or log4j. You use shared libraries typically to point to a set of JAR files and associate those JAR files to an application, a web module, or the class loader of an application server. Shared libraries are especially useful when you have different versions of the same framework that you want to associate to different applications.

Shared libraries are defined using the administration tools. They consist of a symbolic name, a Java class path, and a native path for loading JNI libraries. They can be defined at the cell, node, server, or cluster level. However, simply defining a library does not cause the library to be loaded. You must associate the library to an application, a web module, or the class loader

of an application server for the classes represented by the shared library to be loaded. Associating the library to the class loader of an application server makes the library available to all applications on the server.

Note: If you associate a shared library to an application, do not associate the same library to the class loader of an application server.

You can associate the shared library to an application using one of the following methods:

- ▶ You can use the administrative console. The library is added by using the Shared libraries references link under the References section for the enterprise application.
- ▶ You can use the manifest file of the application and the shared library. The shared library contains a manifest file that identifies it as an extension. The dependency to the library is declared in the application's manifest file by listing the library extension name in an extension list.

For more information about this method, visit the Information Center page:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=ccws_installoptpkg

Shared files are associated with the class loader of an application server using the administrative tools in the Server Infrastructure section. To define a new class loader, complete the following steps:

1. Under the Application Server Configuration tab, click **Java and Process Management**, and select **Class loader**.
2. Click **New**.
3. After you define a new class loader, you can modify it. You can also use the Shared library references link to associate it to the shared libraries that you need.

Refer to 22.4.4, “Example 4: Sharing utility JAR files using shared libraries” on page 805 for more details.

Tip: When designing your application, consider how every class is loaded to avoid having to add classes in different directories to make the application work.

22.3.4 Class loader viewer

If the class loader viewer service is not enabled, the class loader viewer displays only the hierarchy of class loaders and their class paths, but not the classes that are actually loaded by each of the class loaders. Thus, in this case, the search capability of the class loader viewer is lost.

To enable the class loader viewer service, click **Servers** → **Server Types** → **WebSphere application server** → *server_name*, and click **Class Loader Viewer Service** under the **Additional Properties** link. Then, select the **Enable service at server startup** option. You need to restart the application server for this setting to take effect.

In the next section, we provide examples of how to work with the different class loader settings, and then we use the class loader viewer to illustrate the results.

For more information about troubleshooting class loading problems, visit the article *Demystifying class loading problems* at:

http://www.ibm.com/developerworks/java/library/j-dclp1/?S_TACT=106AH10W&S_CMP=NC

22.4 Learning class loaders for Java EE by example

We described all of the different options for influencing class loader behavior. In this section, we take an example and use the options that we discussed so that you can better evaluate the best solution for your applications.

We created a simple application with one servlet and one EJB. Both call a class, `VersionChecker`, shown in Example 22-1. This class can print the class loader that was used to load the class. The `VersionChecker` class also has an internal value that can be printed to check the version of the class that we are using. This information is used later to demonstrate the use of multiple versions of the same utility JAR file.

Example 22-1 VersionChecker class source code

```
package com.itso.classloaders;

public class VersionChecker {
    static final public String classVersion = "v1.0";

    public String getInfo() {
        return ("VersionChecker is " + classVersion +
            ". Loaded by " + this.getClass().getClassLoader());
    }
}
```

After being installed, the application can be invoked through `http://localhost:9080/ClassLoaderTestWeb/ExampleServlet`. This invokes the `ExampleServlet`, which calls `VersionChecker` and then displays the classloader.

The `VersionCheckerV1.jar` file contains the `VersionChecker` class file that returns Version number 1.0. For all of the following tests, we have, unless otherwise noted, left the class loader policies and loading modes to their defaults. In other words, we have one class loader for the application and one for the WAR file. Both have their delegation modes set to use the `Classes loaded with parent class loader first` option.

22.4.1 Example 1: Simple web module packaging

For this example, we start with the assumption that our utility class is used only by a servlet. We placed the `VersionCheckerV1.jar` file under the `WEB-INF/lib` directory of the web module.

Tip: Place JAR files that are used by a single web module or a JAR file that *only* this web module can see in the `WEB-INF/lib` directory.

Example 22-2 shows the results of running the application with such a configuration.

Example 22-2 Class loader - Example 1

VersionChecker called from Servlet

VersionChecker is **v1.0**.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@6c2c0f37[war:ClassLoaderExample/ClassLoaderExampleWeb.war]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Ce1101/ClassLoaderExample.ear/ClassLoaderExampleWeb.war/WEB-INF/classes:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Ce1101/ClassLoaderExample.ear/ClassLoaderExampleWeb.war/WEB-INF/lib/VersionCheckerV1.jar:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Ce1101/ClassLoaderExample.ear/ClassLoaderExampleWeb.war

Parent:com.ibm.ws.classloader.CompoundClassLoader@f1a5f601[app:ClassLoaderExample]

Delegation Mode: **PARENT_FIRST**

We can learn the following information from this trace:

- ▶ The type of the WAR class loader is:

com.ibm.ws.classloader.CompoundClassLoader

- ▶ It searches classes in the following order:

ClassLoaderTestWeb.war/WEB-INF/classes

ClassLoaderTestWeb.war/WEB-INF/lib/VersionCheckerV1.jar

ClassLoaderTestWeb.war

The WEB-INF/classes folder holds unpacked resources (such as servlet classes, plain Java classes, and property files), and the WEB-INF/lib folder holds resources that are packaged as JAR files. You can choose to package your Java code in JAR files and place them in the WEB-INF/lib directory, or you can put them unpacked in the WEB-INF/classes directory. Both directories are on the same class path. Because we developed and exported our sample application from Rational Application Developer, our servlet goes into the WEB-INF/classes folder, because the Java classes are not packaged in a JAR file when exporting an application.

You can also put code or properties in the root of the WAR file. Note, however, that this folder is also the document root for the web server if the File Serving Servlet capabilities are enabled. In this case, any files in this folder are then accessible from a browser. According to the Java EE 6 specification, the WEB-INF folder is protected, which is why the classes and lib folders are under the WEB-INF folder.

The class loader class path is built dynamically at application start.

You can also use the class loader viewer to display the class loader. In the administrative console, click **Troubleshooting** → **Class Loader Viewer**. Next, expand **appserver_name** → **Applications** → **ClassLoaderTest** → **Web modules** → **ClassLoaderTestWeb.war**, as shown in Figure 22-8 on page 802.

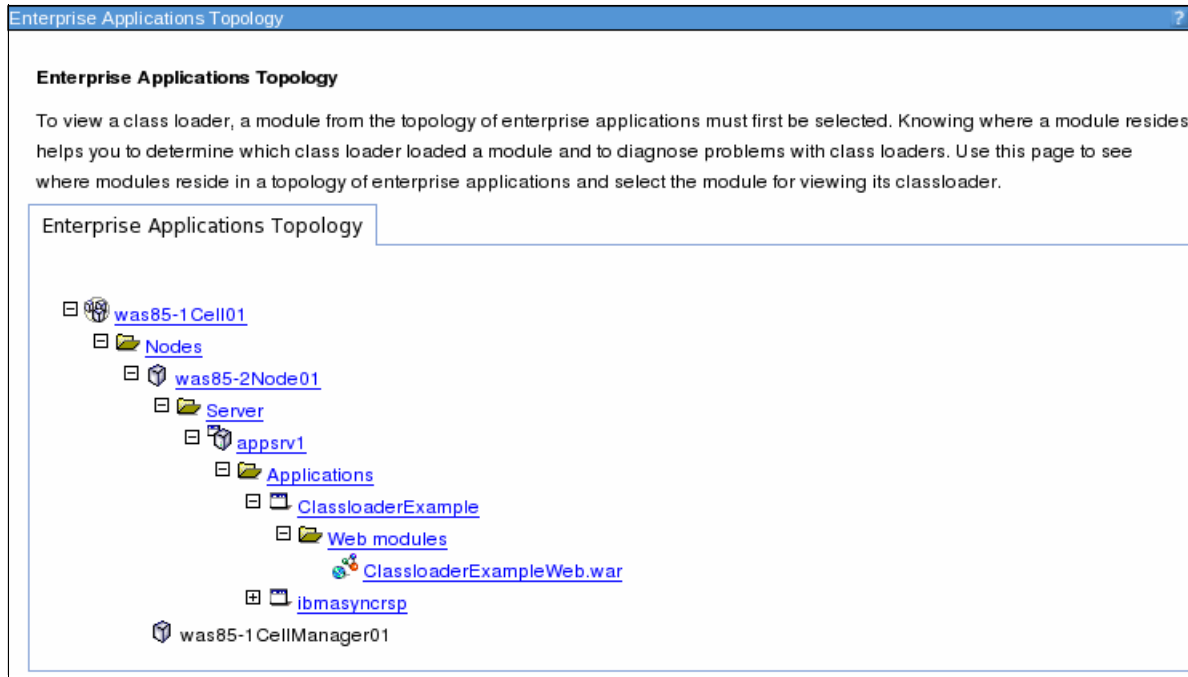


Figure 22-8 Class loader viewer showing applications tree

When the web module is expanded, the class loader viewer shows the hierarchy of class loaders, from the JDK extensions and application class loaders at the top to the WAR class loader at the bottom, called the *compound class loader* (see Figure 22-9).

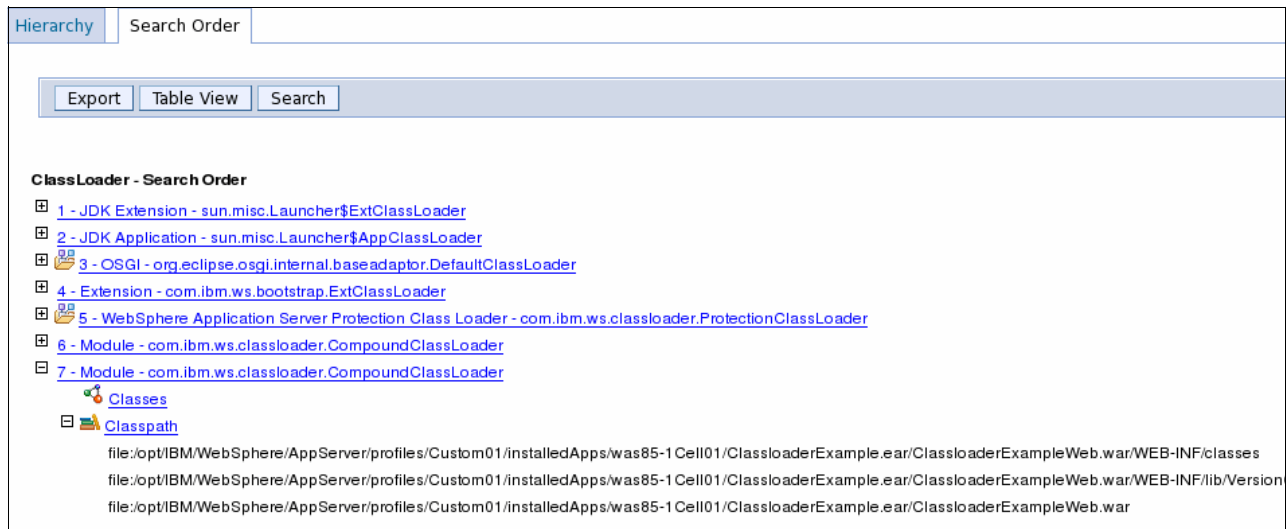


Figure 22-9 Class loader viewer showing class loader hierarchy

If you expand the class path for `com.ibm.ws.classloader.CompoundClassLoader`, you see the same information as the `VersionChecker` class prints (see Figure 22-5 on page 796).

Note: For the class loader viewer to display the classes that are loaded, it must be enabled, as described in 22.3.4, “Class loader viewer” on page 799.

The class loader viewer also has a table view that displays all of the class loaders and the classes that are loaded by each of them on a single page. The table view also displays the Delegation mode. For the Delegation mode, *true* means that classes are loaded with parent class loader first, and *false* means that classes are loaded with local class loader first (parent last) or the WAR class loader in the case of a web module (see Figure 22-10). The WAR class loader loaded our example servlet and the VersionChecker class, as expected. Table view is enabled by clicking **Table View** at the top of the class loader viewer.

| Module - com.ibm.ws.classloader.CompoundClassLoader | |
|---|--|
| Delegation | true |
| Classpath | file:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1 Cell01/ClassloaderExample.ear/ClassloaderExampleWeb.war/WEB-INF/classes |
| | file:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1 Cell01/ClassloaderExample.ear/ClassloaderExampleWeb.war/WEB-INF/lib/VersionCheckerV1.jar |
| | file:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1 Cell01/ClassloaderExample.ear/ClassloaderExampleWeb.war |

Figure 22-10 Class loader viewer table view

The class loader viewer also has a search feature where you can search for classes, JAR files, folders, and so on. This search capability can be particularly useful if you do not know which of the class loaders loaded a class in which you are interested. The search feature is case sensitive but allows wild cards. So, a search for **VersionChecker** finds our VersionChecker class.

22.4.2 Example 2: Adding an EJB module and utility jar

Next, we add an EJB to our application that also depends on the VersionChecker.jar file. For this task, we added a VersionCheckerV2.jar file to the root of our EAR file. The VersionChecker class in this JAR file returns Version 2.0. To make it available as a utility JAR on the extensions class loader, we add a reference to it in the EJB module's manifest file, as shown in Example 22-3.

Example 22-3 Updated MANIFEST.MF for EJB module

```
Manifest-Version: 1.0
Class-Path: VersionCheckerV2.jar
```

The result is that we now have a web module with a servlet in the WEB-INF/classes folder and the VersionCheckerV1.jar file in the WEB-INF/lib folder. We also have an EJB module that references the VersionCheckerV2.jar utility JAR file in the root of the EAR file.

Example 22-4 shows the test results.

Example 22-4 Class loader: Example 2

VersionChecker called from Servlet

VersionChecker is **v2.0**.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@22da9c28[app:ClassloaderExample]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/ClassloaderExample.ear/ClassloaderExampleEJB.jar:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/ClassloaderExample.ear/VersionCheckerV2.jar

```
Parent: com.ibm.ws.classloader.ProtectionClassLoader@5bfa0027
Delegation Mode: PARENT_FIRST
```

VersionChecker called from EJB

```
VersionChecker is v2.0.
Loaded by
com.ibm.ws.classloader.CompoundClassLoader@22da9c28[app:ClassLoaderExample]
Local ClassPath:
/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/ClassloaderExample.ear/ClassLoaderExampleEJB.jar:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/ClassLoaderExample.ear/VersionCheckerV2.jar
Parent: com.ibm.ws.classloader.ProtectionClassLoader@5bfa0027
Delegation Mode: PARENT_FIRST
```

As the results show, the VersionChecker is Version 2.0 when called both from the EJB module and the web module. The reason is that the WAR class loader delegates the request to its parent class loader instead of loading it itself. Thus, the utility JAR file is loaded by the same class loader regardless of whether it was called from the servlet or the EJB.

22.4.3 Example 3: Changing the WAR class loader delegation mode

For this example, we consider that we now want the web module to use the VersionCheckerV1.jar file from the WEB-INF/lib folder. For that task, we have to change the class loader delegation from parent first to parent last.

Set the delegation mode to PARENT_LAST by completing the following steps:

1. Select the **All applications** entry in the navigation area.
2. Select the **ClassLoaderExample** application.
3. Select **Manage modules** under the Modules section.
4. Select the **ClassLoaderExampleWeb** module.
5. Change the Class loader order to the **Classes loaded with local class loader first (parent last)** option. Remember, this entry must really be called classes loaded with *WAR class loader* first, as noted in 22.3.2, “Class loading and delegation mode” on page 797.
6. Click **OK**.
7. Save the configuration.
8. Restart the application.

The VersionCheckerV1 in the WEB-INF/lib folder returns a class version of 1.0. Example 22-5 shows that this is the version now used by the WAR file.

Example 22-5 Class loader - Example 3

VersionChecker called from Servlet

```
VersionChecker is v1.0.
Loaded by
com.ibm.ws.classloader.CompoundClassLoader@cf9e0ce5[war:ClassLoaderExample/ClassLoaderExampleWeb.war]
Local ClassPath:
/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/Classlo
```

```
aderExample.ear/ClassLoaderExampleWeb.war/WEB-INF/classes:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell01/ClassLoaderExample.ear/ClassLoaderExampleWeb.war/WEB-INF/lib/VersionCheckerV1.jar:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell01/ClassLoaderExample.ear/ClassLoaderExampleWeb.war
Parent:com.ibm.ws.classloader.CompoundClassLoader@cac8c5a0[app:ClassLoaderExample]
Delegation Mode: PARENT_LAST
```

VersionChecker called from EJB

```
VersionChecker is v2.0.
Loaded by
com.ibm.ws.classloader.CompoundClassLoader@cac8c5a0[app:ClassLoaderExample]
Local ClassPath:
/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell01/ClassLoaderExample.ear/ClassLoaderExampleEJB.jar:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell01/ClassLoaderExample.ear/VersionCheckerV2.jar
Parent: com.ibm.ws.classloader.ProtectionClassLoader@5bfa0027
Delegation Mode: PARENT_FIRST
```

Tip: Use this technique to specify that a web module must use a specific version of a library, such as Struts, or to override classes coming with the WebSphere runtime. Put the common version at the top of the hierarchy and the specialized version in the WEB-INF/lib folder.

Keep in mind that if the library being loaded in turn depends on other libraries, you must make sure these dependencies are compatible. This problem usually arises when the dependency classes are the same as those from libraries packaged and shipped in WebSphere Application Server. The dependency libraries must also be loaded overriding the internal classes to keep constancy of versions and compatibility.

If you use the search feature of the class loader viewer to search for **VersionChecker**, you see the two entries shown in Example 22-6.

Example 22-6 Class Loader Viewer search feature

WAS Module Compound Class Loader (WAR class loader):

```
file: / opt / IBM / WebSphere / AppServer / profiles / Custom01 / installedApps / was85-1Cell01 / ClassloaderExample.ear / ClassloaderExampleWeb.war / WEB-INF / lib / VersionCheckerV1.jar
```

WAS Module Jar Class Loader (Application class loader):

```
file: / opt / IBM / WebSphere / AppServer / profiles / Custom01 / installedApps / was85-1Cell01 / ClassloaderExample.ear / VersionCheckerV2.jar
```

22.4.4 Example 4: Sharing utility JAR files using shared libraries

In this example, the VersionCheckerV2.jar file is used by a single application. If you want to share the JAR file among multiple applications, you can package it within each EAR file. However, changes to this utility JAR file require that you redeploy all applications. To avoid this situation, you can externalize global utility JAR files using a *shared library*.

Shared libraries can be defined at the cell, node, application server, and cluster levels. After you define a shared library, you must associate it to the class loader of an application server, application, or individual web module. Depending on the target to which the shared library is assigned, WebSphere uses the appropriate class loader to load the shared library.

You can define as many shared libraries as you want. You can also associate multiple shared libraries with an application, web module, or application server.

Using shared libraries at the application level

To define a shared library named *VersionCheckerV2_SharedLib* and associate it to our *ClassLoaderTest* application:

1. In the administrative console, click **Environment** → **Shared Libraries**.
2. Select the scope at which you want this shared library to be defined, such as Cell, and click **New**.
3. Specify the following properties, as shown in Figure 22-11 on page 807:

| | |
|----------------------------|--|
| Name | Enter VersionCheckerV2_SharedLib. |
| Class path | Enter the list of entries on the class path. Press Enter between each entry. Note that if you need to provide an absolute path, you can use WebSphere variables, such as %FRAMEWORK_JARS%/VersionCheckerV2.jar. Make sure that you declare this variable at the same scope as the shared library for cell, node, server, or cluster. Also, if in a Network Deployment Environment, make sure that the contents of the directory referenced here are also present and accessible in the remote nodes. WebSphere Application Server does not replicate its contents if it is not in the master repository directory. |
| Native library path | Enter a list of DLLs and .so files for use by the JNI code. |

If you want to have only one instance of a version of a class shared among applications, select the **Use an isolated class loader for this shared library** option.

General Properties

* Scope
cells:was85-1Cell01

* Name
VersionCheckerV2_SharedLib

Description

* Classpath
/opt/IBM/WebSphere/Examples/

Native Library Path

Class Loading

Use an isolated class loader for this shared library

Apply OK Reset Cancel

Figure 22-11 Defining a shared library

4. Click **OK**.
5. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
6. Select the **ClassLoaderExample** application.
7. In References, select **Shared library references**. Select **ClassLoaderExample** in the Application row.
8. Click **Reference shared libraries** → **VersionCheckerV2_SharedLib**, and click the arrow button to move it to the Selected column.
9. Click **OK**.

The shared library configuration window for the ClassLoaderTest application now looks as shown in Figure 22-12 on page 808.

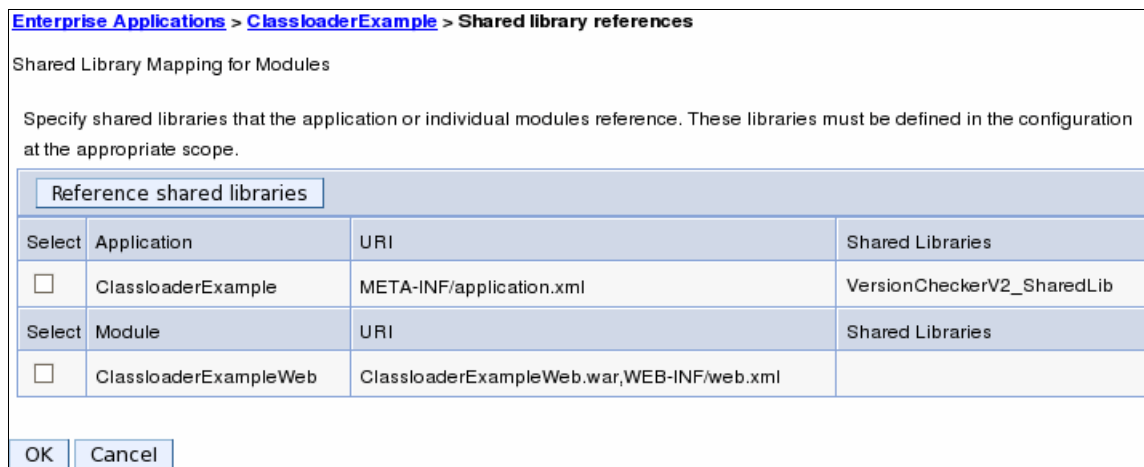


Figure 22-12 Shared library assigned to ClassLoaderTest application

10. Click **OK** and save the configuration.

If we remove the VersionCheckerV2.jar file from the root of the EAR file, remove the reference to it from the EJB module's manifest file, and restart the application server, we see the results shown in Example 22-7. Remember that the class loader order for the web module still uses the Classes loaded with local class loader first (parent last) option.

Example 22-7 Class loader - Example 5

VersionChecker called from Servlet

VersionChecker is **v1.0**.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@c779cc83[war:ClassloaderExample/ClassloaderExampleWeb.war]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/ClassloaderExample.ear/ClassloaderExampleWeb.war/WEB-INF/classes:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/ClassloaderExample.ear/ClassloaderExampleWeb.war/WEB-INF/lib/VersionCheckerV1.jar:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/ClassloaderExample.ear/ClassloaderExampleWeb.war

Parent: com.ibm.ws.classloader.CompoundClassLoader@727540a4[app:ClassloaderExample]

Delegation Mode: **PARENT_LAST**

VersionChecker called from EJB

VersionChecker is **v2.0**.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@727540a4[app:ClassloaderExample]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell101/ClassloaderExample.ear/ClassloaderExampleEJB.jar:/opt/IBM/WebSphere/Examples:/opt/IBM/WebSphere/Examples/VersionCheckerV2.jar

Parent: com.ibm.ws.classloader.ProtectionClassLoader@7c23bd5c

Delegation Mode: **PARENT_FIRST**

As expected, because of the delegation mode for the web module, the VersionCheckerV1.jar file was loaded when the servlet that is needed is the VersionChecker class. When the EJB needed the VersionChecker class, it was loaded from the shared library, which points to the /opt/IBM/WebSphere/Examples/VersionCheckerV2.jar file.

If we want the web module to also use the shared library, we restore the class loader order to the default Classes loaded with parent class loader first option for the web module.

Using shared libraries at the application server level

A shared library can also be associated with an application server. All applications deployed on this server see the code listed on that shared library.

To associate a shared library to an application server, you must first create an additional class loader for the application server, as follows:

1. Select an application server.
2. In the Server Infrastructure section, expand **Java and Process Management**. Select **Class loader**.
3. Choose **New**, and select a class loader order for this class loader, either the **Classes loaded with parent class loader first** option or the **Classes loaded with local class loader first (parent last)** option. Click **OK**.
4. Click the class loader that is created.
5. Click **Shared library references**.
6. Click **Add**, and select the library that you want to associate to this application server. Repeat this operation to associate multiple libraries to this class loader. For our example, we selected the **VersionCheckerV2_SharedLib** entry.
7. Click **OK**.
8. Save the configuration.
9. Restart the application server for the changes to take effect.

Because we attached the VersionCheckerV2 shared library to the class loader of the application server, we obtain the results shown in Example 22-8.

Example 22-8 Class loader - Example 6

VersionChecker called from Servlet

VersionChecker is **v1.0**.

Loaded by

com.ibm.ws.classloader.CompoundClassLoader@ed3b6b7[war:ClassLoaderExample/ClassLoaderExampleWeb.war]

Local ClassPath:

/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell01/ClassLoaderExample.ear/ClassLoaderExampleWeb.war/WEB-INF/classes:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell01/ClassLoaderExample.ear/ClassLoaderExampleWeb.war/WEB-INF/lib/VersionCheckerV1.jar:/opt/IBM/WebSphere/AppServer/profiles/Custom01/installedApps/was85-1Cell01/ClassLoaderExample.ear/ClassLoaderExampleWeb.war

Parent:

com.ibm.ws.classloader.CompoundClassLoader@e2af7a6b[app:ClassLoaderExample]

Delegation Mode: **PARENT_LAST**

VersionChecker called from EJB

VersionChecker is **v2.0**.
Loaded by com.ibm.ws.classloader.**ExtJarClassLoader**@20f56ae6[server:0]
Local ClassPath: **/opt/IBM/WebSphere/Examples/VersionCheckerV2.jar**
Parent: com.ibm.ws.classloader.ProtectionClassLoader@18bdbd69
Delegation Mode: **PARENT_FIRST**

The new class loader that we defined is called *ExtJarClassLoader*, and it loads the VersionCheckerV2.jar file when requested by the EJB module.

The WAR class loader continues to load its own version due to the delegation mode.

22.5 OSGi class loaders

In the previous sections, we described how Java EE class loaders work. OSGi takes a completely different approach when it comes to class loaders, defining that each bundle has its own class path, as illustrated in Figure 22-13. This approach eliminates the problem with the class path parent delegation approach that we described in 22.1, “JVM class loaders” on page 790.

A series of imports and exports are defined when packaging an OSGi bundle. The imports are then used by the class loader to identify which classes to be loaded for the bundle. For more information about packaging OSGi, see 26.4, “Packaging OSGi applications” on page 932.

The tree organization of the Java EE class loading allows only going up in the tree for loading necessary classes. Thus, to make a library available for different branches of the tree, we must put it in a common ancestor of all the branches that will be using it, forcing all descendent children to use that same version of the library.

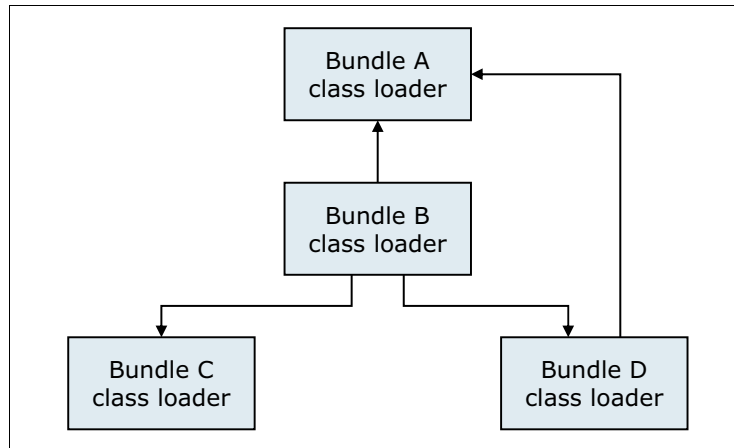


Figure 22-13 OSGi class loading

OSGi bundles allow you to change the standard Java hierarchical and strict class loading structure. Using the information provided in additional metadata properties, an OSGi environment wires a bundle only to declared packages and allows only those packages that are exported explicitly.

Figure 22-14 illustrates the difference between the traditional Java class loading process and the OSGi model. The OSGi model is similar to a network topology, while the standard Java is a static, one-way process.

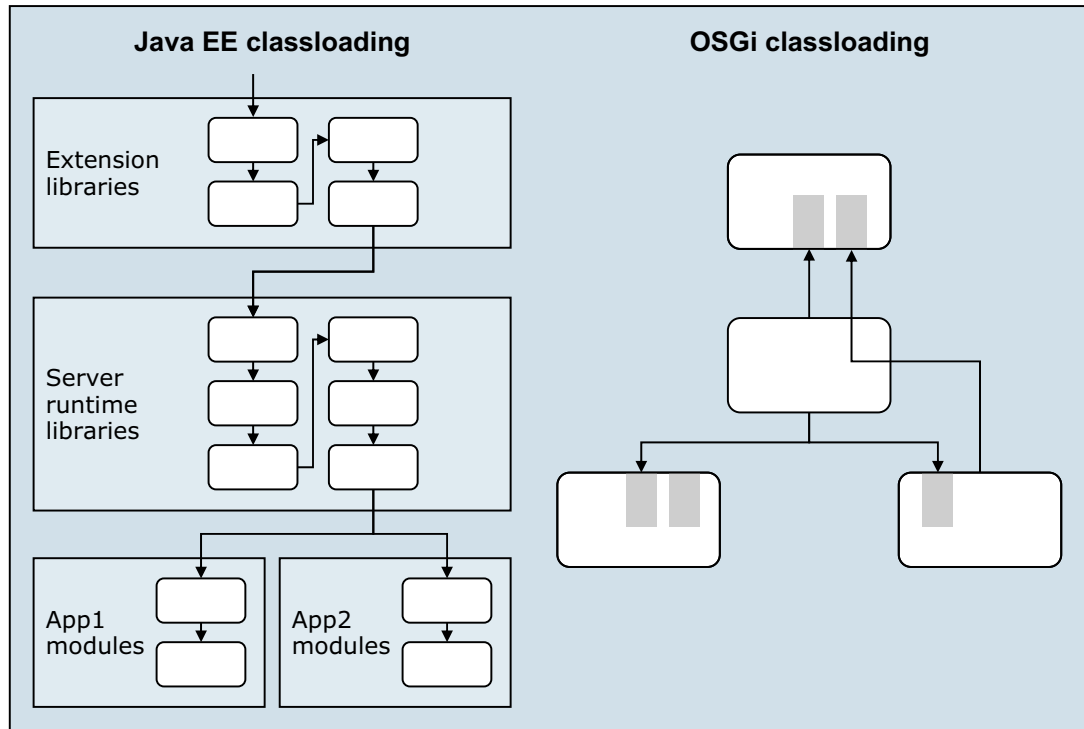


Figure 22-14 Java EE and OSGi class loading

Instead of a hierarchical class loading structure that is traditional in Java, particularly Java EE, OSGi has a network class loading structure, as shown in Figure 22-14. In the hierarchical model, classes are searched from the bottom through various layers of system classes, extension libraries, runtime libraries, and application modules. All classes are visible in the same layer and any lower layers. So, a new library in the extension libraries affects the entire stack as displayed.

In contrast to that approach, OSGi bundles are wired in a network. Furthermore, a bundle has visibility only of the individual packages for which it declares dependencies and no more.

Because by default everything is hidden, the same module can exist many times in such a network, and a single module can depend indirectly on the same module in many different versions. Furthermore, with the modularity metadata, which also allows package dependencies to be marked as optional, it is possible to check whether a bundle will work at run time or whether there are missing dependencies. So, with correctly written metadata, a bundle will never throw a `NoClassDefFoundError` exception at run time.

Note that this change of class loading structure is invasive. Certain assumptions that hold in Java EE no longer hold for OSGi bundles. For example, the thread context class loader is a commonly used mechanism in Java EE to access both application and server runtime classes. However, in OSGi, the thread context class loader contains only the classes that are visible to a single bundle. Consequently, some commonly used libraries are not fully OSGi-compatible even when repackaged.

More information about OSGi applications is in Chapter 26, “Working with OSGi applications” on page 921.



Packaging and deploying Java EE applications

In this chapter, we provide information about how to package and deploy Java Platform, Enterprise Edition 6 (Java EE 6) applications. First, we cover the packaging of the following Java EE artifacts:

- ▶ Enterprise applications
- ▶ EJB 3.1 modules
- ▶ Java Persistence API (JPA) persistence units
- ▶ Resource adapters
- ▶ Web modules
- ▶ EJB 3.1 content in Web modules

Then we provide information about how to deploy a Java EE file, both as an enterprise application and then as an asset in a business-level application. We explain how to set up the environment for the application and then deploy the application. Finally, we explain how to deploy the client part of the application. Note that you can automate the deployment tasks in this chapter using command-line tools, as explained in Chapter 8, “Administration with scripting” on page 319.

This chapter includes the following topics:

- ▶ Java EE applications
- ▶ Preparing to use the sample application
- ▶ Packaging recommendations
- ▶ Creating WebSphere-enhanced EAR files
- ▶ Exporting an application project to an EAR file
- ▶ Preparing the runtime environment for the application
- ▶ Deploying the application
- ▶ Deploying business-level applications
- ▶ Deploying application clients

23.1 Java EE applications

In this section, we describe the components of a Java EE application and how to work with each component.

23.1.1 Java EE 6 EAR files

WebSphere Application Server V8.5 supports the Java EE 6 and the EJB 3.1 specifications, which allow developers to use *annotations* in their source code. These annotations can contain information about how the application is to be deployed. Annotations can also reduce the number of classes and interfaces that the developer needs to manage within the project.

Java EE applications use *deployment descriptors*. A deployment descriptor is an extensible markup language (XML) file that specifies configuration and container options for an application or module. In Java EE 6 applications, the use of deployment descriptors is optional. If the code is correctly annotated, the EAR file or modules do not need to contain any deployment descriptors.

Java EE 6 relies on default values for the settings traditionally found in the deployment descriptors, and as long as the default values are acceptable, you do not need to include a deployment descriptor. However, if you include an optional deployment descriptor, the settings in the descriptor override the defaults and the settings given by the annotations in the source code. This method gives the deployer the flexibility to deploy the application as preferred for the target environment.

As with previous versions of the Java EE specification, Java EE 6 applications are packaged in enterprise archive (EAR) files. An EAR file can contain web archive (WAR) files, EJB modules (packaged as EJB JAR files), resource adapter archive (RAR) files, Java utility projects (packaged as JAR files), and application client modules. Figure 23-1 on page 815 shows a schematic overview of a Java EE EAR file.

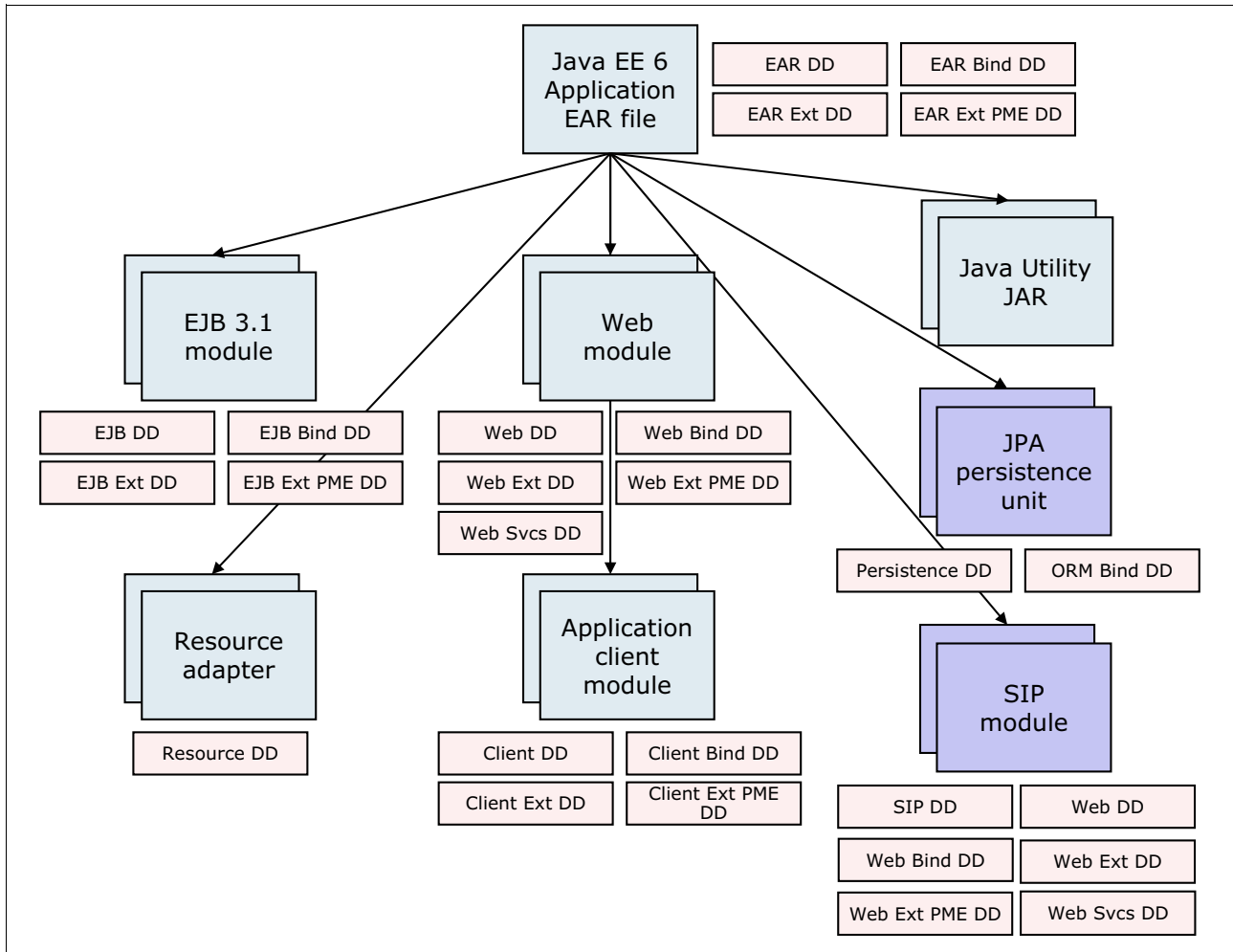


Figure 23-1 Java EE 6 EAR file structure

JPA: JPA is not part of JEE V6 specification. JPA 2.0 is defined in the JSR 317 specification. According to this specification, a JPA persistence unit can be packaged within one or more jar files contained within a WAR or EAR, as a set of classes within an EJB-JAR file or in the WAR classes directory, or as a combination of these.

SIP: SIP is not part of JEE V6 specification. SIP is defined by IETF and JCP organizations. If you want to know the SIP specifications that WebSphere Application Server V8.5 supports, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/a/rsip_refstandard.html

23.1.2 Development tools

You can use the following tools to develop enterprise applications for WebSphere Application Server:

- ▶ IBM Rational Application Developer for WebSphere Software

An integrated development environment and platform for building Java Platform Standard Edition (Java SE) and Java Platform Enterprise Edition (Java EE) applications with a focus on applications to be deployed to IBM WebSphere Application Server and IBM WebSphere Portal.

- ▶ IBM WebSphere Application Server Developer Tools for Eclipse

A new offering that provides plug-ins from the Eclipse Marketplace that can be installed into an existing Eclipse environment to support development for WebSphere Application Server.

- ▶ IBM Assembly and Deploy Tools for WebSphere Administration

This tool ships with the WebSphere Application Server license and is a subset of Rational Application Developer. You can use IBM Assembly and Deploy Tools for WebSphere Administration V8.5 to assemble and deploy applications that run on WebSphere Application Server V8.5. Using the extensible and easily customized workbench, you can quickly assemble and deploy Web, Java, Java EE, and OSGi applications. The tools are not licensed for application development purposes.

We use IBM Rational Application Developer for WebSphere Software V8.5 for the examples in this chapter. The other tools can also be used, but the procedures might change.

Figure 23-2 on page 817 is a comparison between the features included in IBM Rational Application Developer for WebSphere Software V9 and the WebSphere Application Server Developer Tools for Eclipse V8.5.5.

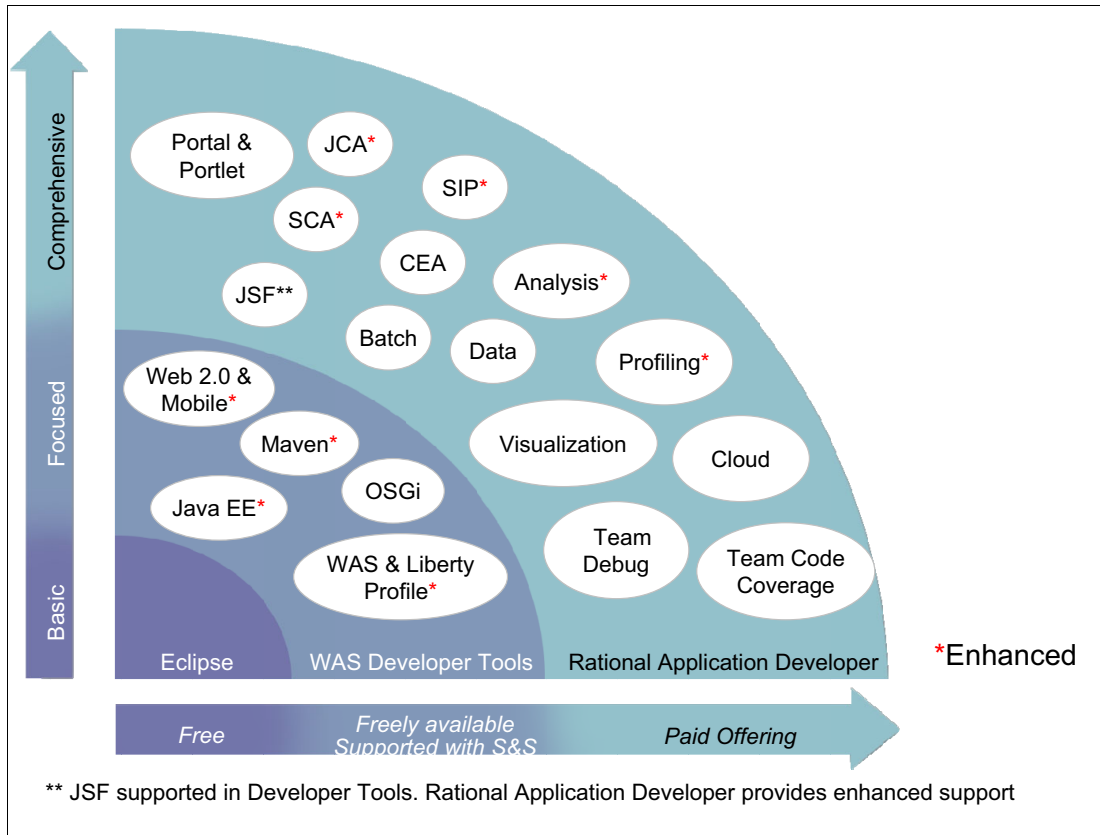


Figure 23-2 IBM Rational Application Developer and IBM WebSphere Application Server Developer Tools

23.1.3 Packaging enterprise applications

WebSphere Application Server V8.5 supports standard deployment descriptors and WebSphere specific deployment descriptors.

Table 23-1 lists the deployment descriptors that are valid for a WebSphere Application Server V8.5 EAR file.

Table 23-1 Enterprise archive deployment descriptors

| File name | Required | Content |
|-----------------------------|----------|---|
| application.xml | No | Defines modules and security roles that are used by the enterprise application. |
| ibm-application-bnd.xml | No | Includes mappings for security roles. |
| ibm-application-ext.xml | No | Defines WebSphere-specific application extensions. |
| ibm-application-ext-pme.xml | No | Includes configuration for WebSphere programming model extensions to the Java EE specification. |

Because the EAR deployment descriptors are no longer required, the current development tools do not generate them automatically. If you choose to include deployment descriptors, complete the following steps:

1. Start a developer tool. You can use IBM Rational Application Developer for WebSphere Software, IBM WebSphere Application Server Developer Tools, or IBM Assembly and Deploy Tools for WebSphere Administration.
2. Create a new Java EE application project or import an existing project. Complete the following actions to create or import projects:
 - To create a new project, click **File** → **New** → **Enterprise Application Project** and then follow the wizard instructions.
 - To import an existing project, click **File** → **Import** and then follow the wizard instructions.
3. In the Enterprise Explorer view, right-click the Java EE project, and click **Java EE** → **Generate Deployment Descriptor Stub**.
4. Expand the Java EE projects META-INF folder to reveal the created application.xml deployment descriptor file. To edit it, either double-click the file or double-click the EAR file's deployment descriptor icon, as shown in Figure 23-3.

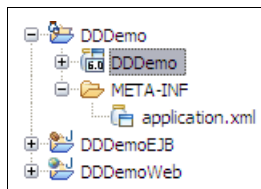


Figure 23-3 JEE EAR deployment descriptor icon in Rational Application Developer-AD

5. On the right side of the deployment descriptor editor is a pane with fields for the information that can be entered into the deployment descriptor. Refer to Figure 23-4 on page 819. Complete the fields and then press Ctrl+s to save the deployment descriptor.

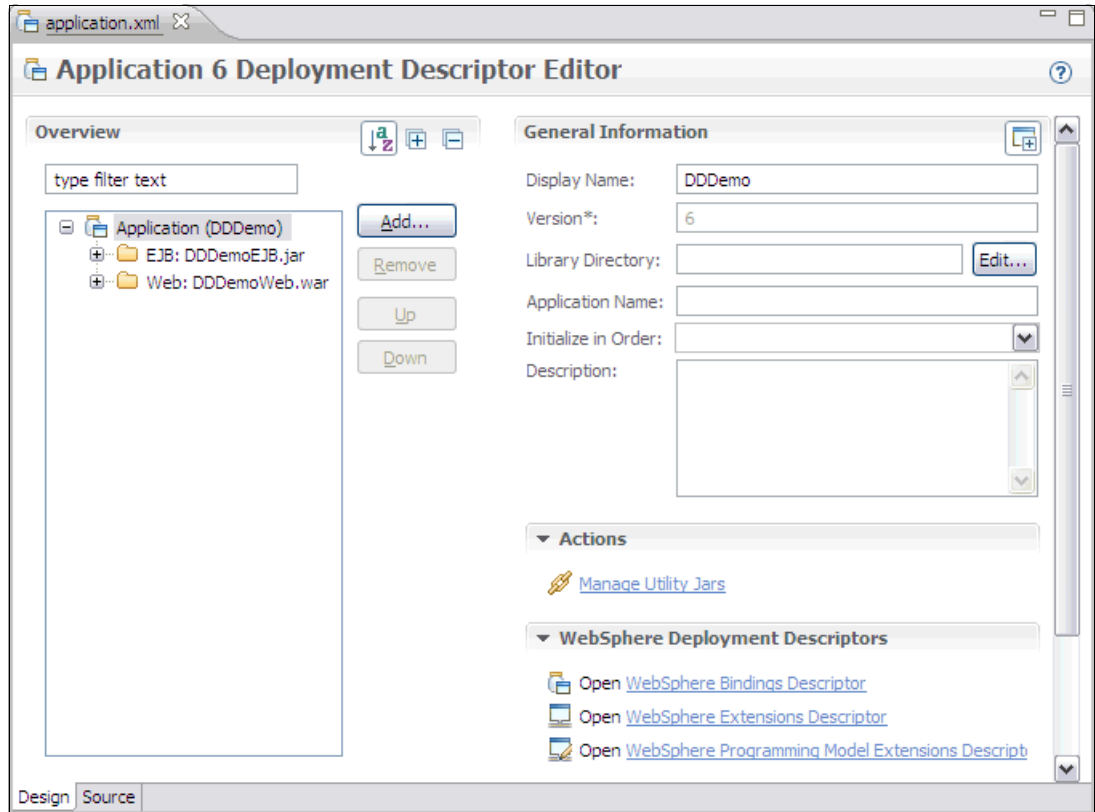


Figure 23-4 Java EE EAR deployment descriptor editor

6. To create other optional Java EE EAR-file level deployment descriptors, click the corresponding link under the WebSphere Deployment Descriptors heading shown in Figure 23-4. The options are noted in the following list:
 - WebSphere Bindings Descriptor
 - WebSphere Extensions Descriptor
 - WebSphere Programming Model Extensions Descriptor

Selecting any of these options creates the corresponding deployment descriptor file. If it does not exist, in the Java EE EAR file's META-INF folder, as shown in Figure 23-5, the file can then be accessed two other ways. You can either double-click it in the META-INF folder or click the corresponding link under the WebSphere Deployment Descriptors heading on the main Java EE EAR deployment descriptor editor.

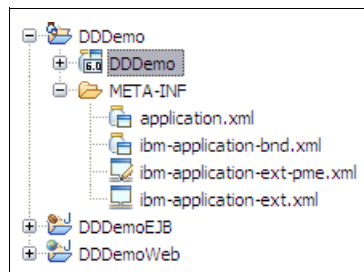


Figure 23-5 WebSphere deployment descriptors

23.1.4 Packaging EJB 3.1 modules

The EJB 3.1 and EJB 3.0 specifications are similar. However, EJB 3.1 content can be packaged and deployed as a part of a WAR file. In 23.1.9, “Packaging EJB 3.1 content in Web modules” on page 829, we show an example of how to package and export a WAR file that has EJB 3.1 content.

EJB 3.1 modules can also be packaged without a deployment descriptor. To package a module without a deployment descriptor, you must create a JAR file or WAR file with metadata in annotations located in the EJB component.

The EJB 3.0 or later specification does not support the use of container-managed persistence (CMP) or bean-managed persistence (BMP). JPA is used for data persistence. The EJB 2 and EJB 2.1 specifications continue to support CMP and BMP as per the J2EE specifications. WebSphere Application Server V8.5 also supports CMP and BMP. Table 23-2 lists the deployment descriptors for an EJB 3 or later module.

Table 23-2 EJB 3.1 deployment descriptors

| File name | Required | Content |
|---------------------|----------|--|
| ejb-jar.xml | No | EJB and EJB method definitions, transaction attributes, resource references, and so on |
| ibm-ejb-jar-bnd.xml | No | Explicit binding names for EJB and resource references |
| ibm-ejb-jar-ext.xml | No | Configuration of WebSphere extensions to the Java EE EJB module specification |
| ibm-web-ext-pme.xml | No | Configuration for WebSphere programming model extensions to the Java EE specification |

EJB interface bindings

WebSphere Application Server V8.5 binds EJB 3.1 interfaces and homes into two distinct JNDI namespaces, one JVM-local and one global namespace. Local interfaces and homes are bound to the JVM-local namespace, and remote interfaces and homes are bound to the global namespace.

Unless overridden by explicitly assigned bindings, the interfaces are bound using default names that are generated automatically by the EJB container. Each default name has a short version and a long version. The short name consists of only the Java package name and class name of the interface. The long name prefixes the short name with a component ID, which is composed of the enterprise application name, the module name, and the component name.

The patterns for each type of default binding are displayed in Table 23-3. In these patterns, strings written in *<bracketed italics>* represent a value. For example, *<package.qualified.interface>* might be `com.mycompany.AccountService`, and *<component-id>* might be `AccountApp/module1.jar/ServiceBean`.

Table 23-3 Default binding patterns

| Binding patterns | Description |
|--|--|
| ejblocal: <i><package.qualified.interface></i> | Short form local interfaces, homes, and no-interface views |
| <i><package.qualified.interface></i> | Short form remote interfaces and homes |

| Binding patterns | Description |
|--|---|
| <code>ejblocal:<component-id>#<package.qualified.interface></code> | Long form local interfaces, homes, and no-interface views |
| <code>ejb/<component-id>#<package.qualified.interface></code> | Long form remote interfaces and homes |

The auto-generated default names can be overridden by placing a file named `ibm-ejb-jar-bnd.xml` in the EJB JAR module's `META-INF` directory with the preferred names. By overriding the default names, you can define your own naming convention independently from how the beans are packaged into the application or module hierarchy.

java:[scope] namespaces

The `java:global`, `java:app`, and `java:module` namespaces are introduced by the Java EE 6 specification. They provide a mechanism for binding and looking up resources that are portable across application servers.

The server always creates a default long-form binding for each EJB interface, including the no-interface view, and places them into the `java:global`, `java:app`, and `java:module` namespaces. If the bean exposes only one interface, including the no-interface view, a short-form binding is also created and placed into the `java:global`, `java:app`, and `java:module` namespaces. The default bindings are only created for session beans. They are not created for entity beans or message driven beans. For example, the bean component `MyBeanComponent` exposes just the one `com.foo.MyBeanComponentLocalInterface` interface, and is packaged in the `myModule.jar` module in the `myApp.ear` file. As a result, the following bindings are created in the `java:[scope]` namespaces:

- ▶ `java:global/myApp/myModule/MyBeanComponent!com.foo.MyBeanComponentLocalInterface`
- ▶ `java:global/myApp/myModule/MyBeanComponent`
- ▶ `java:app/myModule/MyBeanComponent!com.foo.MyBeanComponentLocalInterface`
- ▶ `java:app/myModule/MyBeanComponent`
- ▶ `java:module/MyBeanComponent!com.foo.MyBeanComponentLocalInterface`
- ▶ `java:module/MyBeanComponent`

The EJBLink and AutoLink features

When an EJB client (typically a servlet, or another EJB) wants to call an EJB, it first needs to locate the EJB home in the JNDI namespace. In EJB 2.1 and earlier, this call was completed with a few lines of code written explicitly by the EJB client developer. However, with the EJB 3 or later support and source code annotations, WebSphere Application Server V8.5 supports two different mechanisms that resolve references:

- ▶ EJBLink feature, defined by the EJB specification
- ▶ AutoLink feature that is a WebSphere Application Server extension

When the EJB container encounters an annotation for an EJB reference, it tries to look up the referenced EJB automatically. The EJBLink and AutoLink features use different search criteria to locate the targeted bean component. EJBLink searches for the targeted bean component using the explicitly specified bean name. AutoLink searches for the targeted bean component using the interface that the bean implements. If no explicit bindings are provided, but a bean name is provided, the EJBLink feature is used. If no explicit bindings are provided, and no bean name is provided, the AutoLink feature is used. The EJBLink and AutoLink features are never used together as part of the same search process

The scope of EJBLink and AutoLink is limited to the enterprise application in which the EJB reference appears and within the application server on which the referring module is assigned. If the target EJB resides in an application other than the client's or if it is deployed on an application server other than the client's, EJBLink or AutoLink does not work. In this

case, target bindings must be defined explicitly in the client's bindings file. For an EJB module, this bindings file is the `ibm-ejb-jar.bnd.xml` file, and for a web module, it is the `ibm-web-bnd.xmi` file.

The AutoLink feature handles only EJB references and are available for clients running in the EJB container, web container, or application client container. For more information about EJBLink or AutoLink, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.zseries.doc/ae/cejb_bindingsejbf.html

Just-in-time generation of EJB deployed code

EJB modules must contain EJB deployed code in order for an application server to be able to run the EJB. EJB-deployed code contains application-server specific code that bridges the EJB interface and implementation code to the application server's EJB implementation.

In previous versions of WebSphere Application Server and with previous versions of Java EE, the EJB deployed code was generated using one of the following methods:

- ▶ During development, using the Prepare for Deploy action in Rational Application Developer or the WebSphere Application Server Toolkit
- ▶ Before installing an EAR file to WebSphere Application Server using the EJBDeploy tool from a command line
- ▶ During installation of an EAR file to WebSphere Application Server using the installation windows in the administrative console

WebSphere Application Server V8.5 and EJB 3.x support a feature called *just-in-time* (JIT) deployment. This feature removes the need to process the EJB modules to generate the deployed code. Instead, the EJB container dynamically generates the necessary code in-memory as needed when the application is running. This feature simplifies and speeds up the development, packaging, and deployment of EJB.

For EJB 3 or later, clients that are not running inside a web container, EJB container, or client container that was upgraded to the EJB 3 or later level, the JIT development does not generate the necessary classes. In this case, use the `createEJBStubs` tool to make the generated classes available on the client's class path. An example of this situation is a servlet running in WebSphere Application Server V6.1 calling an EJB 3 or later bean running in WebSphere Application Server V8.5. In this case, the EJB stubs are created manually, and the generated classes are added to the servlet's web module.

For details about the `createEJBStubs` tool and the syntax that it uses, refer to the WebSphere Application Server V8.5 Information Center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.base.doc/ae/rejb_3stubscmd.html

Mixing different EJB versions in an EAR file

WebSphere Application Server V8.5 allows you to mix EJB 1.1, 2 or later, and 3 or later beans in the same Java EE 6 enterprise application. However, EJB 3.x beans are not recognized in EJB 2.x or EJB 1.x modules.

Note that the EJB 3.1 specification does not support the use of BMP and CMP entity beans in EJB 3 or later modules. EJB entity beans can be used on Version 8.5, but they must be packaged in an EJB 2.1 or earlier module. This method gives you time to migrate to the newest specification while you use your existing modules.

23.1.5 Packaging JPA persistence units

Persistence units using JPA can be packaged either into the module that uses the persistence unit or in a separate utility JAR file (packaged as a standard .jar file). If packaged as a separate utility JAR file, it must be referenced from the module that uses the persistence unit using the module's META-INF/MANIFEST.MF class path directive.

Persistence units require a persistence.xml file, which defines a JPA entity manager's configuration. Among other information, the persistence.xml file lists the entity classes and the data source to use.

A persistence unit can also include an optional orm.xml file that specifies the object-relational mapping configuration. The orm.xml file is an alternative to using annotations and can be used to override annotations in the source code to specify how the objects are persisted to the database.

Table 23-4 lists the deployment descriptors that are valid for a JPA persistence unit.

Table 23-4 JPA persistence units deployment descriptors

| File name | Required | Content |
|-----------------|----------|---|
| persistence.xml | Yes | Entity manager's configuration, entity classes, data sources, and so on |
| orm.xml | No | Object-to-relational mapping annotation overrides |

23.1.6 JPA access intent

WebSphere Application Server provides an optimization enhancement for EJB 2 or later entity beans called *access intent*. However, because the EJB 3 or later specifications do not support entity beans, access intent support is not available for EJB 3 or later beans. Instead, WebSphere Application Server V8.5 provides JPA access intent that can be used to improve performance and scalability for JPA applications.

JPA access intent specify the isolation and lock levels that are used when reading data from a data source. JPA access intents can be used, providing that the following restrictions are honored:

- ▶ Access intent is available for the application in the Java EE server environment.
- ▶ Access intent is applicable to non-query entity manager interface methods. Query uses a query hint interface to set its isolation and read lock values.
- ▶ Access intent is only available for DB2 databases.
- ▶ Access intent is in effect only when pessimistic lock manager is used. To specify a pessimistic lock manager, add the following statement to the persistence unit's property list:

```
<property name="openjpa.LockManager" value="pessimistic"/>.
```

Table 23-5 on page 824 compares EJB 2 or later access intent to JPA access intent.

Table 23-5 JPA access intent properties

| EJB 2 entity bean access intent | JPA access intent | Description |
|---------------------------------|---------------------------------|--|
| optimistic | isolation: Read Committed | Data is read but no lock is held. Version ID is used on update to ensure data integrity. Other transactions can read and update data. |
| | lockManager: Optimistic | |
| | query Hint: ReadLockMode: READ | |
| pessimistic read | isolation: Repeatable Read | Data is read with shared locks. Other transactions attempting to update data are blocked. |
| | lockManager: Optimistic | |
| | query Hint: ReadLockMode: READ | |
| pessimistic update | isolation: Repeatable Read | Data is retrieved with update or exclusive lock. Other writes are blocked until commit. This access intent can be used to serialize update access to data when there are multiple writers. |
| | lockManager: Pessimistic | |
| | query Hint: ReadLockMode: WRITE | |
| pessimistic exclusive | isolation: Serializable | Data is retrieved with update or exclusive lock. Other writes are blocked until commit. This access intent can be used to serialize update access to data when there are multiple writers. |
| | lockManager: Pessimistic | |
| | query Hint: ReadLockMode:WRITE | |

For more information about JPA access intent, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/t_ejb_accessintentjpa.html

For information about EJB 2 or later access intent, refer to *WebSphere Application Server V6.1: System Management and Configuration*, SG24-7304.

23.1.7 Packaging resource adapters

A resource adapter archive (RAR) module, also called a *connector module*, contains code that implements a library for connecting with a back-end enterprise information system (EIS), such as CICS, SAP, and PeopleSoft. RAR files (called *connectors*) are packaged as a standard Java archive with a .rar file extension. A resource adapter can be installed as a stand-alone adapter or as part of an enterprise application, in which case the resource adapter is referred to as an *embedded* adapter.

A connector module contains a mandatory deployment descriptor file named ra.xml that resides in the module's META-INF directory. For more information about resource adapters refer to Chapter 10, "Accessing EIS applications from WebSphere" on page 383.

23.1.8 Packaging Web modules

Java EE 6 web modules are packaged just like web modules in earlier Java EE versions. A web module can contain servlet code, JSPs, static HTML pages, images, JavaScript, style sheets, and so on.

A common challenge when working with web modules is to make sure that the correct version of a required Java library is loaded. Often web application developers need to include specific third-party libraries, such as log4j or Xalan/Xerces, and must make sure that the correct version of a library is loaded for an application. This requires knowledge about how the EAR and web module's class loaders work. Refer to *Chapter 22, "Understanding class loaders"* on page 789 for detailed information about this topic.

A web module supports several deployment descriptors, as shown in Table 23-6.

Table 23-6 Web module deployment descriptors

| File name | Required | Purpose |
|---------------------|----------|---|
| web.xml | No | Servlet definitions, URL mappings, and init parameters, servlet listeners, and so on |
| ibm-web-bnd.xml | No | Mapping of logical resources that are used by the web module to their runtime managed resources |
| ibm-web-ext.xml | No | Configuration of WebSphere extensions to the Java EE web module specification |
| ibm-web-ext-pme.xml | No | Configuration for WebSphere programming model extensions to the Java EE specification |
| webservices.xml | No | Configuration of web services and implementation code |

Because the web deployment descriptors are no longer required, the current development tools do not generate them automatically. To create the web deployment descriptor in Rational Application Developer, right-click the web module in the Enterprise Explorer view and select **Generate Deployment Descriptor Stub**.

Deployment descriptor note: If an application.xml deployment descriptor is not included in the EAR file, the context root for a web module defaults to the web module's name without the .war extension.

WebSphere extensions to web modules

WebSphere Application Server provides multiple extensions for web modules. These extensions are configured in the `ibm-web-ext.xml` deployment descriptor in the web module. To create this file in Rational Application Developer, right-click the web module in the Enterprise Explorer view, and click **Java EE** → **Generate WebSphere Extensions Deployment Descriptor**. This works if the project has the **WebSphere Web (Extended) 8.5** facet. To edit the file, use either of the following methods:

- ▶ Expand the web module's WebContent/WEB-INF folder, and then double-click the **ibm-web-ext.xml** file.
- ▶ Click the **Open WebSphere Extensions Descriptor** link under the **WebSphere Deployment Descriptors** heading on the web module's deployment descriptor editor for the web.xml file, as shown in Figure 23-6 on page 826.

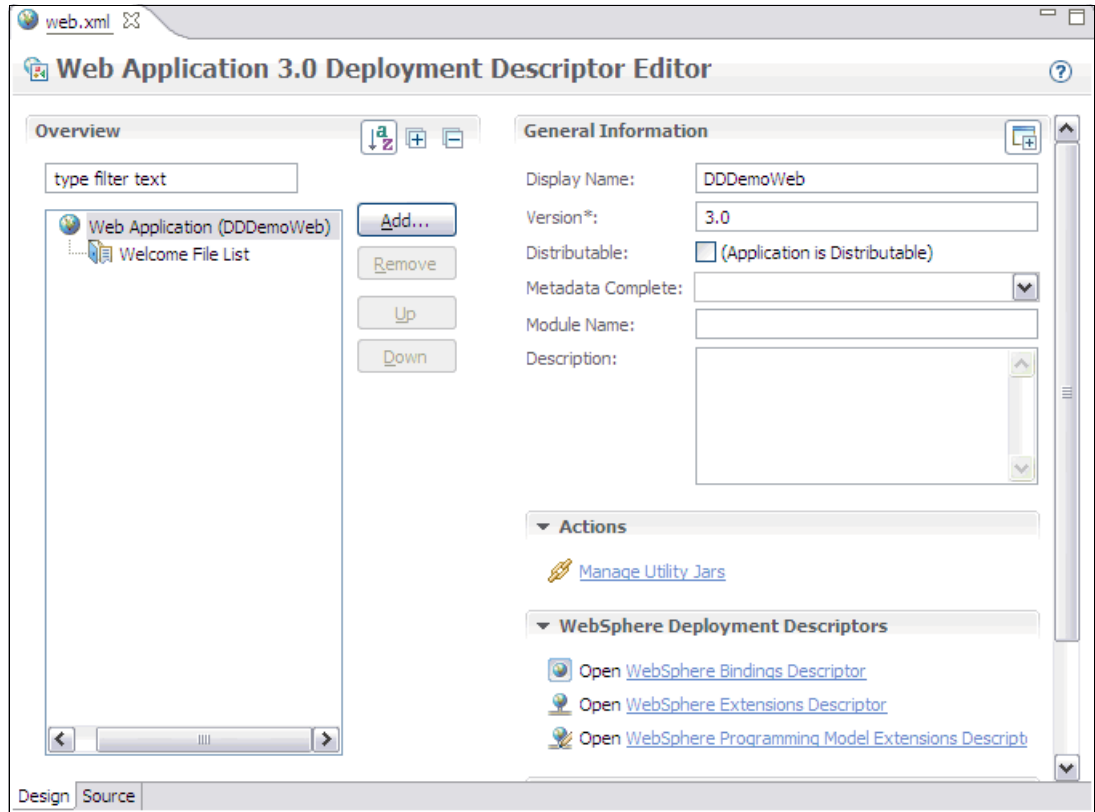


Figure 23-6 Web module's deployment descriptor editor

The Web module extensions editor is shown in Figure 23-7 on page 827.

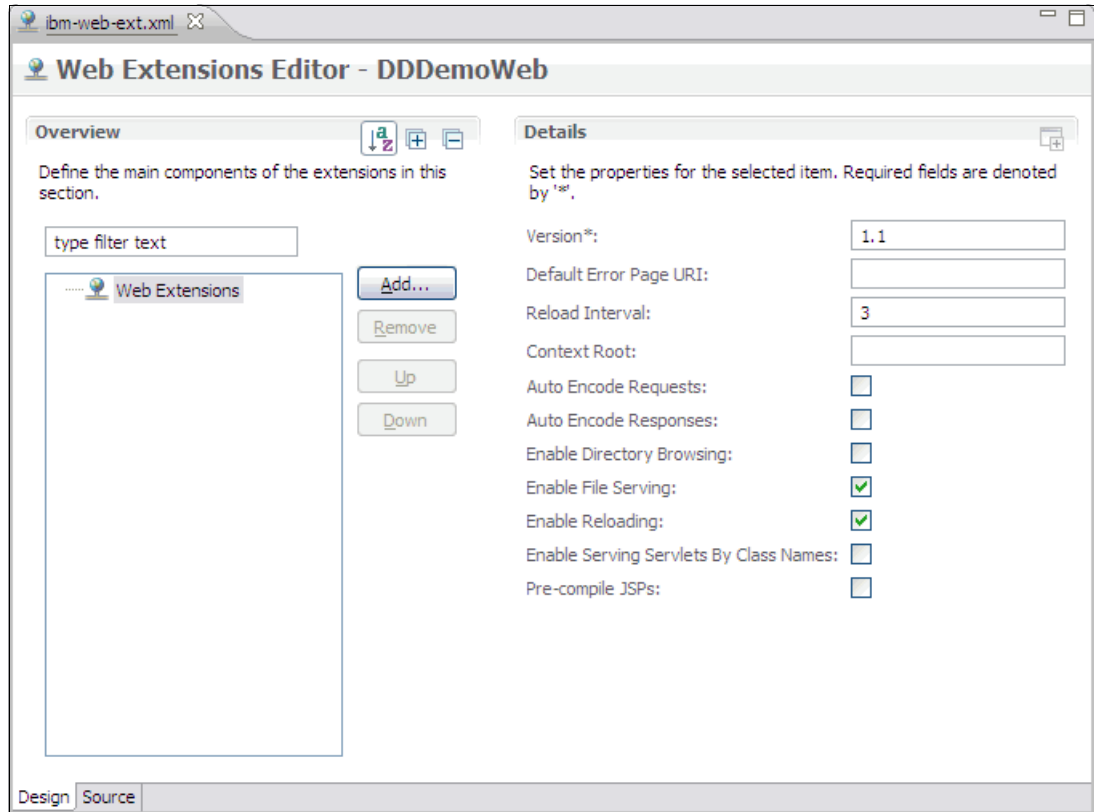


Figure 23-7 WebSphere Web module extensions editor

The following sections provide information about the options in the web module extensions.

File serving

When dealing with static content (HTML pages, images, style sheets, and so on), you can choose to have these resources served by WebSphere or have them served by the HTTP server itself.

If you want WebSphere to serve the static content of your application, you must enable file serving in the web module extensions deployment descriptor (Figure 23-7). Enabling this feature activates a servlet that serves up any resource file that is packaged in the WAR file. The file serving enabled attribute is set to true by default. By changing it to false, the web server plug-in does not send requests for static content to WebSphere but leaves it up to the HTTP server to serve them.

To enable this option, select the **Enable File Serving** option.

Tip: You can experience better performance serving static content from the web server than using WebSphere to serve the static content of your application because the web server serves the content directly. Moreover, a web server has more customization options than the file servlet can offer.

However, using the WebSphere file serving servlet has the advantage of keeping the static content organized in a single, deployable unit with the rest of the application. Additionally, it allows you to protect the static pages using WebSphere security.

Web application auto reload

If you select the **Enable Reloading** option in the web module extensions, the class path of the web application is monitored and all components, JAR files, or class files are reloaded when a component update is detected. The web module's class loader is shutdown and restarted. The Reload Interval is the interval between reloads of the web application. It is set in seconds.

The automatic reload feature plays a critical role in hot deployment and dynamic reload of your application.

Important: You must set the Enable Reloading enabled option to **true** for JSP files to be reloaded when they are changed on the file system. Reloading a JSP does not trigger the reload of the web module because separate class loaders are used for servlets and JSP.

This option is enabled by default with the reload interval set to three seconds. Thus, the classloader checks the classes on the class path for updates every three seconds. If any changes are found, those classes are reloaded. If no changes are detected, nothing happens to the classloader or the classes that are loaded. In production mode, you might consider turning this feature off or making the reload interval much higher.

Serving servlets by class name

You can use the invoker servlet to invoke servlets by class name. Note that there is a potential security risk with leaving this option set in production. Use it more as a development-time feature for testing servlets quickly.

A better alternative than this option is to define servlet mappings in the web deployment descriptor for the servlets that must be available.

You configure the invoker servlet using the Enable Serving Servlets By Class Names option.

Default error page

This page is invoked to handle errors if no error page is defined or if none of the defined error pages matches the current error.

Directory browsing

This option defines whether it is possible to browse the directory if no default page is found.

Turn off this option for improved security.

Pre-compile JSPs

When a JSP is hit for the first time, it is compiled automatically into a servlet and then executed. To avoid this performance penalty the first time a JSP is hit, WebSphere allows JSPs to be pre-compiled during application installation, instead of at first invocation. Selecting this option causes the installation of the application to WebSphere to take longer, but the JSPs are served faster on the first hit. You can enable this feature from the deployment descriptor, as shown on Figure 23-7 on page 827.

Automatic HTTP request and response encoding

The web container no longer automatically sets request and response encodings and response content types. The programmer is expected to set these values using the methods available in the Servlet 3.0 API. If you want the application server to attempt to set these values automatically, select the **Auto Encode Requests** option to have the request encoding

value set. Similarly, you can select the **Auto Encode Responses** option to have the response encoding and content type set.

The default value of the `autoRequestEncoding` and `autoResponseEncoding` extensions is `false`, which means that both the request and response character encoding is set to the Servlet 3.0 specification default of ISO-8859-1. Different character encodings are possible if the client defines character encoding in the request header or if the code uses the `setCharacterEncoding(String encoding)` method.

The web container tries to determine the correct character encoding for the request parameter and data in any of the following cases:

- ▶ If the `autoRequestEncoding` value is set to `true`
- ▶ If the client did not specify character encoding in the request header
- ▶ If the code does not include the `setCharacterEncoding(String encoding)` method

The web container performs each step in the following list until a match is found:

1. Looks at the character set (`charset`) in the Content-Type header.
2. Attempts to map the server's locale to a character set using defined properties.
3. Attempts to use the `DEFAULT_CLIENT_ENCODING` system property, if one is set.
4. If a match is not found, uses the ISO-8859-1 character encoding as the default.

If you set the `autoResponseEncoding` value to `true` and the following conditions are also true:

- ▶ The client did not specify character encoding in the request header.
- ▶ The code does not include the `setCharacterEncoding(String encoding)` method.

Next, the web container performs the following actions:

1. Attempts to determine the response content type and character encoding from information in the request header.
2. If it fails, uses the ISO-8859-1 character encoding as the default.

23.1.9 Packaging EJB 3.1 content in Web modules

In Java EE 6 applications, it is possible to package EJB content in Web modules. A bean that is packaged inside a Web module has the same behavior as a bean that is packaged inside an EJB JAR module, but the rules for packaging vary, depending on the type of module that is being used.

The bean class files must be placed in one of two locations within the WAR file:

- ▶ The `WEB-INF/classes` directory
- ▶ Within a JAR file that is placed in the `WEB-INF/lib` directory

In the first approach, the Java file is developed as a part of the same Web project. When the class is compiled, it is placed in the `WEB-INF/classes` directory.

The second approach imports the JAR file that contains the EJB contents. This JAR file can also be the result of the packaging of other projects.

If the same class is placed in both the `WEB-INF/classes` and `WEB-INF/lib` directories, the instance of the class placed in the `WEB-INF/classes` directory is loaded, and the instance placed in a JAR file in the `WEB-INF/lib` directory is ignored.

If the same bean class is placed in two different JAR files in the `WEB-INF/lib` directory, the server arbitrarily picks one class instance and loads it, ignoring the other class. This method can result in erratic behavior.

If you need to use deployment descriptors, you must place them in the WEB-INF directory.

23.2 Preparing to use the sample application

As an example of how to package and deploy a Java EE 6 application using EJB 3.1, we use the ITSO Bank application that was developed for *Rational Application Developer for WebSphere Software V8 Programming Guide*, SG24-7835. The structure of this application, as seen in the Enterprise Explorer view in Rational Application Developer, is shown in Figure 23-8.

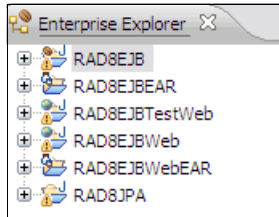


Figure 23-8 Imported ITSO Bank application

When the application is imported, the workspace has two enterprise archive (EAR) projects, called *RAD8EJB* and *RAD8EJBWebEAR*. *RAD8EJB* contains EJBs and a simple servlet for testing (in the *RAD8EJBTestWeb* project). A more sophisticated web application is available in the *RAD8EJBWeb* project. We use this application in our example.

The *RAD8EJBWeb* project uses the EJB in the *RAD8EJB* project, which in turn relies on the Persistence Unit in the *RAD8JPA* project.

This section provides the information needed to import the sample application into Rational Application Developer for WebSphere Software. If you are not planning to download the sample application, you can skip to 23.2.5, “Configuring web module extensions” on page 833.

23.2.1 Downloading the application

To download the sample application:

1. Go to the following website:
<http://www.redbooks.ibm.com/abstracts/sg247835.html?open>
2. Click the **Additional Material** link.
3. Click the **sg247835.zip** file, and select **Save** to save the compressed file to your computer.
4. Extract the contents of the compressed file. You will have two directories called *7835code* and *7835codesolution*.

23.2.2 Importing the application to the development tool

If you navigate to the *7835codesolution* directory, you notice a number of directories. For our discussion in this section, we use the *ejb* directory. This directory contains two compressed files that include Rational Application Developer project interchange files:

- ▶ *RAD8EJB.zip*
- ▶ *RAD8EJBWeb.zip*

To use the sample application for our exercise, import both files into Rational Application Developer for WebSphere Software by completing the following steps:

1. Start Rational Application Developer for WebSphere Software.
2. To import the code, click **File** → **Import** and then expand the **General** section. Select **Existing projects into workspace**. Click **Next**.
3. Click the **Select archive file** option. Click **Browse** next to the Select Archive file field, and browse to the `ejb` directory where you extracted the sample code. Select the **RAD8EJB.zip** file, and click **Open**.
4. Click **Select All** to select all projects in the file. Click **Finish**.
5. You might see a pop-up window during the import process that says the project needs to be migrated. If so, click **Next** until the migration completes.
6. Repeat the process for the `RAD8EJBWeb.zip` project.

If you see errors in the Markers view:

- ▶ Make sure you have the WebSphere Application Server V8.5 runtime environment installed and an application server defined when you import the application.
- ▶ If you have a server defined, but see errors that indicate a build path error, right-click each module where the error occurs, and click **Properties** → **Java Build Path**. Click the **Libraries** tab. Click **JRE System Library (WebSphere Application Server V8.5 JRE)** and then click **Edit**. Select **Alternate JRE** and then in the drop-down menu, select **WebSphere Application Server JRE**. Click **Finish** and then **OK**.
- ▶ If you have errors in the `persistence.xml` file, select **RAD8JPA/src/META-INF/persistence.xml** in the Enterprise Explorer view. Right-click and select **JPA Tools** → **Synchronize Class List**.
- ▶ If you see an error in the `ejb-jar.xml` file of the `RAD8EJB` project, you can ignore it. That file will be deleted in the next section.

After you finish to import the projects, make sure the Enterprise Explorer view looks like Figure 23-8 on page 830.

23.2.3 Customizing the sample application

You can use the ITSO Bank application without customizing it. The team that created the application did all of the necessary development for the application to work. However, to illustrate common packaging tasks, customize the application for this example by completing the following steps:

1. Remove the unnecessary deployment descriptors that were included in the application by the development team as follows:
 - a. Expand the `RAD8EJB` project, and expand the `ejbModule` folder. Expand the `META-INF` folder, and delete the `ejb-jar.xml` file.
 - b. Expand the `RAD8EJBWebEAR` project, and expand its `META-INF` folder. Delete the `application.xml` file.
 - c. Expand the `RAD8JPA` project, and expand the `src` folder. Expand the `META-INF` folder, and delete the `orm.xml` file.
2. The `RAD8EJB` project depends on the Persistence Unit defined in the `RAD8JPA` project. To verify that this dependency is set up correctly, use the Deployment Assembly properties sheet. The integrated development environment (IDE) updates the `application.xml` file automatically, if one exists.

To access the Deployment Assembly properties sheet, right-click the **RAD8EJBEAR** project, and select **Properties**. Click **Deployment Assembly** in the left pane.

3. In the EAR Module Assembly window, the list of projects included in the EAR must look like Figure 23-9. All three projects, including the RAD8JPA.jar project, are in the list.

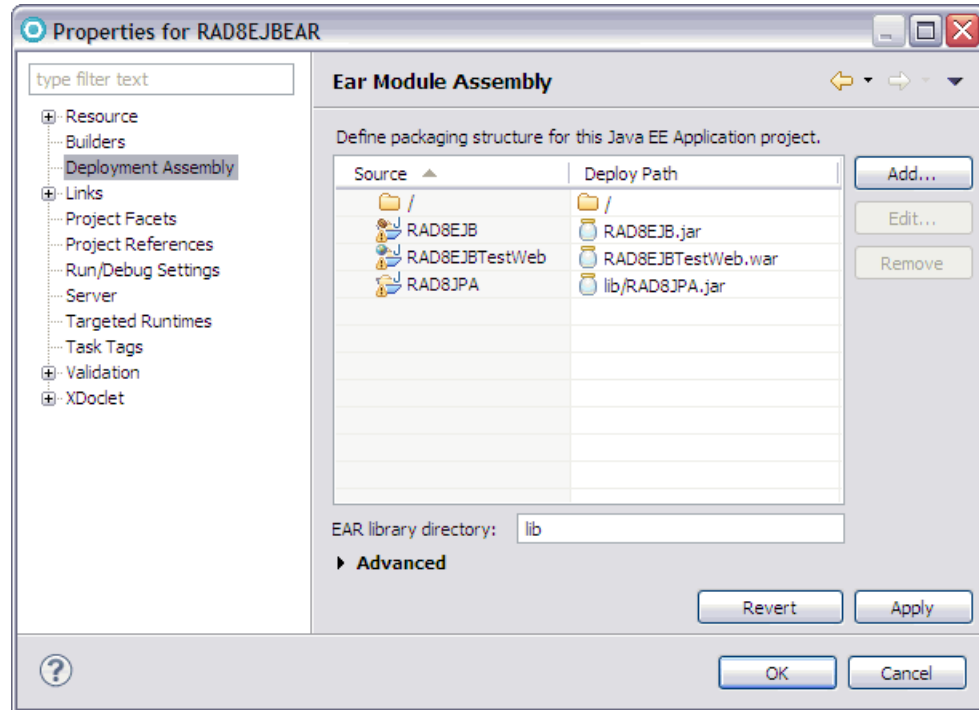


Figure 23-9 ITSOBank application deployment assembly configuration

If all three packages shown in Figure 23-9 are not listed in your properties:

- a. Click **Add**.
- b. In the New Assembly directive wizard, locate the Select Directive Type window, click **Project**, and click **Next**.
- c. In the New Assembly directive wizard, in the Project window, select the missing module, for example **RAD8JPA**, and click **Finish**.

Otherwise, click **Cancel**.

23.2.4 Creating the ITSO Bank DB2 database

If you plan to deploy and test this application as described in this chapter, you need to create the database on a DB2 system.

In the extracted files for the sample application, locate the 7835code folder. The database directory in this compressed file contains scripts that we use to prepare the database for the application.

To set up the DB2 database, make sure that DB2 is installed and running and then complete the following steps:

1. Open a command prompt.
2. Change to the 7835code/database/db2 folder in the extracted files for the sample application.

3. Execute the createbank.bat file to define the database and table.
4. Execute the loadbank.bat file to delete the existing data and add records.
5. Execute the listbank.bat file to list the contents of the database.

Each command opens a new window where the DB2 script executes. Each command also leaves a connection to the database open, so you might want to execute a **db2 connect reset** command in each window opened to disconnect from the database so no unused connections are kept open. The database is now configured for the ITSO Bank application.

23.2.5 Configuring web module extensions

To prepare an application for deployment, you want to review and possibly customize the WebSphere web module extensions. In this example, the serve servlets, by class name and directory browsing options, are disabled. It is a best practice to disable these options in production environments so that only the servlets and folders that the developers intended to be accessible are accessible.

To configure the webs module extensions, complete the following steps.

1. Expand the RAD8EJBWeb project, and double-click the **RAD8EJBWeb** heading to open the web module deployment descriptor.
2. When the window opens, click the **Open WebSphere Extensions Descriptor** link in the bottom right corner, as shown in Figure 23-10, to open the extensions editor.

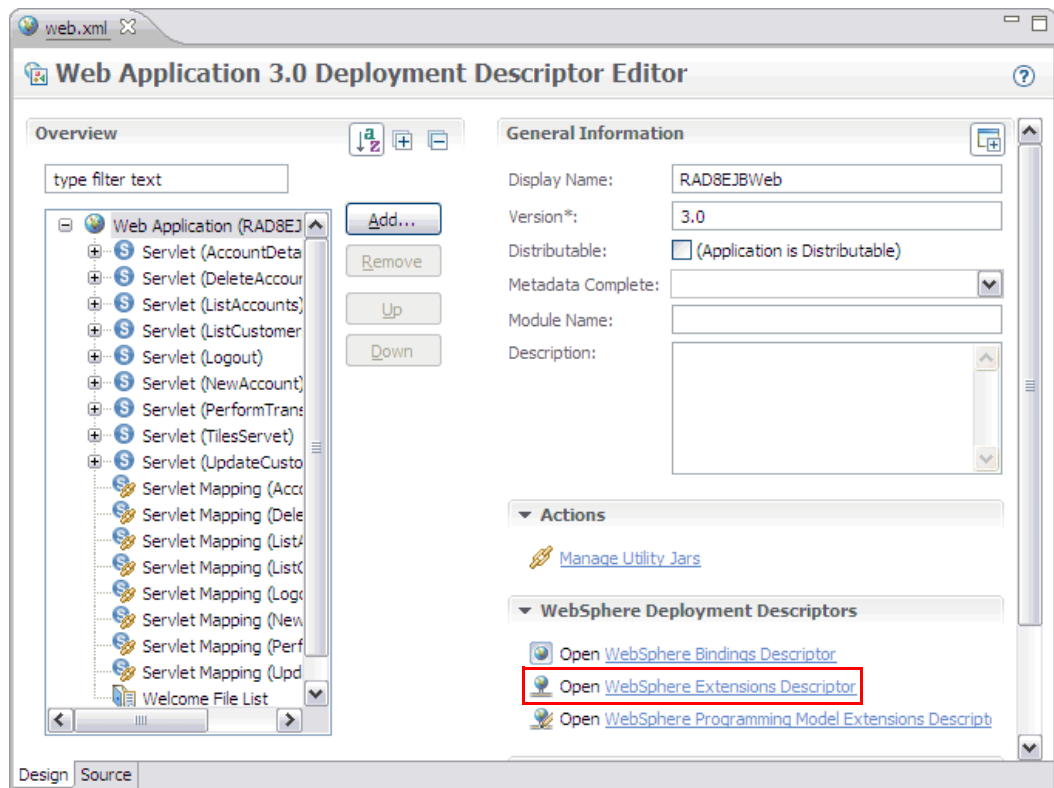


Figure 23-10 Web module deployment descriptor editor

3. The web module extensions editor contains options to configure the optional WebSphere extensions to web modules. In our application, we clear the Enable Directory Browsing and Enable Serving Servlets By Class Names options, as shown in Figure 23-11.

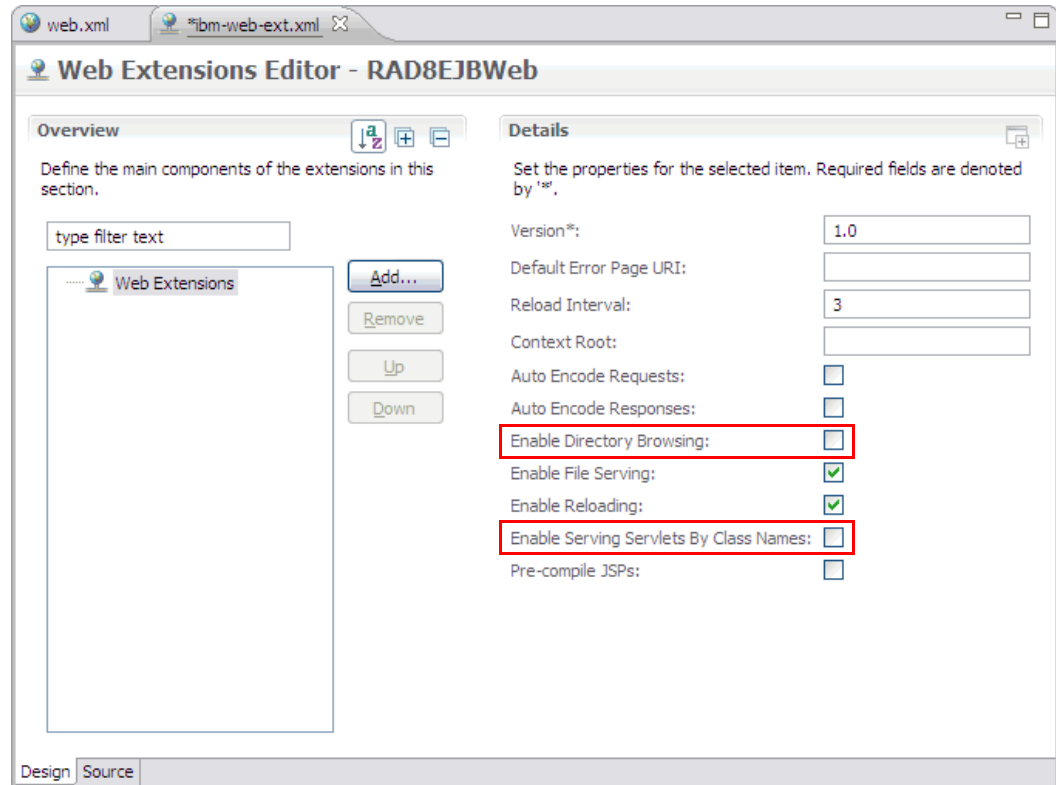


Figure 23-11 Web module extensions editor

4. When finished, press Ctrl+s to save the deployment descriptor and then close both the extensions file (ibm-web-ext.xml) and the web.xml file in the editor window.

23.3 Packaging recommendations

Consider the following basic rules when packaging an enterprise application:

- ▶ Package EJB JAR modules and WAR modules that make up an application together in the same EAR module and execute them within the same application server JVM process. This configuration avoids remote EJB calls (RMI/IIOP) across application server JVM processes, which is costly from a performance perspective.
- ▶ Place utility classes that are used by a single web module only within the web module's WEB-INF/lib folder.
- ▶ Place utility classes that are used by multiple modules within an enterprise application at the root of the EAR file as Utility Projects so that they are accessible by both servlets and EJB.
- ▶ Place utility classes that are used by multiple enterprise applications outside the applications on a directory that is referenced through a shared library definition.
- ▶ Keep the class path clean and reference only required libraries for your application.

23.4 Creating WebSphere-enhanced EAR files

A WebSphere-enhanced EAR file is a regular Java EE EAR file but with additional configuration information for resources required by Java EE applications. This information is processed by WebSphere Application Server. Any other application server ignores this information. Although adding this extra configuration information at packaging time is not mandatory, it can simplify deployment of Java EE applications to multiple run times if the environments are similar.

If you plan to provide the configuration of these resources by using the runtime environment administrative tools, you can skip this section and go directly to 23.5, “Exporting an application project to an EAR file” on page 845.

When an enhanced EAR is deployed to WebSphere Application Server, the resources specified in the enhanced EAR are automatically configured at application level scope. When an enhanced EAR is uninstalled, the resources that are defined at the application level scope are removed as well. However, resources that are defined at a scope other than application level are not removed because they might be in use by other applications. Resources that are created at the application level scope are limited in visibility to only that application.

Table 23-7 shows the resources that are supported by the enhanced EAR and the scope in which they are created.

Table 23-7 Scope for resources in WebSphere enhanced EAR file

| Resource | Scope |
|-----------------------------|-------------|
| JDBC providers | Application |
| Data sources | Application |
| Resource adapters | Application |
| JMS resources | Application |
| Substitution variables | Application |
| Class loader policies | Application |
| Shared libraries | Server |
| JAAS authentication aliases | Cell |
| Virtual hosts | Cell |

To view the application scoped resources using the administrative console, click **Applications** → **Application Types** → **WebSphere Enterprise Applications** → **<application>**. Select **Application scoped resources** in the References section. If there are no application scoped resources, you do not see this option.

23.4.1 Configuring a WebSphere enhanced EAR

You can modify the supplemental information in an enhanced EAR using the WebSphere Application Server Deployment editor of any of the assembly tools mentioned in 23.1.2, “Development tools” on page 816. The deployment information is contained in XML files in a folder called `ibmconfig` in the EAR file’s `META-INF` folder.

In the sample application, the provider is changed to use DB2. To make this change, the following configuration items are added to the deployment file:

- ▶ JAAS authentication alias
- ▶ JDBC provider for DB2
- ▶ Data source for DB2 database

A new virtual host for a domain called `www.itsobank.ibm.com` is also added.

To access the enhanced EAR deployment options, right-click the **RAD8EJBWebEAR** project, and select **Java EE**. Click **Open WebSphere Application Server Deployment** to open the editor, as shown in Figure 23-12.

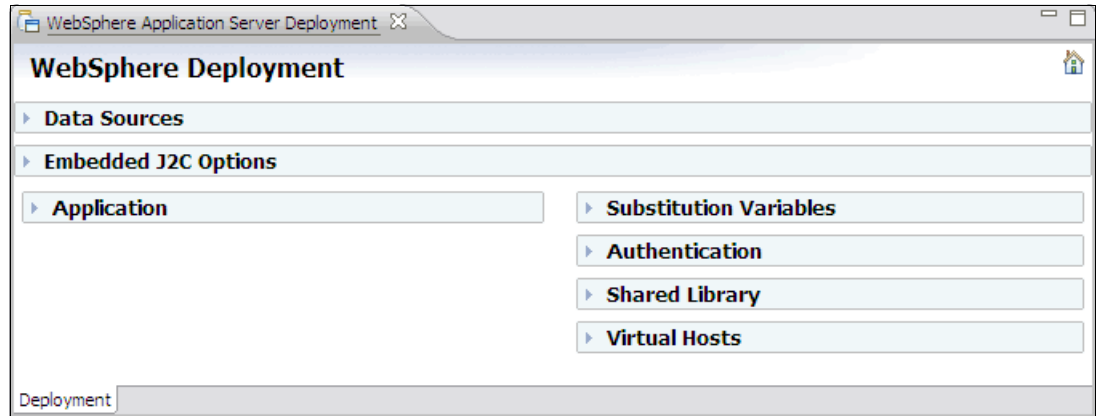


Figure 23-12 WebSphere enhanced EAR editor

23.4.2 Configuring application options

The Application section shown in Figure 23-13 contains the class loader policies and class loader mode configured for each of the containing modules. ITSO Bank runs with the default policies and modes. You do not need to change them. The *Auto start* feature is new with WebSphere Application Server V8.5. When set to Yes, the application starts at the application server start.

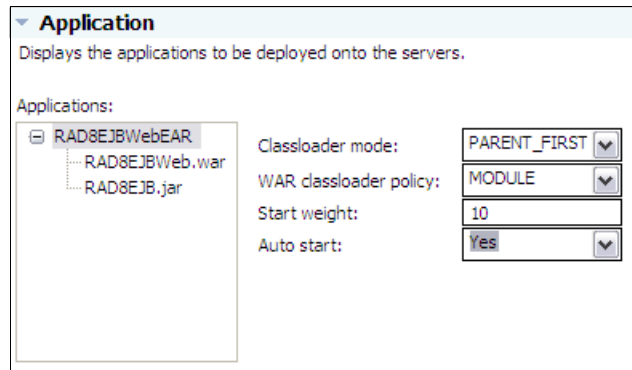


Figure 23-13 Configuring class loader mode and class loader policies

To configure the JAAS authentication alias required to access the application database:

1. Expand the **Authentication** section.
2. Click **Add**.

3. In the window that opens, enter the following information:
 - An alias of `itsobank`
 - A user ID with access to the ITSOBANK database (`db2inst1` in our case)
 - The password for the user ID
 - An optional description of ITSO Bank
4. Click **OK**. Figure 23-14 shows the results.

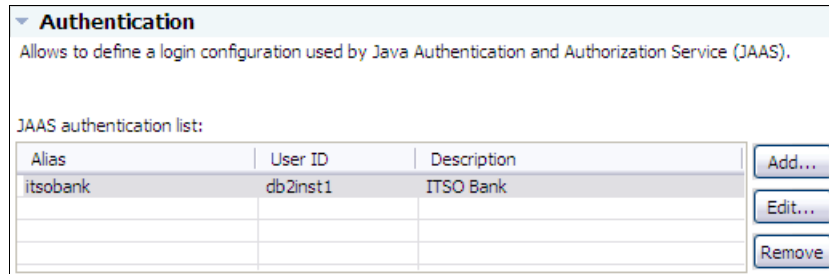


Figure 23-14 Configuring JAAS authentication alias for ITSO Bank

23.4.3 Configuring the JDBC provider and data source for DB2

To configure JDBC providers, expand the Data Sources section. Before you add the DB2 provider, delete the pre-configured Derby JDBC Provider (XA) provider by selecting it and by clicking **Remove**.

To configure the DB2 JDBC provider:

1. Click **Add** next to the JDBC provider list.
2. In the window that opens, select the following options, as shown in Figure 23-15 on page 838:
 - **IBM DB2** as the Database type.
 - **DB2 Universal JDBC Driver Provider** as the JDBC provider type.

Click **Next**.

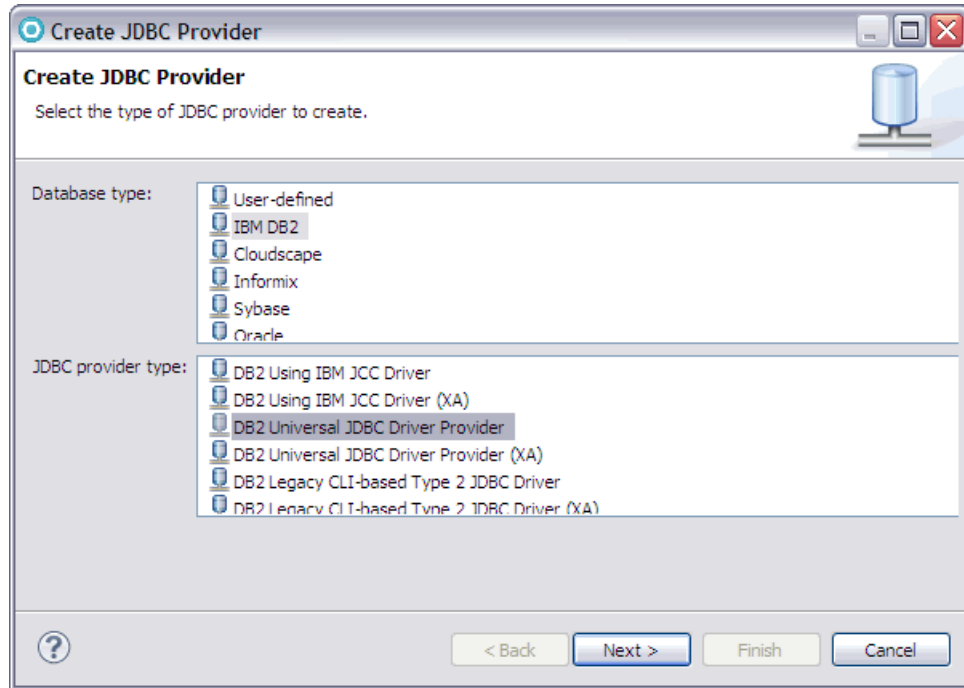


Figure 23-15 Creating a DB2 JDBC provider - Selecting the JDBC provider type

3. In the next window, enter a name for the JDBC provider (for administration purposes only), and leave the other properties as the default values. See Figure 23-16 on page 839. Click **Finish**.

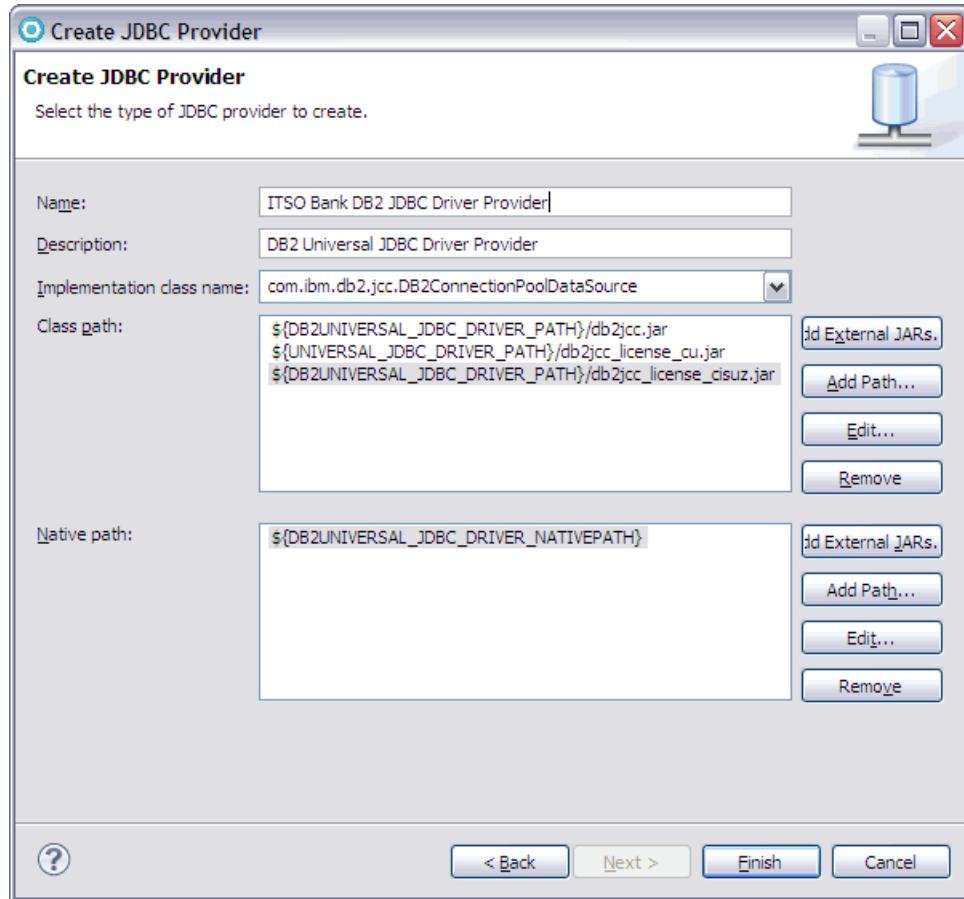


Figure 23-16 Creating a DB2 JDBC provider - Additional configuration

4. Click **Add** next to the Data source list, as shown in Figure 23-17.

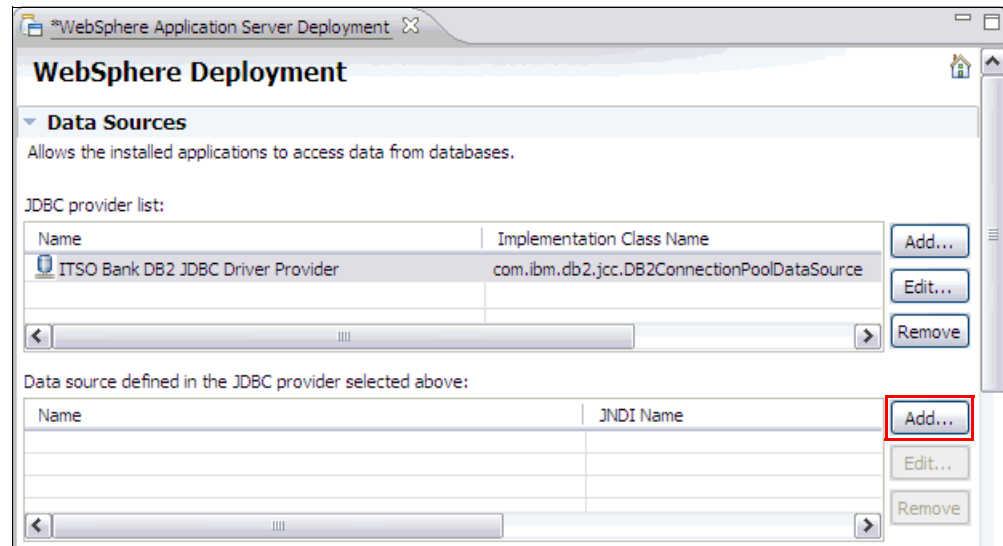


Figure 23-17 Creating a DB2 data source

5. In the Create a Data Source window, select **DB2 Universal JDBC Driver Provider** as the JDBC provider type and **Version 5.0 data source** as the data source type, as shown in Figure 23-18. Click **Next**.

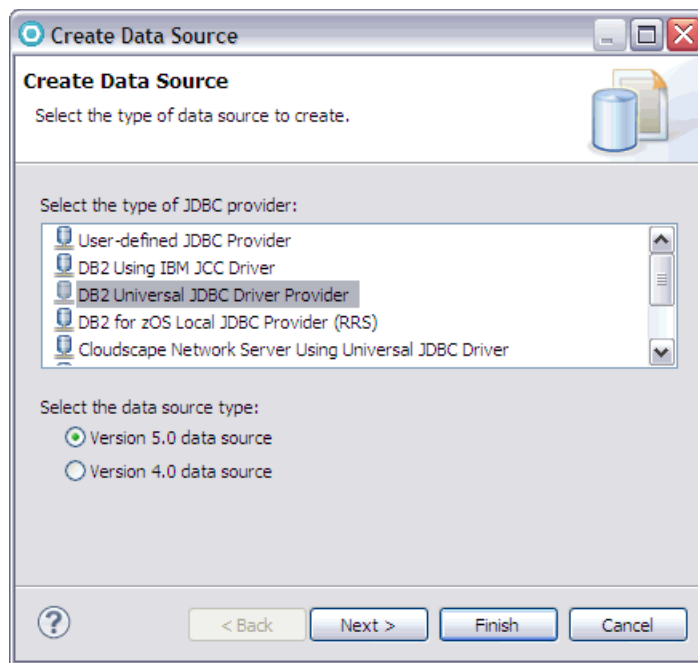


Figure 23-18 Creating a DB2 data source - Selecting the driver

6. In the window that opens, enter the appropriate values for the DB2 data source:
- Enter **ITS0BankDS** as the name.
 - Enter **jdbc/itsobank** as the JNDI name.
 - Enter **DB2 Data Source for ITS0 Bank** as the description.
 - Select **itsobank** as the component-managed authentication alias (you might need to scroll the window to the far right to see the drop-down menu). In most cases, you only need to supply the container-managed authentication alias. Setting the component-managed authentication alias allows the application to supply credentials to the application server when component-managed authentication is used.
 - Select **itsobank** as the container-managed authentication alias.
 - Clear the **Use this data source in container manager persistence (CMP)** option. The ITS0 Bank application uses JPA for persistence, so you do not need to add support for CMP Entity beans for this data source.

Click **Next**.

The window with these options is shown in Figure 23-19 on page 841.

Create Data Source

Select the type of data source to create.

Name: IITSOBankDS

JNDI name: jdbc/itsobank

Description: DB2 Data Source for ITSO Bank

Category:

Statement cache size: 10

Data source helper class name: com.ibm.websphere.rsadapter.DB2UniversalDataStoreHelper

Connection timeout: 180

Maximum connections: 10

Minimum connections: 1

Reap time: 180

Unused timeout: 1800

Aged timeout: 0

Purge policy: EntirePool

Component-managed authentication alias: itsobank

Container-managed authentication alias: itsobank

Use this data source in container managed persistence (CMP)

* Required field.

< Back Next > Finish Cancel

Figure 23-19 Creating a DB2 data source - Additional Information

7. Select the **databaseName** property in the Resource properties section. Enter ITS0BANK in the Value field. Select the **driverType** property, and change the value from type 4 to type 2. Finally, select the **serverName** property, and enter the IP address or host name of your database server.

Type 2 means that the driver supports only one-phase commit capabilities. Type 4 means that the driver supports two-phase commit capabilities. In this example, we do not need two-phase commit capabilities. See Figure 23-20 on page 842.

Click **Finish**.

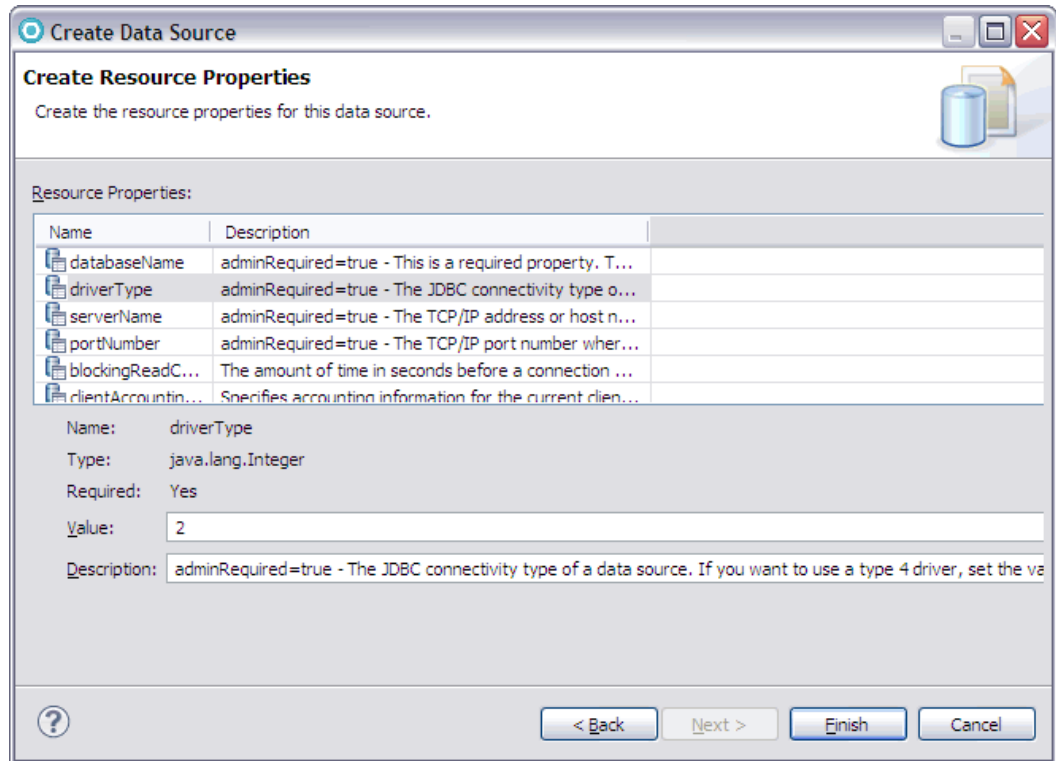


Figure 23-20 Creating a DB2 data source - Setting database name and driver type

When you are finished, your data source configuration looks similar to the window shown in Figure 23-21.

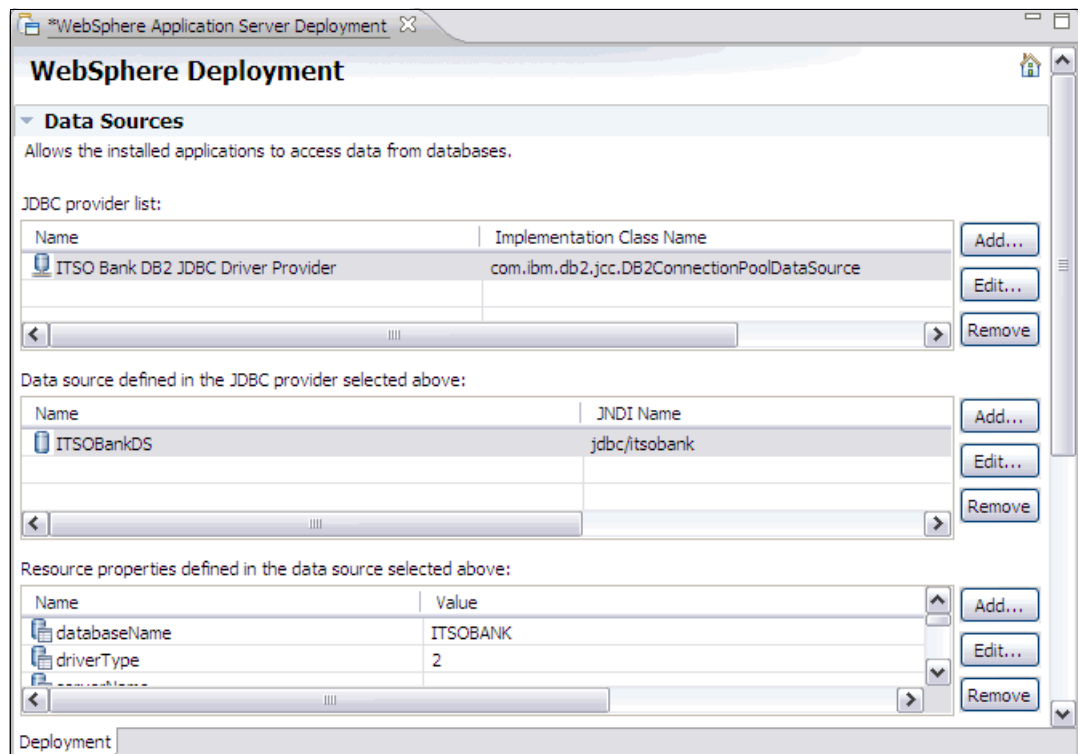


Figure 23-21 DB2 data source configured

23.4.4 Configuring substitution variables

As shown in Figure 23-16 on page 839, the JDBC driver provider configuration relies on the following variables:

- ▶ DB2UNIVERSAL_JDBC_DRIVER_PATH
- ▶ UNIVERSAL_JDBC_DRIVER_PATH
- ▶ DB2UNIVERSAL_JDBC_DRIVER_NATIVEPATH

We need to configure the values that these variables take when the application is deployed. To configure substitution variables:

1. Expand the **Substitution variables** section of the Deployment tab, and click **Add** next to the variables list.
2. Enter DB2UNIVERSAL_JDBC_DRIVER_PATH as the name of the variable.
3. Enter the location where the DB2 JDBC driver is located in the server, for example /opt/ibm/SQLLIB/java or C:\Program Files\IBM\SQLLIB\java.
4. Optionally enter a description.
5. Click **OK**.
6. Repeat the previous steps for the second and third variables. The variables list appear in the list as in Figure 23-22.

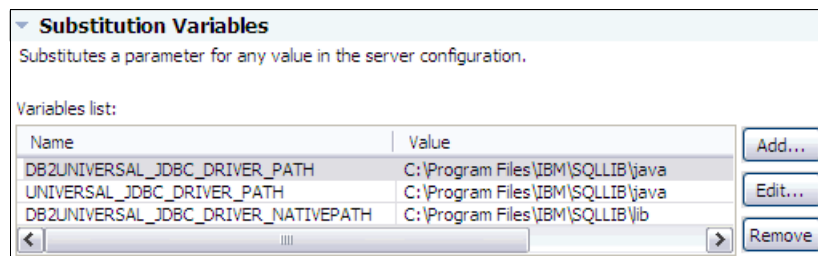


Figure 23-22 Substitution variables configured

23.4.5 Configuring a virtual host

To configure a virtual host:

1. Expand the **Virtual Hosts** section of the Deployment tab, and click **Add** next to the Virtual host name list.
2. In the Add Host Name Entry dialog box, enter itsobank_host, and click **OK**. Your new virtual host displays in the Virtual Hosts list.
3. Click **Add** next to the Host aliases list.
4. In the Add Host Alias Entry dialog box, enter www.itsobank.ibm.com for the host name and 80 for the port number. Click **OK**.

Repeat the procedure, and add number 9080 too. You use this port when you deploy the application later. If your server uses another port, use that port number instead. You can have as many host aliases as you want.

The virtual hosts and host aliases appear in lists, as shown in Figure 23-23 on page 844.

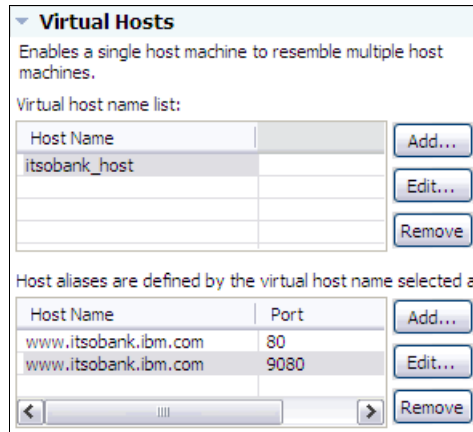


Figure 23-23 Virtual Hosts configuration

5. When you are finished, press Ctrl+s to save the deployment descriptor editor.

23.4.6 Setting the default virtual host for web modules

Although you configured a new virtual host, `itsobank_host` in the enhanced EAR file, the web modules do not use it automatically. The default virtual host for a web module that is created in Rational Application Developer or IBM Assembly and Deploy Tools for WebSphere Administration is `default_host`, which is also the case for the ITSO Bank application.

To modify the web modules to use `itsobank_host` instead, complete the following steps:

1. Expand the **RAD8EJBWeb** project, and double-click the **RAD8EJBWeb** heading to open the web module deployment descriptor.
2. In the lower-right corner of the window, click the **Open WebSphere Bindings Descriptor** link.
3. Change the Virtual Host Name to `itsobank_host`, as shown in Figure 23-24.

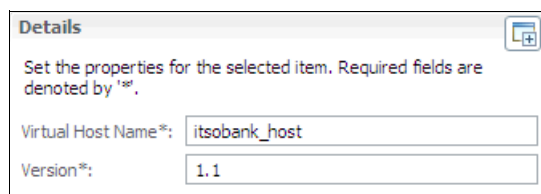


Figure 23-24 Setting the default virtual host for a web module

4. Save the deployment descriptor by pressing Ctrl+s and then close it.

23.4.7 Examining the WebSphere-enhanced EAR file

The information about the configured resources is stored in the `ibmconfig` subdirectory of the EAR file's `META-INF` directory. Expanding this directory reveals the directory structure for a cell configuration, as shown in Figure 23-25 on page 845. You can also see the scope level where each resource is configured.

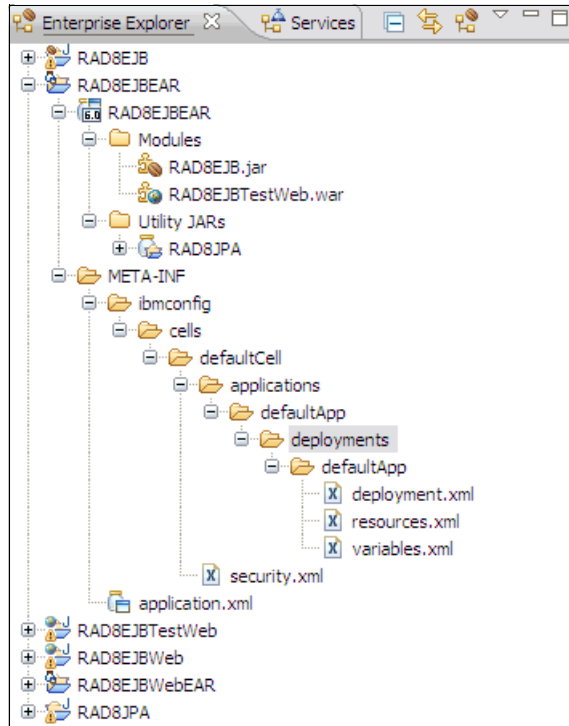


Figure 23-25 Enhanced EAR file contents

At deployment time, WebSphere Application Server uses this information to create the resources automatically.

23.5 Exporting an application project to an EAR file

To deploy an application, package the application as an EAR file. To export the RAD8EJBWebEAR application as an EAR file with all its dependent modules, complete the following steps:

1. Select the **RAD8EJBWebEAR** project, and right-click **Export** → **EAR file** from the pop-up menu.
2. In the Export dialog box, browse to a destination, as shown in Figure 23-26 on page 846.

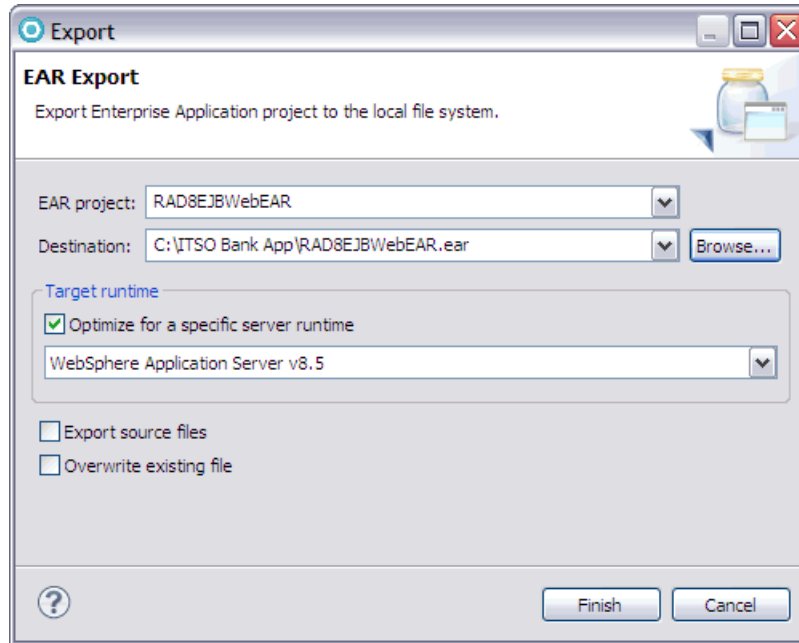


Figure 23-26 Exporting to an EAR file

3. Click **Finish**.

The EAR file that you export is now prepared for installation in WebSphere Application Server V8.5.

Version note: If you select the **Optimize for a specific server runtime** option, the application runs only on the version that you specify or on later versions. In this case, for example, the application will not run on WebSphere Application Server V8 if you select this option.

23.6 Preparing the runtime environment for the application

In this section, we show you how to set up a complete environment for the ITSO Bank application and how to deploy the EAR file. You will not always need or want to customize the environment as extensively as we do in this example. Some steps are optional. If all you want to do is deploy your application quickly, using the WebSphere defaults for directory names, log files, and so on, skip to 23.7, “Deploying the application” on page 855.

To deploy the ITSO Bank application:

1. Optional: Create an environment variable for the application file directory.
2. Optional: Create an application server to host the application.
3. Optional: Customize the IBM HTTP Server configuration.
4. Define a JDBC provider, data source, and authentication alias. If you are using an enhanced EAR file, this step is optional.
5. Define virtual hosts. If you are using an enhanced EAR file, this step is optional.

If the application to be deployed is a WebSphere enhanced EAR file, the resources configured in the enhanced EAR file are created automatically when the application is

deployed. The resources that can be configured in the enhanced EAR are described in the following sections.

23.6.1 Creating an environment variable for the application file directory

Use WebSphere environment variables, rather than hardcoded paths, when deploying an application. In this step, a variable called `ITSOBANK_ROOT` is defined. The variable represents the root directory where application resources are stored. You use this variable in subsequent configuration processes when specifying, for example, the JVM log location.

Complete the following steps to specify the log location:

1. Determine how you want to organize your application files.

There are several ways to organize WebSphere applications. You can create a directory for each application, such as `/apps/application_name`, and keep all resources and directories that are required by the application in subdirectories under this directory. This strategy works well when deploying only one application per application server because the application server's log files can then all be changed to point to the `/apps/application_name/logs` directory.

An alternative is to organize resources by resource type and create directories, such as `/apps/logs/application_name.log`, `/apps/properties/application_name.properties`, and so on.

Finally, you can keep the vendor defaults as much as possible. For WebSphere, vendor defaults mean that the applications are installed in the `profile_root/installedApps` directory and that the logs files are written to the `profile_root/logs/server_name` directory.

The option that you choose is a matter of personal preferences and corporate guidelines.

In this example, we deploy a single application to a single application server. A directory called `/apps/ITSOBANK` will be used to contain the application resources.

2. Create the target directory on the application server operating system:

```
/apps/ITSOBANK
```

Important: If you define and use the variable in configuration steps, but the directory is not created, the application server might not start.

3. Use the steps that are described in 6.1.10, "Using variables" on page 205 to create an `ITSOBANK_ROOT` variable with a value of `/apps/ITSOBANK`.

Be certain that you declare this variable at the correct scope. For example, if you define this variable at the application server scope, it is known only at that level. As long as you work with the WebSphere Application ServerBase or Express editions, this configuration is fine. However, if you are using the Network Deployment edition and you create a cluster of application servers, you need to define the `ITSOBANK_ROOT` variable at the cluster or cell level.

23.6.2 Creating the ITSO Bank application server

In a distributed server environment, you can use a single application server or create multiple application servers or clusters.

The advantage of deploying multiple applications to a single application server is that it consumes less resources. There is no impact for any extra application server processes.

Another benefit is that applications can make in-process calls to each other. For example, servlets in one EAR file can access the local interfaces of the EJB beans in another EAR file.

One alternative to using a single application server is to deploy each application to its own server. The advantages of deploying only one application on an application server is that it gives you greater control over the environment. The JVM heap sizes and environment variables are set at the application server level. Thus, all applications running in an application server share the JVM memory given to the application server and all see the same environment variables. Running each application in its own application server can also make it easier to perform problem determination. For example, if an application consumes a lot of CPU resources, you can determine which application is consuming CPU resources by looking at the process ID of the application server.

In our example, we have a unique application server on which we run the ITSO Bank sample application. Use the instructions in 7.4.1, “Creating an application server” on page 248 to create a new application server called `ITSOBankServer1`.

Changing the application server working directory

This step changes the working directory for the application server process. This directory is the relative root for searching files. For example, if you perform a `File.open(“filename.gif”)` command, `filename.gif` must be present in the working directory. Create a specific working directory for each application server.

To change the working directory:

1. Create the working directory on the operating system:

```
/apps/ITSOBANK/workingDir
```

Important: The working directory is not created automatically. Create the path before starting the application server or the start sequence fails.

2. In the administrative console, click **Servers** → **Server Types** → **WebSphere Application Servers**. Click the `ITSOBankServer1` server.
3. Expand the **Java and Process Management** in the Server Infrastructure section, and select **Process Definition**.
4. Scroll down the window and change the working directory from `${USER_INSTALL_ROOT}` to `${ITSOBANK_ROOT}/workingDir`.
5. Click **OK**. Save the changes to the master configuration.

Changing the application server logging and tracing options

Next, customize the logging and tracing properties for the application server. The properties are updated so the logs are stored in the `ITSOBANK_ROOT/logs` directory.

Complete the following steps to customize logging and tracing properties:

1. Create the `/apps/ITSOBANK/logs` directory on the operating system.
2. Update the logging and tracing properties to use this directory.

The following list notes several ways to access the logging and tracing properties for an application server in the administrative console:

- Click **Troubleshooting** → **Logs and Trace** in the navigation bar and then select a server.

- Click **Servers** → **Server Types** → **WebSphere application servers**, select a server and then select **Logging and Tracing** from the Troubleshooting section.
- Click **Servers** → **Server Types** → **WebSphere application servers**, select a server and then select **Process definition** from the Java and Process Management section.

Because you just updated the application server process definition, you can use the third navigation path to customize the location of the JVM logs, the diagnostic trace logs, and the process logs. Then, select **Logging and Tracing** from the Additional Properties section. For this example, we use basic logging. However, you might want to use High Performance Extensible Logging (HPEL). For more information about HPEL, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/a/ctrb_HPELCompat.html

3. Select **JVM Logs**.

This option allows you to change the JVM standard output and error file properties. Both the JVM standard output and the error file properties are rotating files. You can choose to save the current file and create a new one, either when it reaches a certain size or at a specific moment during the day. You can also choose to disable the output of calls to `System.out.print()` or `System.err.print()`.

Specify the new file name using an environment variable similar to the following syntax:

```
${ITSOBANK_ROOT}/logs/SystemOut.log for System.out  
${ITSOBANK_ROOT}/logs/SystemErr.log for System.err
```

In this window, you can also modify how WebSphere rotates log files.

Click **OK**.

4. Select **Diagnostic Trace**.

Each component of the WebSphere Application Server is enabled for tracing. This trace can be changed dynamically while the process is running using the Runtime tab or can be added to the application server definition from the Configuration tab. As shown in Figure 23-27 on page 850, the trace output can be directed either to memory or to a rotating trace file.

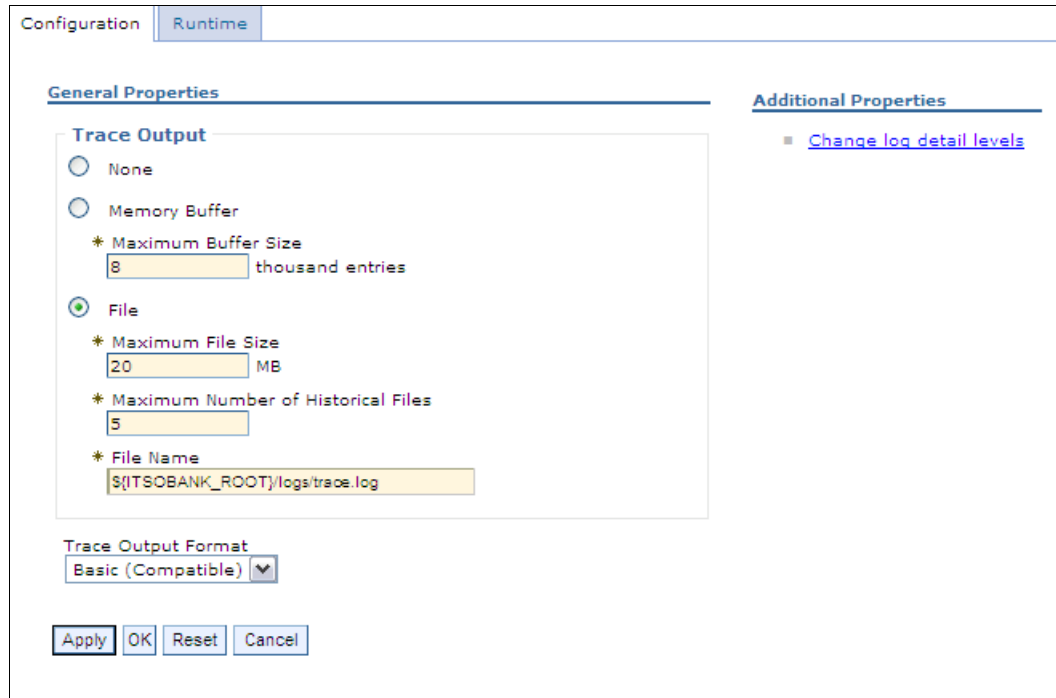


Figure 23-27 Specifying diagnostic trace service options

Change the trace output file name so that the trace is stored in a specific location for the server using the ITSOBANK_ROOT variable, similar to the following syntax:

```
${ITSOBANK_ROOT}/logs/trace.log
```

Select the **Basic** format.

Click **OK**.

5. Select **Process Logs**.

Messages written by native code (JNI) to standard out and standard error streams are redirected by WebSphere to process logs, usually called `native_stdout.log` and `native_stderr.log`. Change the native process logs to use the following paths:

```
${ITSOBANK_ROOT}/logs/native_stdout.log
```

```
${ITSOBANK_ROOT}/logs/native_stderr.log
```

Click **OK**.

6. All log files that are produced by the application server are now redirected to the `${ITSOBANK_ROOT}/logs` directory. Save the configuration.

23.6.3 Defining the ITSO Bank virtual host

Verify the port number: The remainder of this example assumes a default HTTP port of 9080 for the web container. Before you continue, check the application server you created to determine the port that you use by completing the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers**.
2. Select **ITSOBankServer1**.
3. Select **Ports** in the Communications section.
4. Scroll down the window and then note the port listed for `WC_defaulthost`.

Enhanced EAR file users: If you are using an enhanced EAR file, you can define the virtual host at packaging time. See 23.4.4, “Configuring substitution variables” on page 843.

Web modules need to be bound to a specific virtual host. For our sample, we chose to bind the `RAD8EJBWeb` web module to a specific virtual host called `itsobank_host`. This virtual host has the following host aliases:

- ▶ `www.itsobank.ibm.com:80`
- ▶ `www.itsobank.ibm.com:9080`

Any request starting with `itsobank_host_alias/RAD8EJBWeb`, such as `http://www.itsobank.ibm.com:9080/RAD8EJBWeb`, is served by the `RAD8EJBWeb` application.

Tip: You can restrict the list of hosts that are used to access the web application by removing hosts from the virtual host definition. For example, if you want to prevent users from accessing the ITSO Bank application directly from the WebSphere internal HTTP server when they invoke `http://www.itsobank.ibm.com:9080/RAD8EJBWeb`, you can remove `www.itsobank.ibm.com:9080` from the virtual host aliases list. Then, all requests go through the web server plug-in.

To create the `itsobank_host` virtual host:

1. Click **Environment** → **Virtual Hosts** in the navigation pane.
2. Click **New**.
3. Enter the virtual host name, `itsobank_host` and then click **Apply**.
4. Select **Host Aliases** in the **Additional Properties** section.
5. Add the two aliases shown in Figure 23-28 on page 852 by clicking **New**, entering the values and then clicking **OK**.



Figure 23-28 ITSO Bank virtual host aliases

6. Click **OK** and save the changes to the master configuration.

23.6.4 Creating the virtual host for the IBM HTTP Server

Now that you defined the `itsobank_host` virtual host, you need to configure the web server to serve the host aliases in the virtual host.

Configuring virtual hosting

Tip: You do not need to create a virtual host in the `httpd.conf` file. A virtual host in the `httpd.conf` file is required only if you want to customize the configuration, for example, by separating the logs for each virtual host.

You create virtual hosts using the `VirtualHost` directive, as shown in Example 23-1.

Example 23-1 Configuring virtual hosts in IBM HTTP Server `httpd.conf` file

```
<VirtualHost www.itsobank.ibm.com:80>
  ServerAdmin webmaster@itsobank.ibm.com
  ServerName www.itsobank.ibm.com
  DocumentRoot "/opt/IBM/HTTPServer/htdocs/itsobank"
  ErrorLog logs/itsobank_error.log
  TransferLog logs/itsobank_access.log
</VirtualHost>
```

If you want to have multiple virtual hosts for the same IP address, you must use the `NameVirtualHost` directive, as shown in Example 23-2.

Example 23-2 Using the `NameVirtualHost` and `VirtualHost` directives in `httpd.conf` file

```
NameVirtualHost 9.42.171.97:80

<VirtualHost itso_server:80>
  ServerAdmin webmaster@itso_server.com
  ServerName itso_server
```

```
DocumentRoot "/opt/IBM/HTTPServer/htdocs/itso_server"
ErrorLog logs/itso_server_error.log
TransferLog logs/itso_server_access.log
</VirtualHost>

<VirtualHost www.itsobank.ibm.com:80>
ServerAdmin webmaster@itsobank.ibm.com
ServerName www.itsobank.ibm.com
DocumentRoot "/IBM/HTTPServer/htdocs/itsobank"
ErrorLog logs/itsobank_error.log
TransferLog logs/itsobank_access.log
</VirtualHost>
```

The `www.itsobank.ibm.com` and the `itso_server` hosts have the same IP address, `9.42.171.97`. We set this address by inserting the following line in the machine hosts file, which is located in `%windir%\system32\drivers\etc` in Windows systems or in `/etc` on UNIX systems:

```
9.42.171.97 www.itsobank.ibm.com itso_server
```

In a production environment, name resolution is achieved by creating aliases at the DNS level. In any event, you must be able to ping the host that you defined using a command, such as `ping www.itsobank.ibm.com`.

As shown in Example 23-2 on page 852, each virtual host has a different document root. Make sure that the directory that you specify exists before you start the HTTP server. While testing the setup, you can place an `index.html` file at the document root stating which virtual host is being called, which lets you determine easily the virtual host that is being used.

You must restart the IBM HTTP Server to apply these changes. If you are running a Windows system, try to start the server by running `apache.exe` from the command line rather than from the Services window. This method allows you to spot error messages that are thrown at server start. In Linux or UNIX servers, you can run `apachectl` with the `-t` flag to check for errors in the `httpd.conf` file.

If your virtual hosts are correctly configured, invoking `http://www.itsobank.ibm.com` or `http://itso_server` returns different HTML pages.

23.6.5 Creating a DB2 JDBC provider and data source

Enhanced EAR file users: If you are using an enhanced EAR file, you can define the JDBC provider, data source, and J2C authentication entry at packaging time. Refer to 23.4.3, “Configuring the JDBC provider and data source for DB2” on page 837 for more information.

The ITSO Bank sample application uses a relational database with the Java Persistence API to store information. To access this database, you need to define a data source with a JNDI name that matches the data source configuration in the JPA module `persistence.xml` file. Figure 23-29 on page 854 shows the `persistence.xml` file for the `RAD8JPA` project open in the Rational Application Developer for WebSphere Software. As you can see, the JNDI name in this case is `jdbc/itsobank`.

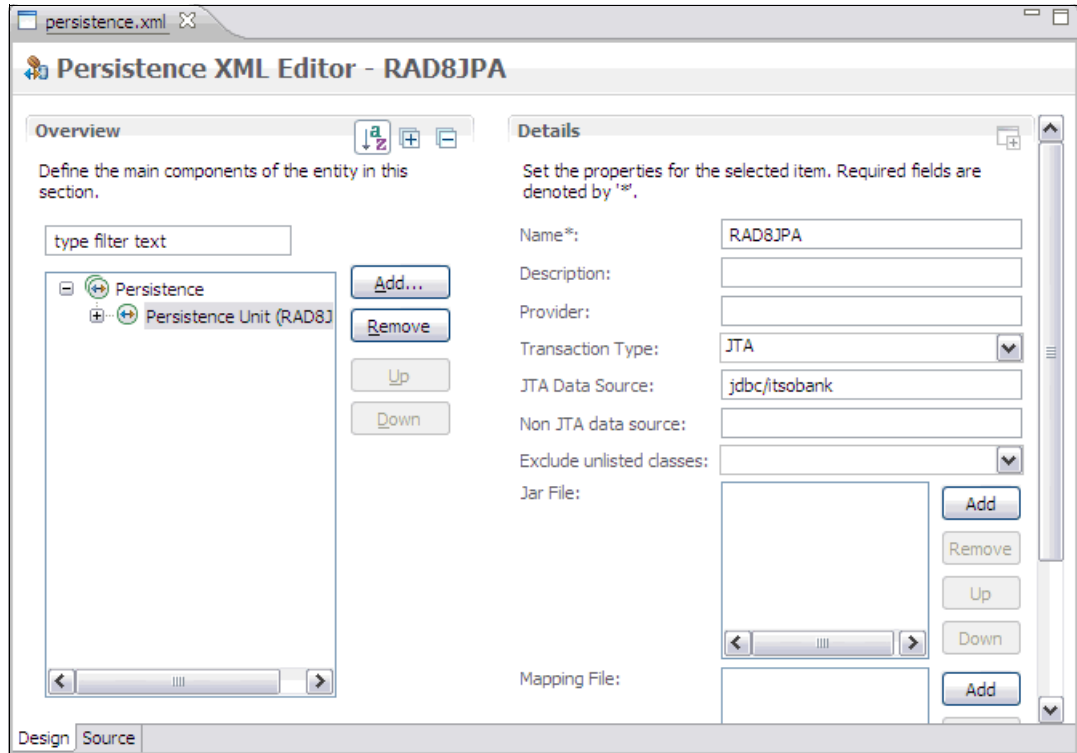


Figure 23-29 Persistence.xml file opened in an editor

The ITSO Bank sample application is configured for Derby by default, but we use a DB2 database with the application. In our example here, we create the DB2 JDBC provider, data source, and Java Authentication and Authorization Service (JAAS) authentication alias that are required to run against DB2.

Create the data source for the ITSO Bank DB2 database with the following information:

- ▶ Data source name: ITS0BankDS
- ▶ JNDI name: jdbc/i tsobank
- ▶ Driver type: 2
- ▶ Database name: ITS0BANK
- ▶ Server name: The DB2 server IP address or host name
- ▶ Port number: 50000
- ▶ Clear the option *Use this Data Source in container-managed persistence (CMP)*
- ▶ Container-managed authentication alias: Selecting the corresponding authentication alias.

Be sure to test the connection before you continue.

If you need further explanation about how to achieve these configurations, refer to Chapter 9, “Accessing relational databases from WebSphere” on page 351.

23.7 Deploying the application

WebSphere Application Server V8.5 provides the following ways to deploy a JEE application:

- ▶ Administrative console install wizard
- ▶ Monitored directory
- ▶ Application properties files
- ▶ wsadmin scripts
- ▶ Job manager runs wsadmin scripts
- ▶ Java application programming interfaces
- ▶ Rapid deployment tools

In this chapter, we cover three different options: deploying through the administrative console, deploying using the monitored directory feature, and deploying through the job manager.

For information about using the other options, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/crun_app_install.html

23.7.1 Deploying using the administrative console

To deploy the application using the administrative console:

1. Click **Applications** → **New Application** from the administrative console navigation bar and then click **New Enterprise Application** on the window.
2. Select the **Local file system** option and then click **Browse** to locate the RAD8EJBWebEAR.ear file. Select the file, and click **Open**.

From the installation window, you can install files that are located either on the same system as the browser that you are using to access the administrative console (the local file system option) or on any node in the cell (the remote file system option).

After you make your selection, click **Next**. The selected file is uploaded to the application server or to the deployment manager if this is a distributed server environment.

3. The administrative console allows you to take a shortcut when installing an application. If you select the **Fast Path - Prompt only when additional information is required** option, only the configuration windows where you actually need to complete information during installation are shown.

For this example, however, we take the detailed path to explain the options. Select the **Detailed - Show all installation options and parameters** option.

If you expand **Choose**, to generate default bindings and mapping, you can alter the bindings for the application that you are deploying. If you select the **Generate Default Bindings** option, WebSphere Application Server completes any incomplete bindings in the application with default values, but it does not alter any existing bindings. Selecting the **Override existing bindings** option allows you to specify a bindings file that contains new bindings.

The contents of the application or module that you are installing determines which options are displayed on the bindings window. For our ITSO Bank application, the options documented in Table 23-8 on page 856 are displayed.

Table 23-8 Application default bindings

| Binding name | Detailed information |
|-------------------------|--|
| Specific bindings file | You can create a specific bindings file using your favorite editor and load it during application installation by clicking Browse next to the specific bindings file. |
| Unique prefix for beans | You can generate default EJB JNDI names using a common prefix. EJB beans for which you did not specify a JNDI name get a default name, built by concatenating the prefix and the EJB name. If you specify a prefix of myApp/ejb, JNDI names default to myApp/ejb/EJBName, such as myApp/ejb/Account. |
| Virtual host bindings | You can bind all web modules to a specific virtual host, such as itsobank_host. |

Because our application uses Java EE 6 and EJB 3.1 and relies on the bindings and mappings that are generated automatically by the EJB container, you do not need to override bindings. Leave all options for bindings cleared.

Click **Next**.

The remainder of the wizard is divided into steps. The number of steps depends on your application. For example, if the application contains EJB modules or web modules, you are prompted for the information necessary to deploy them.

4. Step 1: Select installation options.

Step 1 gives you a chance to review the installation options. You can specify various deployment options, such as JSP precompiling and whether you want to generate EJB deployment code (not applicable for EJB 3 or later beans). In this step, you can set the following options:

- If you are deploying an enhanced EAR file, you specify here whether to use the resource configuration information that is packaged in the enhanced EAR file. The installation window pre-selects the Process embedded configuration option. If you do not want to use the resource configuration information that is packaged in the enhanced EAR file, clear this option. In this example, we configured the necessary resources using the administrative console, so make sure that this option is not selected. If you did not configure the resources using the administrative console and you are deploying the enhanced EAR created in 23.4, “Creating WebSphere-enhanced EAR files” on page 835, leave this option selected.

- Selecting the **Precompile JavaServer Pages files** option makes WebSphere compile all JSP pages in the EAR file during installation instead of during run time. Thus, the first user who accesses the application does not have to wait for the JSP pages to compile.

An alternative to precompiling JSP pages is to use the **JspBatchCompiler** script found in the `bin` directory of the profile that you are using to compile the JSP pages after the application is installed.

- You can specify file permissions for files in your application. To use one of the predefined file permissions, select it. You can also specify your own file permissions using regular expressions.
- The administrative console displays the Application Build ID of the application that is being installed. This string is specified in the MANIFEST.MF file in the EAR file’s META-INF folder and can be set using the Rational Application Developer for WebSphere tool.

The following example shows a version number that is specified in the MANIFEST.MF file:

Implementation-Version: Version 1.2.3

- The dispatching and servicing of remote resources are extensions to the web container that allows frameworks, servlets, and JSP pages to include content from outside of the current executing resource’s JVM as part of the response that is sent to the client.

To enable these features, select the options to allow dispatching or servicing to or from remote resources.

- The Allow EJB reference targets to resolve automatically option is used for EJB 2.1 or earlier or Web 2.3 or earlier modules and allows WebSphere Application Server to provide a default value or to resolve EJB references automatically for any EJB reference that does not have a binding. Because the sample application is at EJB 3.1, this option does not apply.

For more information about the options in this window, refer to the following website:


http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/a/urun_rapp_installoptions.html

In this example, we keep the default values. Click **Next**.

5. Step 2: Map modules to servers.

Here you can select the server or cluster on which you want each module deployed. For better performance, deploy all modules from one application in a single server. Specifically, do not separate the EJB clients, usually servlets in web modules, from the EJB beans themselves.

If you only have one server defined, the modules are mapped to this server by default.

Click the  icon to select all modules in the ITSO Bank EAR file. In the Clusters and Servers field, select **ITSOBankServer1**. Click **Apply** to assign all modules to the ITSOBankServer1 application server. If you deploy to a cluster, select the cluster instead of the single application server. See Figure 23-30.

Web servers: If you have a web server defined, select both the **web server** and **ITSOBankServer1** in the server list. Press and hold the Ctrl key to select multiple servers. Mapping web modules to web servers ensures that the web server plug-in is generated properly.

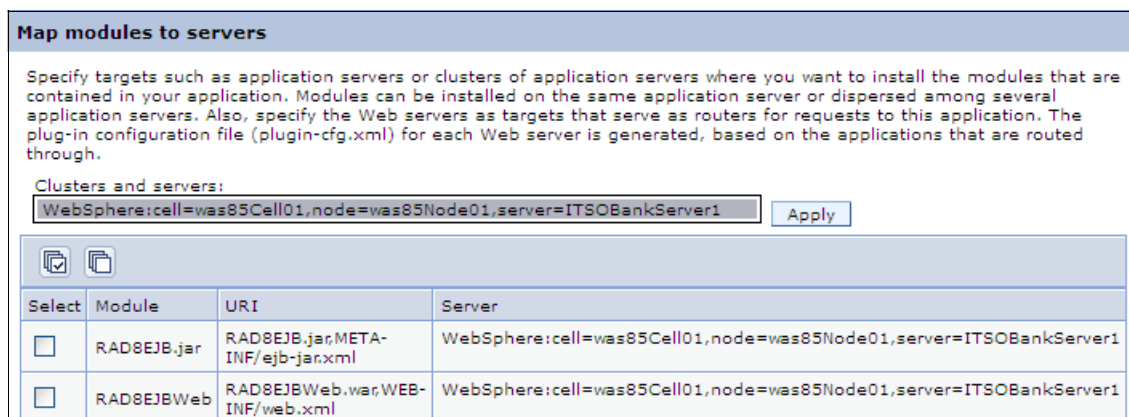


Figure 23-30 Mapping modules to application servers

After you map the module, click **Next**.

6. Step 3: Provide JSP reloading options for web modules.

This setting allows you to configure if and how often WebSphere checks for updates to JSP files and whether they are reloaded. In a production environment. You might want to disable this setting to improve performance. Click **Next**.

7. Steps 4 and 5: Map shared libraries, and Map shared library relationships.

If your application depends on shared libraries, you can specify them here. Click **Next**.

8. Step 6: Initialize parameters for servlets.

For servlets that honor initialization parameters (specified by the `init-param` tag in the web module's `web.xml` deployment descriptor), you can configure the value of the parameters. Click **Next**.

9. Step 7: Provide JNDI names for beans.

Use this window to bind the enterprise beans in your application or module to a JNDI name. Because our application is at EJB 3.1, we can leave it blank to have WebSphere Application Server use the default names. Click **Next**.

10. Step 8: Bind EJB Business interfaces to JNDI names.

Specify a JNDI name for the business interfaces of your EJB beans. Because our application is at EJB 3.1, we can leave this setting blank to have WebSphere Application Server assign default JNDI name. Click **Next**.

11. Step 9: Map EJB References to beans.

Each EJB reference that is defined in your application must be mapped to an enterprise bean. We can leave it blank to have WebSphere Application Server use the default names. If the reference was in an EJB 2 or earlier or Web 2.3 or earlier module, you can select the **Allow EJB reference targets to resolve automatically** option and then not need to specify a target JNDI name either. Click **Next**.

12. Step 10: Map virtual hosts for web modules.

Select the virtual host that you created for the application (`itsobank_host`).

Virtual host definitions: Only those virtual host definitions that are defined to the WebSphere environment are configurable at this step. To map the web modules to the virtual host that is defined in the enhanced EAR file, you need to configure that host after deploying the application.

Click **Next**.

13. Step 11: Map context roots for web modules.

Select the context root against to bind the module. Click **Next**.

14. Step 12: Map JASPI provider.

If your application has web modules, you can specify values to override the JASPI settings from the global or domain security configuration. Click **Next**.

15. Step 13: Metadata for modules.

Selecting the **metadata-complete** attribute tells WebSphere Application Server to ignore any deployment information that is specified in source code annotations. Leave both options cleared to use the information from the annotations. Click **Next**.

16. Step 14: Display module build Ids.

If the `MANIFEST.MF` file of a module in an enterprise application specifies a build identifier, this page shows the build identifier of the module. Click **Next**.

17. Step 15: Summary.

The Summary window gives an overview of the application deployment settings. If those settings are fine, click **Finish** to deploy the application.

18. Save the configuration.

If you are working in a distributed server environment, make sure that you synchronize the changes with the nodes so that the application is propagated to the target application server or servers.

If you mapped the web modules to a web server, make sure that the web server plug-in is regenerated and propagated to the web server. For a quick refresh, restart the web server.

19. Start the application:

- a. Click **Applications** → **Application Types** → **WebSphere Enterprise Applications**.
- b. Select the **RAD8EJBWebEAR** application and then click **Start**.

Deployment of the RAD8EJBWebEAR application is now complete. You can verify that the application works by pointing your browser to the following website:

<http://www.itsobank.ibm.com:9080/RAD8EJBWeb>

If successful, the web page for the ITSO Bank application displays. Provide customer number 111-11-1111 to see an example of a customer's accounts, and click **Submit**, as shown in Figure 23-31.



The screenshot shows the ITSO RedBank web application interface. At the top left is a logo with two red spheres. The title "ITSO RedBank" is displayed in red. Below the title is a form for customer information. The SSN is 111-11-1111. The Title is Mr, First name is Henry, and Last name is Cui. There are three buttons: Update, New Customer, and Delete Customer. Below the form is a table with two columns: Account Number and Balance. The table contains three rows of account information. At the bottom of the page are two buttons: Add Account and Logout.

| Account Number | Balance |
|----------------|-----------|
| 001-111001 | 12,345.67 |
| 001-111002 | 6,543.21 |
| 001-111003 | 98.76 |

Figure 23-31 ITSO Bank web application

If you have any problems related to virtual hosts, restart the server and try again. WebSphere Application Server needs a restart to pick up virtual host changes.

23.7.2 Deploying using the monitored directory support feature

With WebSphere Application Server V8.5, you can deploy or update applications automatically by copying an application archive to a specified, *monitored directory*. Monitored directories support the following types of applications:

- ▶ Enterprise archive (EAR)
- ▶ Web archive (WAR)
- ▶ Java archive (JAR)
- ▶ Session Initiation Protocol (SIP) module (SAR)

The monitored directory feature is disabled by default.

IBM i: Using a monitored directory for application deployment is not supported on IBM i operating systems.

The default monitored directory for WebSphere Application Server V8.5 is *profile_root/monitoredDeployableApps/servers/server_name* directory for stand-alone servers. For distributed systems, the following directories are used:

Deployment managers use the following default monitored directories:

- ▶ *dmgr_profile_root/monitoredDeployableApps/servers/server_name* directory for all servers named *server_name*.
- ▶ *dmgr_profile_root/monitoredDeployableApps/nodes/node_name/servers/server_name* directories for a specific *server_name* on a specific *node_name*
- ▶ *dmgr_profile_root/monitoredDeployableApps/clusters/cluster_name* directory for the members of *cluster_name*

Figure 23-32 shows a cell topology with two nodes (was85Node01 and was85Node02). Each node has one stand-alone server (server1) and one server that is a member of a cluster (member1 and member2).



Figure 23-32 Sample topology with three possible run times

Figure 23-33 illustrates the monitored directory structure for the topology shown in Figure 23-32 on page 860. The directories marked with boxes in Figure 23-33 are the directories where you can copy an application to deploy it at a specified run time.

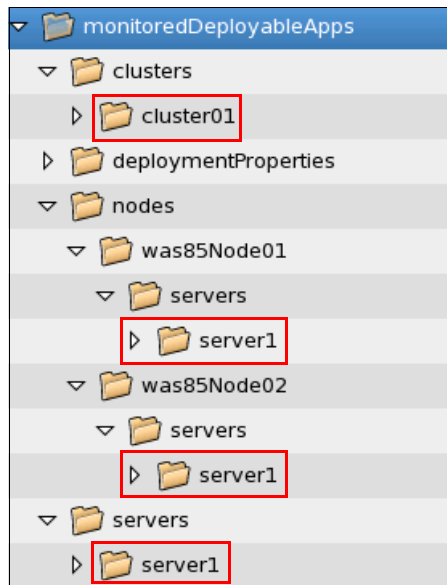


Figure 23-33 Sample monitored directory structure

Important: WebSphere Application Server will not create the whole directory structure. You have to manually create the directory structure for monitored directory.

Setting up a monitored directory

To enable and set up the monitored directory in WebSphere Application Server, complete the following steps:

1. In the administrative console, click **Applications** → **Global deployment settings**.
2. Select the **Monitor directory to automatically deploy applications** option, as illustrated in Figure 23-34 on page 862. You can also change the default location of the monitored directory in the Monitored directory field.

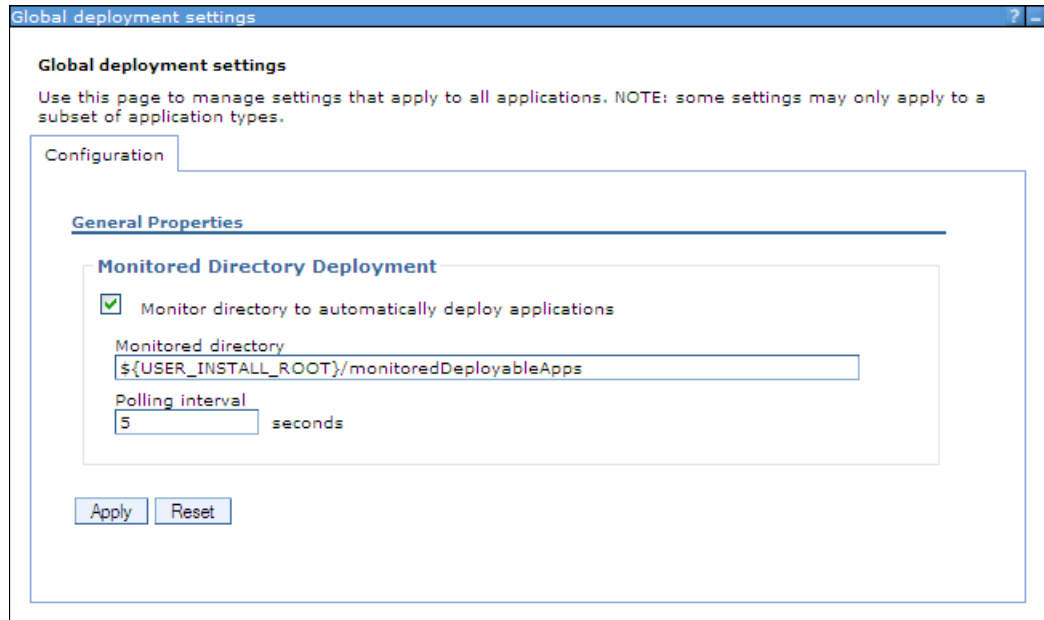


Figure 23-34 Configuring the monitored directory

3. To change the standard polling interval setting, specify a number of seconds in the Polling interval field. Note that the minimum value is 5 seconds. If you specify a lower or negative value, WebSphere Application Server uses 5 seconds automatically.
4. Click **Apply** to save the configuration.
5. Restart the stand-alone application server or the deployment manager.

For more information about configuring the monitored directory using **wsadmin** commands, go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/trun_app_set_dragdrop.html

Working with a monitored directory

When an application is copied into the monitored directory, WebSphere Application Server creates a temporary copy of that application in another directory and installs it on the server. If you copy an application with a name that is already deployed on the server, it will be updated.

The monitored directory works only when the application server or cluster is running.

Binding values: Installing an EAR, JAR, WAR, or SAR file by adding it to a monitored directory does not change existing Java Naming and Directory (JNDI) or other application bindings. If you need to set binding values during deployment, install the file using one of the following methods which set the bindings:

- ▶ The administrative console application installation wizard
- ▶ A **wsadmin** script
- ▶ A properties file

For more information about this topic, refer to “Deploying applications by adding properties files to a monitored directory” on page 865).

The following examples demonstrate how to work with a monitored directory. We use the sample ITSO Bank application RAD8EJBWebEAR.ear as previously mentioned. Refer to 23.2, “Preparing to use the sample application” on page 830 for more information.

Deploying and removing an EAR file on a stand-alone application server

To deploy and remove an EAR file on a stand-alone application server:

1. Ensure that your server is running and that you configured the monitored directory. You might have to restart the application server after enabling monitored directory.
2. Locate the monitored directory, for example, for a profile called AppSrv01, copy your files to the following directory:

```
install_root/profiles/AppSrv01/monitoredDeployableApps/servers/server1
```

install_root is the directory where you installed the WebSphere Application Server product.

3. Copy the RAD8EJBWebEAR.ear application to the monitored directory.
4. Verify that the application was successfully deployed and started in the SystemOut.log file. Example 23-3 lists the key information that is logged during the application deployment.

Example 23-3 Installation application log when using a monitored directory

```
[6/22/12 13:23:22:308 EDT] 00000034 WatchService I   CWLDD0007I: Event id
320986637-1. Start of processing. Event type: Added, File path:
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/monitoredDeployableApps/servers/
server1/RAD8EJBWebEAR.ear.
[6/22/12 13:23:22:336 EDT] 00000034 ModelMgr      I   WSVR0801I: Initializing
all server configuration models
...
[6/22/12 13:23:32:490 EDT] 00000034 AppManagement I   CWLDD0015I: Event id
320986637-1. Application RAD8EJBWebEAR is installed successfully.
[6/23/12 13:23:32:491 EDT] 00000034 AppManagement I   CWLDD0020I: Event id
320986637-1. Starting application RAD8EJBWebEAR...
...
[6/22/12 13:23:38:195 EDT] 00000034 CompositionUn A  WSVR0191I: Composition
unit WebSphere:cuname=RAD8EJBWebEAR in BLA WebSphere:blaname=RAD8EJBWebEAR
started.
```

You can also verify that the application is available in the administrative console from the **Application** → **Application Types** → **WebSphere enterprise application** view.

5. Invoke the application using a URL in a browser. For the ITSO Bank application, use the following URL:

```
http://www.itsobank.ibm.com:9080/index.jsp
```

A page similar to that shown in Figure 23-35 on page 864 displays in the browser.



Figure 23-35 ITSO Bank application output

After you deploy the application, note that the application still exists in the monitored directory. If you overwrite this file with a new file, the application is updated with your changes. However, the file must have the same name.

To uninstall the application from the server using the monitored directory, remove the RAD8EJBWebEAR.ear file from the monitored directory. This removal triggers the monitoring service, and the application is uninstalled. Inspect the SystemOut.log file for information about the application status.

Deploying an EAR file on a federated node

To install and remove an application on a federated node, you can use the same procedure as we described previously in “Deploying and removing an EAR file on a stand-alone application server” on page 863. However, you use the deployment manager monitored directory instead of the stand-alone server monitored directory.

After configuring the monitored directory in deployment manager, copy the file to one of the following directories:

- ▶ The `dmgr_profile_root/monitoredDeployableApps/servers/server1` directory to deploy the application on all servers named server1
- ▶ The `dmgr_profile_root/monitoredDeployableApps/nodes/node01/servers/server1` directory to deploy the application only on server1 on node01.

Deploying an EAR file on a cluster

To install an application on a cluster, use the cluster directory in the deployment manager monitored directory. For example, if your cluster name is `cluster01`, to deploy an application to this cluster, copy the application to the following directory:

```
dmgr_profile_root/monitoredDeployableApps/clusters/cluster01
```

To update the application, overwrite it with a new EAR file with the same name. To remove the application from cluster, delete it from the monitored directory.

Note that if a server is a member of a cluster, you cannot deploy an application by copying it to its directory. If you do, you receive the following error:

```
A server named member1 was targeted by a monitored directory application deployment operation but was skipped because it is a member of cluster cluster01
```


Deploying applications by adding properties files to a monitored directory

Installing applications by copying them to a monitored directory does not change any bindings or JNDI configurations that the applications might use. To configure application resources and deploy an application using a monitored directory, use the properties file.

The monitoring service scans both the server directory and the property directory. If it finds a valid property file, it runs the `wsadmin applyConfigProperties` command.

As an example, to install an application on a cluster using a property file, complete the following steps:

1. Create the property file that defines the deployment task that you want to complete. In this example, we install the ITS0 Bank application on a cluster using the file shown in Example 23-4.

Example 23-4 Example of property file that installs ITS0 Bank application

```
#Header
ResourceType=Application
ImplementingResourceType=Application

# Properties
Name=ITS0Bank Application
EarFileLocation=/apps/RAD8EJBWebEAR.ear
TargetNode=was85node02
TargetCluster=cluster01

EnvironmentVariablesSection

#Environment Variables
cellName=was85Cell01
```

2. Ensure that the target server or cluster is running and that the monitored service is enabled and configured.
3. Copy the properties file to the `deploymentProperties` directory in the monitored directory.

WebSphere Application Server reads the property file and executes its tasks against its runtime environment. It then installs and configures the application.

Note that if you remove the property file from the monitored directory, the monitoring service notices this event but will not uninstall the application. Example 23-5 shows information that is logged when removing a property file from a monitored directory.

Example 23-5 Monitor service processing logs

```
[6/22/12 13:52:52:957 EDT] 00000053 WatchService I CWLDD0007I: Event id
818406234-3. Start of processing. Event type: Deleted, File path:
/opt/IBM/WebSphere/AppServer/profiles/Dmgr01/monitoredDeployableApps/deploymentPro
perties/install_RAD8EJBWebEAR.txt.
[6/22/12 13:52:52:959 EDT] 00000053 WatchService I CWLDD0061I: Event id
818406234-3. The property file
/opt/IBM/WebSphere/AppServer/profiles/Dmgr01/monitoredDeployableApps/deploymentPro
properties/install_RAD8EJBWebEAR.txt is deleted from the folder. No action is
performed.
```

[6/22/12 13:52:53:002 EDT] 00000053 WatchService I CWLDD0008I: Event id 818406234-3. End of processing.

To update an application, you can copy the property file and overwrite an existing property file (if not deleted). This action triggers the monitoring service to reinstall the application.

You have to uninstall an application manually or prepare a property file with uninstallation instructions. For the JSFSample application, you can use the file shown in Example 23-6.

Example 23-6 Example of property file that uninstalls the JSFSample application

```
#Header
ResourceType=Application
ImplementingResourceType=Application
DELETE=true

# Properties
Name=ITSOBank Application
```

To learn more about options and capabilities of using property files, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/trun_app_install_dragdrop_prop.html

23.7.3 Deploying applications using the job manager

In a flexible management environment, you can submit the Install application job to deploy an enterprise application on managed application server targets of the job manager. This is useful if you want, for example, to deploy applications in servers located in different cells or different data centers.

Note: When you use the job manager to install an application, the only options you can specify from the job manager console is the location to install. All the other options that you see in the administrative console are the defaults.

The first step is to make the application available to the system where the installation job will run. Complete the following steps to transfer the application binaries from the job manager to the remote node using the **distributeFile** job:

1. Copy the EAR file to the `jmgr_profile_root/config/temp/JobManager` directory.
2. In the Job manager console, click **Jobs** → **Submit**. This action launches the Job properties wizard.
3. Select the **Distribute file** job type, and click **Next**.
4. Select the deployment manager or administrative agent as the target node and enter the target credentials.
5. Enter the EAR file location on the job manager and the location to store the EAR file on the remote node. The source file URL is relative to the job manager profile's `config/temp/JobManager` directory. The destination parameter gives the location relative to the default distribution provider's directory tree for downloaded content. The arguments are entered as shown in Figure 23-36 on page 867.

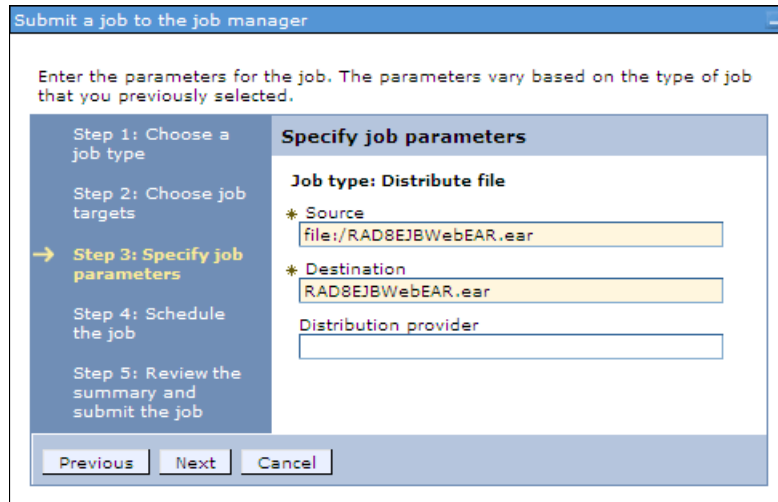


Figure 23-36 Specify the location of the source and target file

Using these arguments, the script file is distributed to the following location:

`dmgr_profile_root/downloadedContent/RAD8EJBWebEAR.ear`

Click **Next**.

6. Use the defaults for the job schedule. The defaults execute the distribute file job one time. Click **Next**.
7. Review the summary, and click **Finish**. Monitor the status of the job and ensure it completes successfully.

To install the application from the job manager:

1. Click **Jobs** → **Submit**.
2. Select the **Install application** job type, and click **Next**.
3. Select the target node where the job will run.
Enter the user ID and password with administrative authority on the target node.
4. Specify the job parameters.
At a minimum, complete the following actions:
 - Specify the name of the new application.
 - If the target is a deployment manager, enter the name of the node and server or cluster on which the application will be installed (Figure 23-37 on page 868).

Figure 23-37 Specify the options for application install

5. Use the defaults for the job schedule. The defaults execute the job one time. Click **Next**.
6. Review the summary, and click **Finish**. Monitor the status of the job and ensure it completes successfully.

23.8 Deploying business-level applications

A *business-level application* (BLA) is a concept that aims to expand the notion of an application beyond Java EE. Its administration model provides the entire definition of an application because it makes sense to the business. In contrast with an enterprise application (EAR file), a business-level application is only a logical WebSphere configuration artifact, similar to a server or cluster, that is stored in the configuration repository.

In this section, we introduce business-level applications and show how you can deploy a Java EE application by creating a business-level application and adding the Java EE application to it as an asset.

Support for business-level applications: Business-level applications are supported only on WebSphere Application Server V7 or later nodes.

Figure 23-38 on page 869 shows the concept of business-level applications.

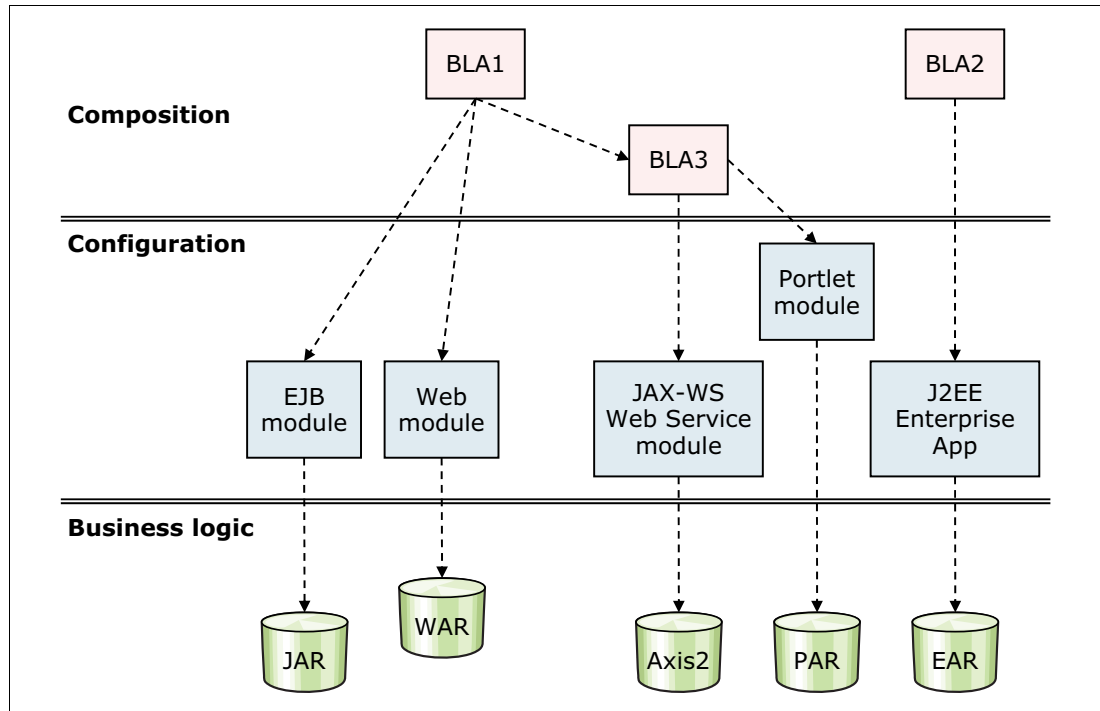


Figure 23-38 Business-level application concept

Business-level applications can be used in several different ways. Often a business application, such as an Order System, does not consist of only one enterprise application (EAR). That system can have multiple applications that must all be running for the whole business application to work.

One way of using business-level applications is to group the separate enterprise applications that make up the business application into one manageable unit that can be started, stopped, updated, and so on. However, a business-level application can reference Java EE components and assets that are not part of the Java EE concept. An example of this concept is CORBA (C++) executables that are hosted in a generic server or as files on the file system that are not managed by WebSphere but that are required by the application.

A business-level application does not represent or contain application binary files. Instead, it is a configuration that lists one or more composition units that represent the application binary files. A business-level application uses the binary files to run the application business logic. Administration of binary files is done using the normal methods for managing modules (for example, web or EJB modules) and is separate from administration of the application definition.

A business-level application does not introduce any new programming, run time, or packaging models and the following concerns are mitigated:

- ▶ You do not need to change your application business logic. The business-level application function does not introduce new application programming interfaces (APIs).
- ▶ You do not need to change your application runtime settings. WebSphere supports all of the runtime characteristics, such as security, class loading, and isolation, required by individual programming models to which business components are written.
- ▶ You do not need to change your application packaging. There is no specific unique packaging model that provides a business-level application definition.

The terminology for business-level applications introduces the following two new terms:

- ▶ An *asset* represents one or more application binary files that are stored in an asset repository. Typical assets include application business logic, such as EAR files, EJB modules, web modules, service component architecture (SCA) modules, shared library files, static content, and other resource files. The asset repository is managed by WebSphere Application Server and does not require any third-party software.

You must register files as assets before you can add them to one or more business-level applications. At the time of asset registration, you can import the physical application files into WebSphere's configuration repository, or you can specify an external location where the asset resides.

- ▶ A *composition unit* represents a configured asset in a business-level application. Configured in this context means installed, so a configured web module means a web module that is installed.

WebSphere Application Server handles the following types of composition units:

- ▶ Asset composition units
Composition units created from assets by configuring each deployable unit of the asset to run on deployment targets.
- ▶ Shared library composition units
Composition units created from JAR-based assets by ignoring all the deployable objects from the asset and treating the asset JAR file as a library of classes.
- ▶ Business-level application composition units
Composition units created from business-level applications that are added to existing business-level applications.

Figure 23-39 on page 871 shows the relationship between assets, composition units, and business-level applications.

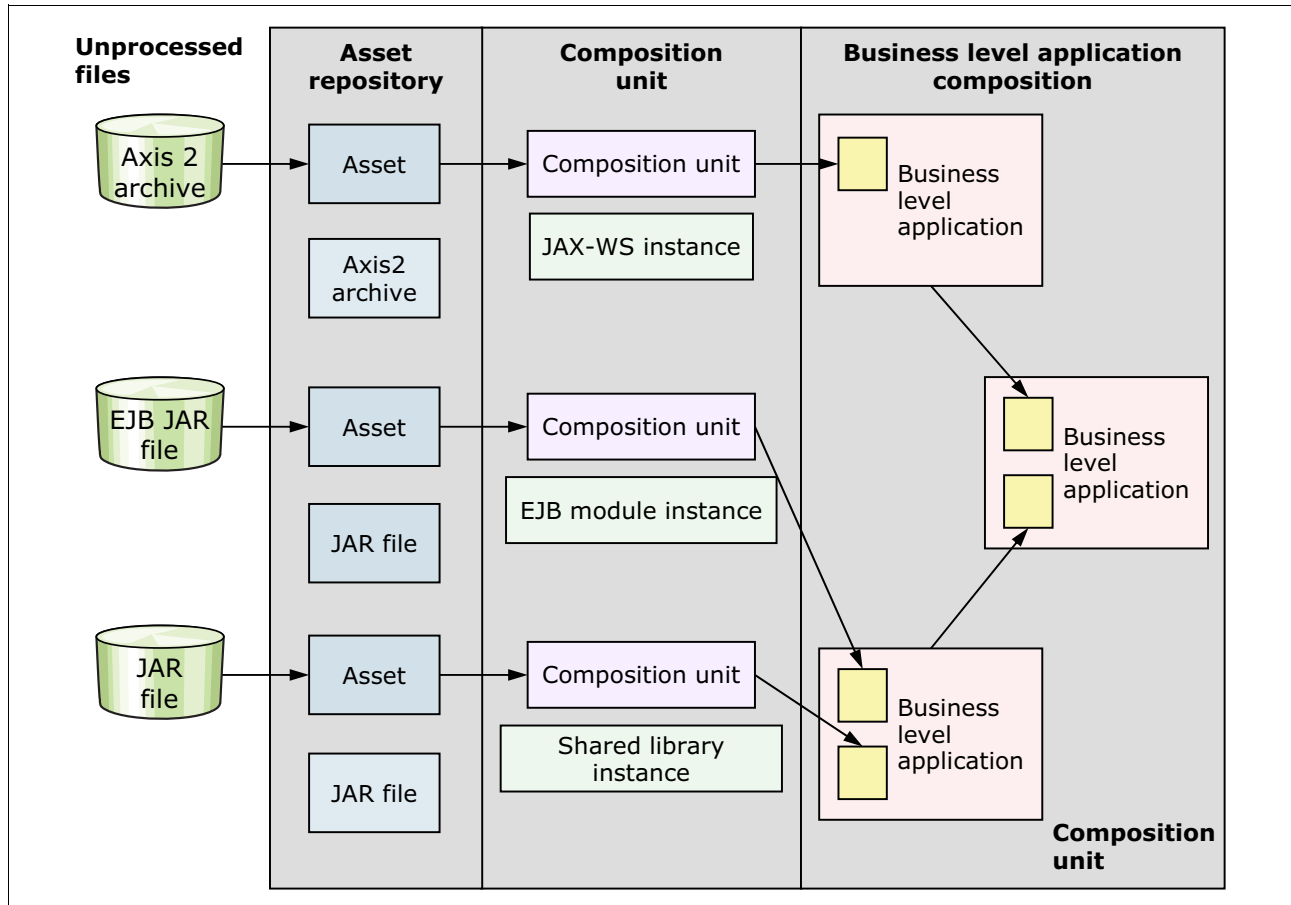


Figure 23-39 Relationship between business-level application artifacts

If an asset depends on another asset, you can create a relationship between them to allow, for example, a web module asset to reference classes in a shared library.

In summary, a business-level application consists of composition units. When you add an asset to a business-level application, a composition unit is created for the asset. The composition unit is the configured (installed) asset.

23.8.1 Creating a business-level application

As a basic example of how to create a business-level application, we use the RAD8EJBWeb application (see 23.2, “Preparing to use the sample application” on page 830) and combine it with the WebSphere DefaultApplication’s web module. An example from the DefaultApplication’s web module that can be combined is the SnoopServlet that can be extracted from the DefaultApplication’s EAR file. We create one asset for the RAD8EJBWeb application and another for the DefaultWebApplication. We then create a business-level application containing these two assets.

When you create a business-level application and assets, those assets are deployed as part of the process. The configuration of the runtime environment and the resources that the deployed assets will need is to be done first. This example assumes that the configuration described in 23.6, “Preparing the runtime environment for the application” on page 846 was completed.

To create the two assets, complete the following steps:

1. Open the WebSphere administrative console, and click **Applications** → **Application Types**. Click the **Assets** link.
2. Click **Import**. Select the **Local file system** option and then click **Browse** to locate the RAD8EJBWebEAR.ear file. Select the file, and click **Open**. Click **Next**.
3. Select the options for importing an asset. Enter a brief description of the asset, if wanted. If you want to import the asset to a specific path on your file system, enter the path in the Asset binaries destination URL field. If you leave this field blank, the file is imported to its default location, which is *profile_root/installedAssets/asset_name/BASE/*. Click **Next**.

Tip: If specifying another name for the asset, make sure to keep the asset's file extension (such as .ear or .war). Otherwise, WebSphere cannot keep track of the asset type and fails to import the asset.

4. On the Summary window, click **Finish**. The asset is now imported into WebSphere's asset repository, but it is not yet configured (thus, it is still just an asset and not a composition unit).
5. Repeat steps 1 to 4 and import the DefaultWebApplication.war file. This file was extracted from the DefaultApplication EAR file and stored on the local operating system. You can find this EAR in the `${WAS_INSTALL_ROOT}/installableApps` path.

Because the two assets do not depend on each other and do not require any shared library, you do not need to set up any relationships.

6. Select **Save to master configuration** when finished.
After the assets are imported into the asset repository, you can create a business-level application.
7. Click **Applications** → **Application Types** → **Business-level applications**. Click **New**.
8. Enter a name, such as ITS0Bank System, and click **Apply**.
9. On the Business-level applications page, click **Add** under the Deployed assets section, and select the **Add Asset** option, as shown in Figure 23-40 on page 873.

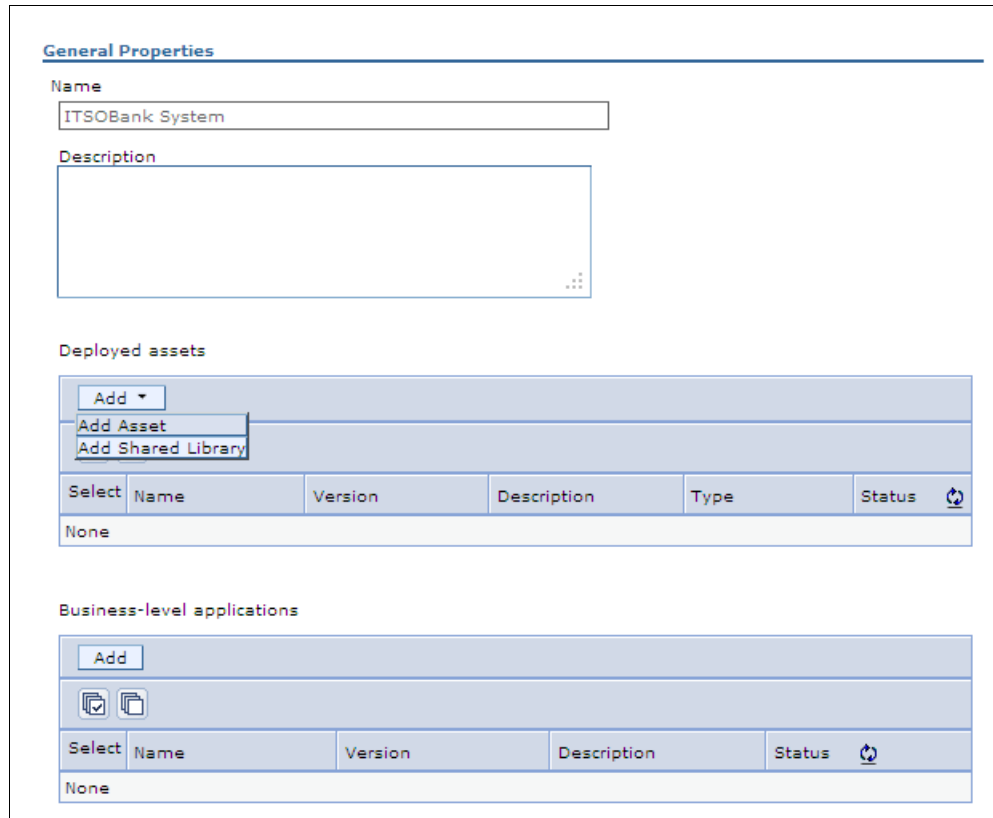


Figure 23-40 Business-level application configuration window

10. On the next window, select the **RAD8EJBWebEAR.ear** asset, and click **Continue**.
11. Configure (install) this asset with the necessary deployment options. The installation process follows the same steps for installing an application to WebSphere Application Server.

For detailed step-by-step instructions, refer to 23.7, “Deploying the application” on page 855. Proceed through the installation windows until the Summary window opens, and click **Finish**. WebSphere now installs the application, and it is a composition unit, that is, an asset that has been configured.

Installation note: In Step 3 on page 872 of the installation process, select installation options, WebSphere generates a unique application name, such as app1143018803114601914, for the application. You might want to change this name to a more descriptive name.

12. Click **Applications** → **Application Types** → **Business-level applications**, and click the link for the **ITSOBank System** application.
13. Repeat steps 9 on page 872 and 10, and add the DefaultWebApplication.war file to the ITSOBank System business-level application as well.
14. Save to the master configuration when done.
15. After completing the steps, you can stop the business-level application by clicking **Applications** → **Application Types** → **Business-level applications** and then clicking the corresponding links.

Deleting a business-level application: To delete a business-level application, you must first unmap (delete) the composition units (configured assets) that belong to the business-level application. Select the business-level application, and select the deployed assets. Click **Delete**. After you delete all assets from the business-level application, you can delete the business-level application itself. The assets, however, still remain in WebSphere's asset repository and can be used to configure other business-level applications.

You can manage the individual applications (composition units) that make up the business-level application individually. To do so, click **Applications** → **Application Types** → **WebSphere enterprise applications**, and use the links to start, stop, update, and so on.

23.9 Deploying application clients

To run a Java-based client/server application, the client application executes in a client container of some kind. You might, for example, use a graphical swing application that calls EJB beans on an application server. WebSphere Application Server V8.5 supports several different application client environments. The following application client environments are available:

► Java EE client

This client uses services that are provided by the Java EE client container.

This client is a Java application program that accesses EJB beans, JDBC databases, and JMS queues. The Java EE application client depends on the application client run time to configure its execution environment, and it uses the JNDI name space to access resources, the same as you do in a server application (similar to a servlet).

The Java EE application client provides the following components:

- XML deployment descriptors
- Java EE naming (`java:comp/env`), including EJB references and resource references

The Java EE application client is launched using the **launchClient** script, which sets up the environment with the necessary class paths and other settings for you.

► Java thin client

This client does not use services that are provided by the Java EE client container.

The Java thin client runtime environment provides the support needed by full-function Java SE client applications but does not support a client c that provides easy access to these services. The Java thin client is designed to support those users who want a full-function Java SE client application programming environment using the supplied IBM JRE, without the impact of the Java Platform, Enterprise Edition (Java EE) platform on the client machine. The Java thin client does not perform initialization of any of the services that the client application might require. For example, the client application is responsible for the initialization of the naming service, either through CosNaming or JNDI APIs. The Java thin client runtime environment does provide support for Java SE client applications to access remote enterprise beans, and provides the implementation for various enterprise bean services. Client applications can also use the Java thin client runtime environment to access CORBA objects and CORBA based services.

The thin client supports JVMs from IBM, Sun and HP-UX. When launching the thin application client, you must set up the correct class paths yourself and make sure that the required libraries for your application and the WebSphere libraries are included.

- ▶ Applet client

In the Applet client model, a Java applet that is embedded in an HTML document executes in a web browser. With this type of client, the user accesses an enterprise bean in the application server through the Java applet in the HTML document.

- ▶ ActiveX to EJB Bridge application client

The ActiveX application client allows ActiveX programs to access enterprise beans through a set of ActiveX automation objects. The ActiveX application client uses the Java Native Interface (JNI) architecture to programmatically access the Java virtual machine (JVM) API. Therefore, the JVM code exists in the same process space as the ActiveX application (Visual Basic, VBScript, or Active Server Pages files) and remains attached to the process until that process terminates. The ActiveX to EJB Bridge is supported on Windows systems only.

For detailed capabilities of each client container, search the information center for *Client Applications*, or go to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.base.doc/ae/ccli_clientapps.html

23.9.1 Installing application client environments

You can install the application client environments using IBM Installation Manager and the WebSphere Application Server supplements package files. The installation packages contains the following installable components:

- ▶ IBM WebSphere SDK for Java Technology Edition 6.
- ▶ Standalone thin clients, resource adapters, and embeddable containers.
- ▶ Samples for both Java EE and non-Java EE client environments.

Refer to 2.6.2, “Application Client for WebSphere Application Server V8.5” on page 56 for detailed information about installing the application client environments.

23.9.2 Preparing the sample application

The ITSO Bank application also has a stand-alone application client that we use to demonstrate the WebSphere Application Server application client container.

We need to import the project that contains the stand-alone application client into IBM Rational Application Developer for WebSphere Software and export it as an EAR file. This process is similar to the one covered in detail in 23.2, “Preparing to use the sample application” on page 830.

Complete the following steps to import the project and the stand-alone application:

1. Import the `j2eeclient\RAD8AppClient.zip` Project Interchange file from the `7835codesolution` folder. Select both projects when importing.
2. Because the `RAD8AppClient` project has a specific binding that is configured for the `EJBBankBean` in the `RAD8EJBPEAR` file, you need to modify this binding to point to the same bean in the `RAD8JBWebEAR` file, which is the application that you installed on your server. To modify the binding complete the following steps:
 - a. Expand the **RAD8AppClient** project, and double-click the **RAD8AppClient** deployment descriptor.
 - b. Click the **Open WebSphere Bindings Descriptor** link from the right pane.

- c. Select **EJB Reference (ejb/bank)**, and change the name from `ejb/RAD8EJBEAR/RAD8EJB.jar/EJBBankBean#itso.bank.service.EJBBankRemote` to `itso.bank.service.EJBBankRemote`.
By using the short name, we rely on the Autolink feature to search for and invoke a matching EJB interface. This method works in our environment where we only have one instance of the EJB installed and running.
 - d. Press Ctrl+s to save the deployment descriptor.
3. Export the `RAD8AppClientEAR` as described in 23.5, “Exporting an application project to an EAR file” on page 845 to `/apps/RAD8AppClientEAR_withModifiedBinding.ear`.

23.9.3 Launching the J2EE client

A Java EE client application needs a container in which to run. In this example, we use the Java EE application client container. You can start this container using the `launchClient` program in the `install_root/bin` directory. By default the install root is `/opt/IBM/WebSphere/AppClient` in Linux systems. The `launchClient` program has the following syntax:

```
Usage: launchClient [-profileName pName | -JVMOptions options | -help | -?]
<userapp.ear> [-CC<name>=<value>] [app args]
```

The following list describes the elements that follow the `launchClient` command in the previous syntax:

| | |
|----------------------------|---|
| -profileName | Defines the profile of the application server process in a multi-profile installation. The -profileName option is not required for running in a single profile environment or in an application client installation. The default is default_profile . |
| -JVMOptions | Specifies a valid Java standard or nonstandard option string. Insert quotation marks around the option string. |
| -help, -? | Prints the usage information. |
| <userapp.ear> | Enter the path name and file name of the <code>.ear</code> file that contains the client application. |
| -CC | These properties are for use by the application client run time. Numerous parameters are available. For full explanation of all parameters, run launchClient -help . |
| app args | The application arguments are for use by the client application and are ignored by WebSphere. |

To start the ITSO Bank stand-alone application client using the `launchClient` command:

1. Execute the command shown in Example 23-7.

Example 23-7 Launching the ITSO Bank stand-alone application client

```
/opt/IBM/WebSphere/AppClient/bin> ./launchClient.sh
/apps/RAD8AppClientEAR_withModifiedBinding.ear
```

```
IBM WebSphere Application Server, Release 8.5
Java EE Application Client Tool
Copyright IBM Corp., 1997-2011
WSCL0012I: Processing command line arguments.
WSCL0013I: Initializing the Java EE Application Client Environment.
```

```
[2012-06-22 17:24:05:421 CEST] 00000001 W UOW=null
source=com.ibm.ws.ssl.config.SSLConfig org=IBM prod=WebSphere
component=Application Server thread=[P=916627:
0=0:CT]
```

```
CWPKI0041W: One or more key stores are using the default password.
WSCL0035I: Initialization of the Java EE Application Client Environment has
completed.
WSCL0014I: Invoking the Application Client class
itso.rad8.client.control.BankDesktopController
```

2. The first time you execute this process shown in Example 23-7 on page 876, you will receive a message to accept the SSL certificate, as shown in Figure 23-41. Click **y**.

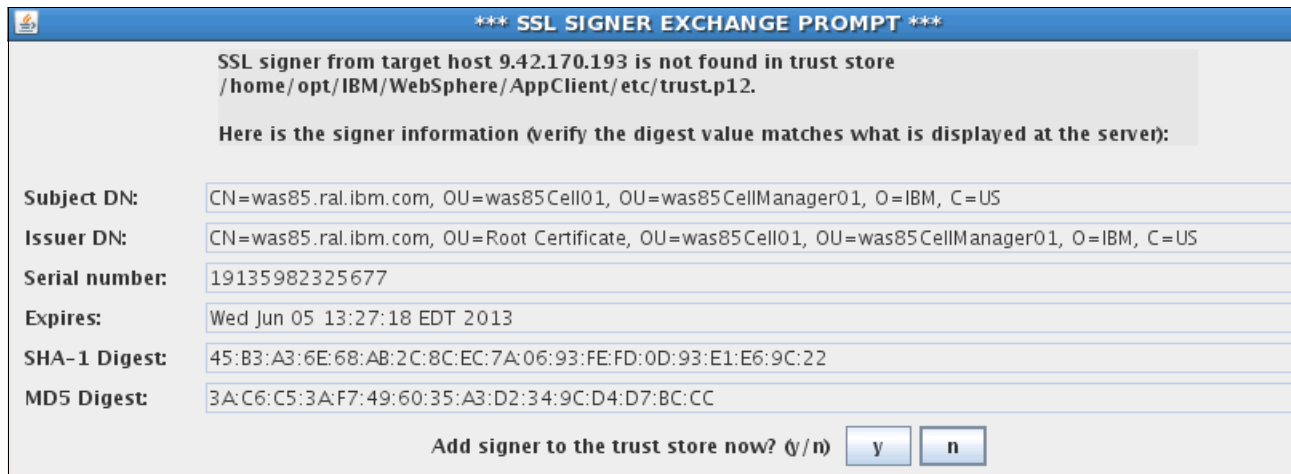


Figure 23-41 SSL certificate window message

3. If you have administrative security enabled, you have to provide the WebSphere Application Server credentials as in Figure 23-42.

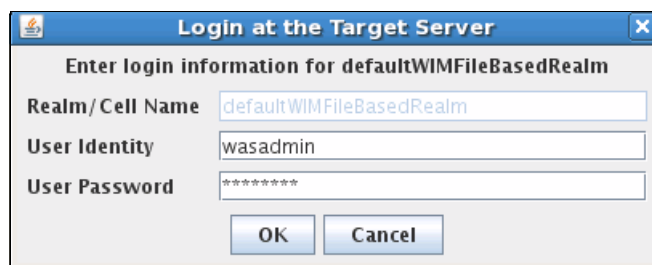


Figure 23-42 Login information window

The application opens and displays a graphical window. Enter 111-11-1111 as the social security number. The results look like Figure 23-43 on page 878.

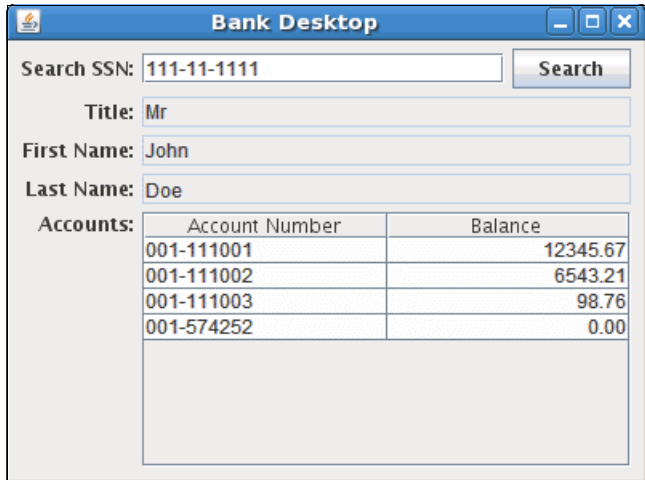


Figure 23-43 Running the ITSO Bank stand-alone application client



Updating Java EE applications

WebSphere Application Server has features that allow applications to be updated and restarted at a fine-grained level. It is possible to update only parts of an application or module and to restart only the necessary parts. It can also do interruption-free application upgrades by using editions. With these features, you can perform the following tasks:

- ▶ Replace an entire application (.ear file).
- ▶ Replace, add, or remove a single module (.war, EJB .jar, or connector .rar file).
- ▶ Replace, add, or remove a single file.
- ▶ Replace, add, and remove multiple files by uploading a compressed file that describes the actions to take.
- ▶ Do an interruption-free application upgrade.

In this chapter, we explain these features within the following topics:

- ▶ Working with applications
- ▶ Replacing an entire application EAR file
- ▶ Replacing or adding an application module
- ▶ Application edition management and rollout
- ▶ Hot deployment and dynamic reloading

24.1 Working with applications

If an application is running when it is updated, WebSphere Application Server stops the application or the affected components automatically, updates the application, and then restarts the application or components.

When updating an application, only the portion of the application code that is changed needs to be presented to the system. The application management logic calculates the minimum actions that the system needs to execute to update the application. Under certain circumstances, the update can occur without stopping any portion of the running application.

WebSphere Application Server also has support for managing applications in a cluster for continuous availability. The Rollout Update action updates sequentially an application that is installed on multiple cluster members across a cluster. After you update an application's files or configuration, use the Rollout Update option to install the application's updated files or configuration on all cluster members of a cluster on which the application is installed.

The Rollout Update option completes the following actions for each cluster member in sequence:

1. Saves the updated application configuration.
2. Stops all cluster members on a given node.
3. Updates the application on the node by synchronizing the configuration.
4. Restarts the stopped cluster members on that node.

This action updates an application on multiple cluster members while providing continuous availability of the application.

With WebSphere Application Server V8.5, you can also use application editions to manage updates to your application, making it easier to rollout or rollback updates. The section 24.4, "Application edition management and rollout" on page 889 details this feature.

WebSphere Application Server V8.5 introduces a feature for validating an application edition before rolling it out. Setting an application edition to a validation state allows the application to be tested in a more realistic environment without impacting the users. We show how to accomplish this in 24.4, "Application edition management and rollout" on page 889.

WebSphere Application Server V8.5 supports an additional OSGi application type that has a modular and dynamic design. WebSphere Application Server V8.5 features additional functionality that is designed to manage the OSGi application lifecycle. We explain this feature in more detail in 26.1.2, "OSGi bundle lifecycle" on page 926.

24.2 Replacing an entire application EAR file

If you are not using application editions to manage application updates, follow these instructions to replace a full EAR file with a newer version:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update and then click **Update**.

2. In the Preparing for the application update window (Figure 24-1), select the **Replace the entire application** option (this is the default option).

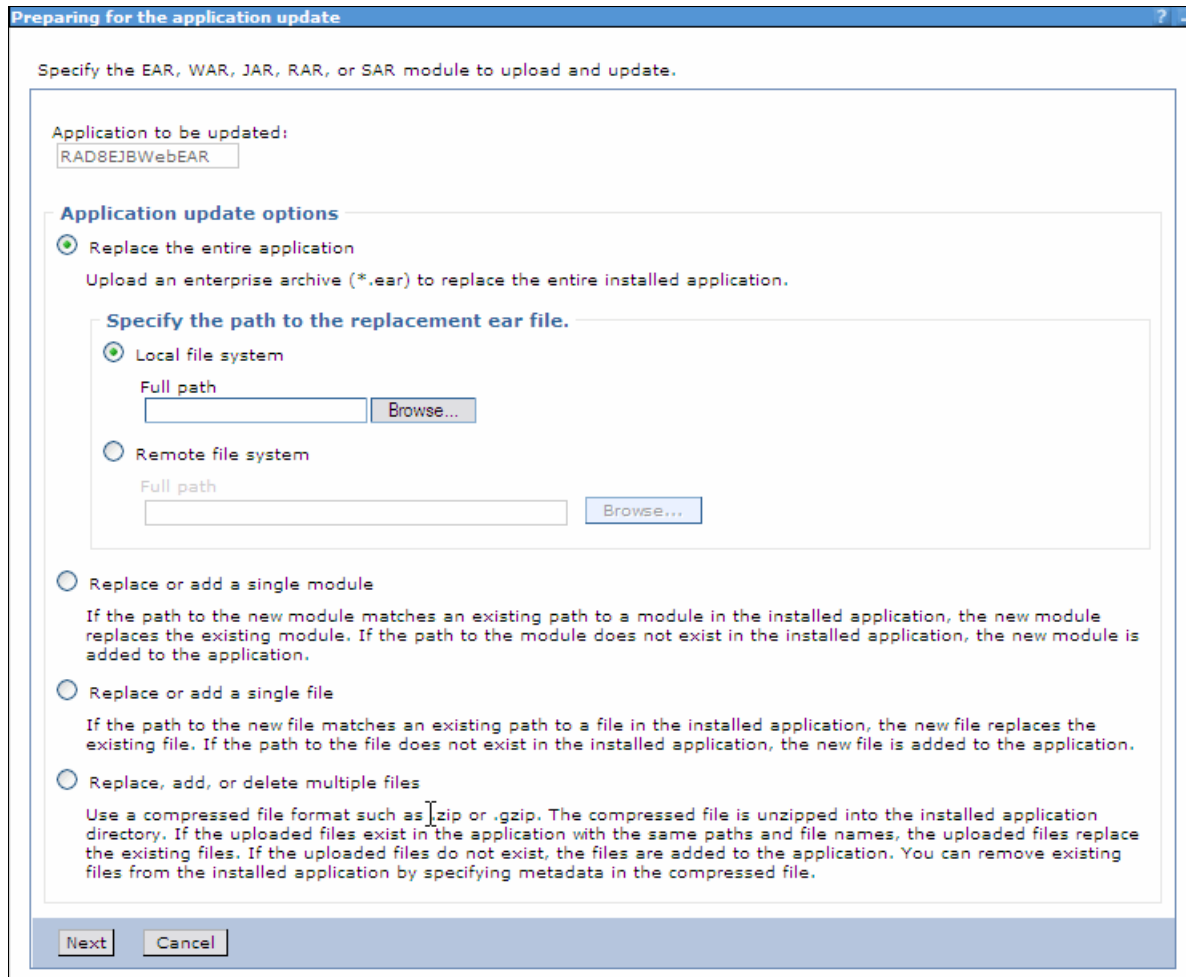


Figure 24-1 Application update options

3. Select either the **Local file system** or **Remote file system** option. Click **Browse** to select the updated EAR file. Click **Next**.
4. Proceed through the application installation windows, and make any necessary changes. The options for an application installation are described in 23.7, “Deploying the application” on page 855. On the Summary window, click **Finish**.
5. To preview the files that are changed after the update installation, click the **Review** link. Figure 24-2 on page 882 shows a changed files list.

The screenshot shows a window titled "Enterprise Applications" with a "Save" dialog. The dialog contains instructions to save workspace changes to the master configuration and a table listing 21 changed documents. The table has two columns: "Changed Items" and "Status".

| Changed Items | Status |
|---|---------|
| cells / aix-target1Cell01 / nodes / aix-target1Node01 / serverindex.xml | Updated |
| cells / aix-target1Cell01 / applications / DefaultApplication.ear / deltas / DefaultApplication / delta-1213634107218 | Deleted |
| cells / aix-target1Cell01 / applications / DefaultApplication.ear / DefaultApplication.ear | Updated |
| cells / aix-target1Cell01 / applications / DefaultApplication.ear / deltas / DefaultApplication / delta-1310495461173 | Added |
| cells / aix-target1Cell01 / applications / DefaultApplication.ear / deployments / DefaultApplication / Increment.jar / META-INF / ibm-ejb-jar-bnd.xml | Updated |
| cells / aix-target1Cell01 / applications / DefaultApplication.ear / deployments / DefaultApplication / META-INF / MANIFEST.MF | Updated |
| cells / aix-target1Cell01 / applications / DefaultApplication.ear / deployments / DefaultApplication / META-INF / application.xml | Updated |
| cells / aix-target1Cell01 / applications / DefaultApplication.ear / deployments / DefaultApplication / Increment.jar / META-INF / ibm-ejb-jar-ext.xml | Updated |

Figure 24-2 Change log of the updated application files

After you review the files, save the configuration by clicking **Save** at the bottom of the window.

6. If you are working in a distributed server environment, make sure that you also synchronize the changes with the nodes.
7. If the application update changes the set of URLs that are handled by the application (that is, if servlet mappings are added, removed, or modified), make sure that the web server plug-in is regenerated and propagated to the web server.

Timing note: It might take a few seconds for the WebSphere run time to pick up the changes and to restart the application as necessary. If your changes do not seem to have effect, wait and try again. You can also look at the SystemOut.log file for the application server to see when it restarts the application.

24.3 Replacing or adding an application module

To replace only a module, such as an EJB or web module of an application, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update and then click **Update**.
2. In the Preparing for the application update window (Figure 24-1 on page 881), select the **Replace or add a single module** option.
3. In the Specify the path beginning with the installed application archive file to the module to be replaced or added field, enter the relative path to the module to replace. For example, if you were to replace the HelloWeb module, enter `Hel1oWeb`. If you enter a path or file that does not exist in the EAR file, the module will be added.
4. Select either the **Local file system** or **Remote file system** option and then click **Browse** to select the updated module.
5. Click **Next**.

6. Proceed through the remaining windows, and make any necessary changes. In the Summary window, click **Finish**.

Note: If you are adding a web module, make sure that you select the detailed installation option. This option allows you to select the correct target server for the module in the Map modules to servers step and to specify a context root for the module.

7. To update the configuration in the master repository, select the **Save** link.
8. If you are working in a distributed server environment, make sure that you also synchronize the changes with the nodes.
9. If the application update changes the set of URLs that are handled by the application (that is, servlet mappings are added, removed, or modified), make sure that the web server plug-in is regenerated and propagated to the web server.

Tip: Modules can also be managed using the Manage Modules page. Click **Applications** → **Application Types** → **WebSphere enterprise applications** and then click the link for the application. Click the **Manage Modules** link in the Modules section. Select the module to modify and then click the **Remove**, **Update**, or **Remove File** buttons.

24.3.1 Replacing or adding single files in an application or module

To replace a single file, such as a GIF image or a properties file in an application or module, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update, and then click **Update**.
2. In the Preparing for the application installation window (Figure 24-1 on page 881), select the **Replace or add a single file** option.
3. In the Relative path to file field, enter the relative path to the file to replace in the EAR file. For example, to replace the logo.gif file in the images directory of the HelloWeb.war web module, enter `HelloWeb.war/images/logo.gif`. If you enter a path or file that does not exist in the EAR file, it is added.
4. Select either the **Local file system** or **Remote file system** option and then click **Browse** to locate the updated file. Click **Next**.
5. In the Updating Application window, click **OK**.
6. To update the configuration in the master repository, select the **Save** link.
7. If you are working in a distributed server environment, make sure that you also synchronize the changes with the nodes.

24.3.2 Removing application content

You can remove files either from an EAR file or from a module in an EAR file, as we describe in the sections that follow.

Removing files from an EAR file

To remove a file from an EAR file:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to remove the file from and then click **Remove File**.
2. In the Select the file to be removed dialog box, select the file to be removed and then click **OK**.
3. Save the configuration.

Removing files from a module

To remove a file from a module:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**, and click the link for the application to which the module belongs.
2. Click the **Manage Modules** link under the Modules section.
3. Select the module from which to remove the file, and click **Remove File**.
4. In the Select the file to be removed dialog box, select the file to be removed and then click **OK**.
5. Save the configuration.

24.3.3 Performing multiple updates to an application or module

Multiple updates to an application and its modules can be packaged in a compressed file, in .zip or .gzip format and uploaded to WebSphere Application Server. The uploaded file is analyzed, and the necessary actions to update the application are taken.

Depending on the contents of the compressed file, this method to update an application can replace files in, add new files to, and delete files from the installed application all in one single administrative action. Each entry in the compressed file is treated as a single file, and the path of the file from the root of the compressed file is treated as the relative path of the file in the installed application. To perform these multiple updates, the following conditions must be true:

- ▶ To replace a file, the file in the compressed file must have the same relative path as the file to be updated in the installed application.
- ▶ To add a new file to the installed application, the file in the compressed file must have a different relative path than the files in the installed application.
- ▶ To remove a file from the installed application, specify metadata in the compressed file using a file named META-INF/ibm-partialapp-delete.props at any archive scope.

The ibm-partialapp-delete.props file must be an ASCII file that lists files to be deleted in that archive with one entry for each line. The entry can contain a string pattern, such as a regular expression that identifies multiple files. The file paths for the files to be deleted must be relative to the archive path that has the META-INF/ibm-partialapp-delete.props file.

- ▶ To delete a file from the EAR file (not a module), include a META-INF/ibm-partialapp-delete.props file in the root of the compressed file. In the .props file, list the files to be deleted. File paths are relative to the root of the EAR file. For example, to delete a file named docs/readme.txt from the root of the HelloApp.ear file, include the docs/readme.txt line in the META-INF/ibm-partialapp-delete.props file in the compressed file.
- ▶ To delete a file from a module in the EAR, include a module_uri/META-INF/ibm-partialapp-delete.props file in the compressed file. The

module_uri part is the name of the module, such as HelloWeb.war. For example, to delete images/logo.gif from the HelloWeb.war module, include the images/logo.gif line in the HelloWeb.war/META-INF/ibm-partialapp-delete.props file in the compressed file.

- ▶ Multiple files can be deleted by specifying each file on its own line in the metadata .props file.

Regular expressions can also be used to target multiple files. For example, to delete all JavaServer Pages (.jsp files) from the HelloWeb.war file, include the line *.jsp in the HelloWeb.war/META-INF/ibm-partialapp-delete.props file. The line uses a regular expression, *.jsp, to identify all .jsp files in the HelloWeb.war module.

As an example, assume we prepared the compressed HelloApp_update.zip file shown in Figure 24-3.

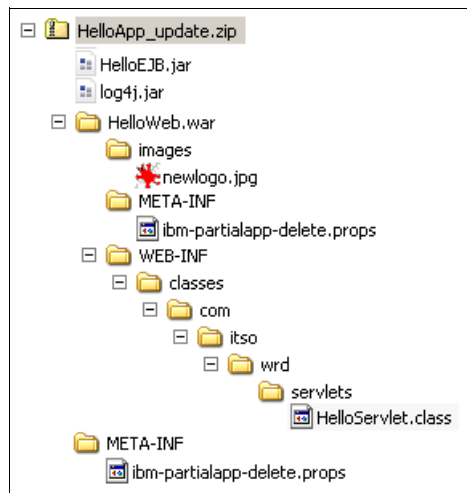


Figure 24-3 HelloApp_update.zip compressed file

The META-INF/ibm-partialapp-delete.props file contains the following line:

```
docs/readme.txt
```

The HelloWeb.war/META-INF/ibm-partialapp-delete.props contains the following line:

```
images/logo.gif
```

When performing the partial application update using the compressed file, WebSphere performs the following actions:

- ▶ Adds the log4j.jar file to the root of the EAR.
- ▶ Updates the entire HelloEJB.jar module.
- ▶ Deletes the docs/readme.txt file (if it exists) from the EAR file but not from any modules.
- ▶ Adds the images/newlogo.jpg file to the HelloWeb.war module.
- ▶ Updates the HelloServlet.class file in the WEB-INF/classes/com/itso/wrd/servlets directory of the HelloWeb.war module.
- ▶ Deletes the images/logo.gif file from the HelloWeb.war module.

To perform the actions specified in the HelloWeb_updated.zip file, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update and then click **Update**.

2. In the Preparing for the application installation window (Figure 24-1 on page 881), select the **Replace, add, or delete multiple files** option.
3. Select either the **Local file system** or **Remote file system** option, and click **Browse** to select the compressed file with the modifications that you created. Click **Next**.
4. In the Updating Application window, click **OK**.
5. To update the configuration in the master repository, select the **Save** link.
6. If in a distributed server environment, make sure the **Synchronize changes with Nodes** option is selected so that the application is distributed to all nodes. Click **Save**. The application is distributed to the nodes, is updated, and is restarted as necessary.
7. If the application update changes the set of URLs that are handled by the application (that is, if servlet mappings are added, removed or modified), make sure that the web server plug-in is regenerated and propagated to the web server.

24.3.4 Rolling out application updates to a cluster

The Rollout Update feature allows you to roll out a new version of an application or part of an application to a cluster. Application updates can be done with a new version of an existing application or a new edition of an existing application edition.

Application version rollout is the traditional way for rolling out application updates, either by adding, removing, or replacing parts or the entire enterprise application. When an existing application is updated, the application is replaced with the new contents and the previously deployed code is lost. The only way to recover the previous version is by making another update using the previous version of the packaged file.

Application edition Management offers a way of managing multiple editions of the same application. This management process makes it possible to change from edition to edition with interruption-free updates, validate a new edition in a live environment before it goes to production, or even have multiple editions of the same application running in different targets. For details about how to do edition rollouts refer to 24.4, “Application edition management and rollout” on page 889.

Note: Interruption-free edition upgrades and edition validation depends on the Intelligent Management features, such as the use of On Demand Router to allow for traffic to be correctly routed.

Application version rollout

The Rollout Update feature allows you to roll out a new version of an application or part of an application to a cluster. The Rollout Update feature stops the cluster members, distributes the new application, synchronizes the configuration, and restarts the cluster members. The operation is done sequentially over all cluster members to keep the application continuously available.

When stopping and starting the cluster members, the Rollout Update feature works at the node level. Thus, all cluster members on a node are stopped, updated and then restarted before the process continues to the next node.

Because the web server plug-in module cannot detect that an individual application on an application server is unavailable, the Rollout Update feature always restarts the application server that is hosting the application. Thus, if HTTP session data is critical to your application,

either persist that data to a database or replicate it to other cluster members using the memory-to-memory replication feature.

The order in which the nodes are processed and the cluster members are restarted is the order in which they are read from the cell configuration repository. The Rollout Update feature cannot process the nodes and cluster members in any particular order.

Assume that we have an environment with two nodes, *aix-target1Node01* and *aix-target2Node01*, and a cluster called *cluster01*, which has one cluster member on each node (member1 on *aix-target1Node01* and member2 on *aix-target2Node01*). Assume also that we have an application named JSFSample that is deployed and running on the cluster.

To update this application using the Rollout Update feature, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**. Select the application to update and then click **Update**.
2. In the Preparing for the application installation window, select the appropriate action, depending on the type of update. In this example, we update the entire application EAR file to a new version. So, select the **Replace the entire application** option.
3. Select either the **Local file system** or **Remote file system** option and then click **Browse** to select the updated EAR file. Click **Next**.
4. Proceed through the remaining windows, and make any changes necessary. In the Summary window, click **Finish**.
5. After the application is updated in the master repository, the status window shown in Figure 24-4 opens.

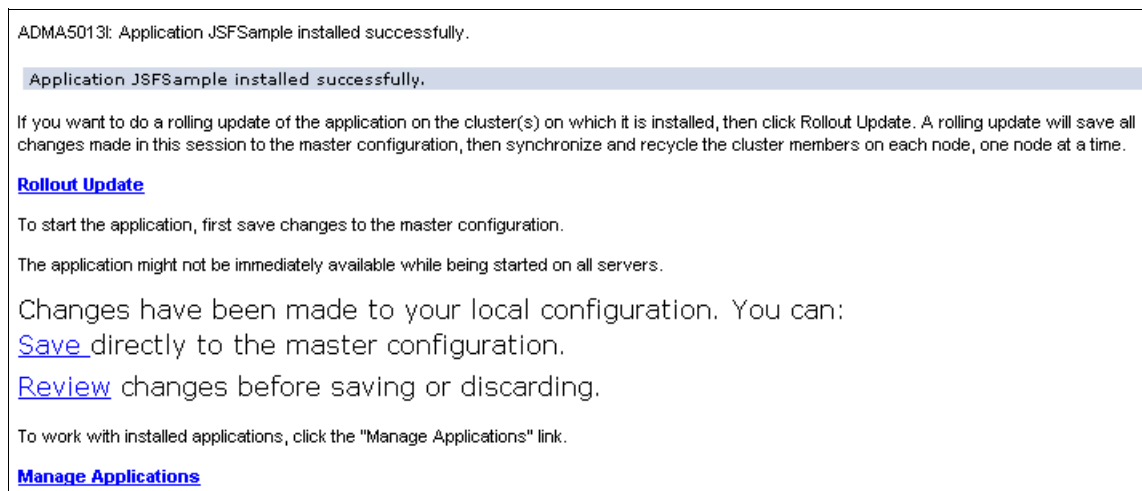


Figure 24-4 Preparing for application rollout

You then have the following options to start the rollout action:

- Click the **Rollout Update** link.
- Click the **Manage Applications** link. In the Enterprise Applications window, select the application, and click **Rollout Update**.

During the rollout, the window in Figure 24-5 opens in the status window.

Saving directly to the master configuration: Do not click the **Save directly to the master configuration** link or otherwise save the configuration yourself. The Rollout Update feature does that task for you. If you save the configuration yourself, the Rollout Update action is canceled, and it is handled as a normal application update.

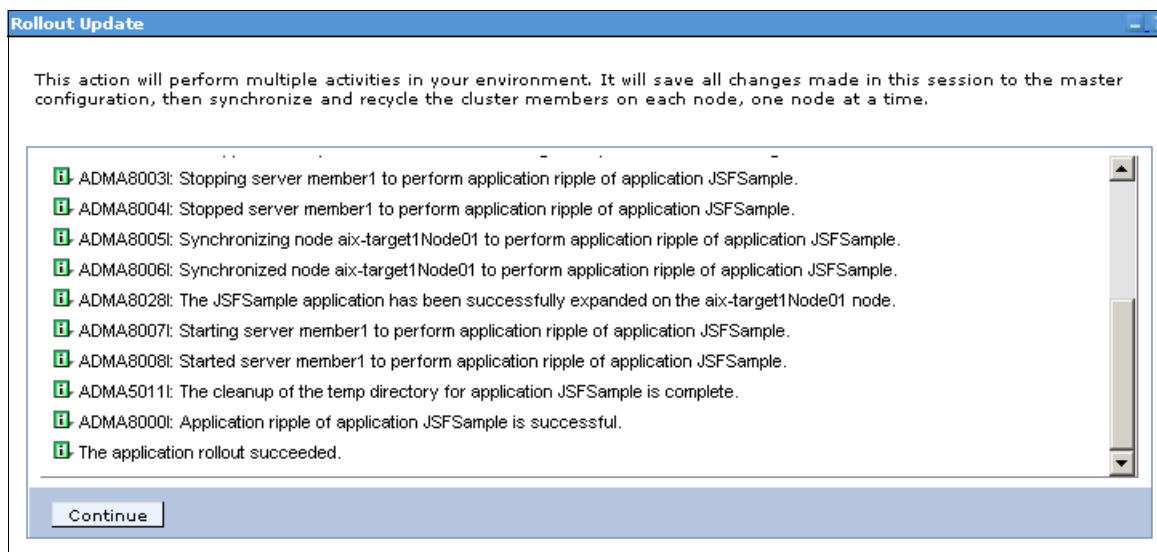


Figure 24-5 Rolling out an application

For each node, the cluster members are stopped, the application is distributed to the node, and the cluster members are restarted. When the rollout is complete (that is, when the The application rollout succeeded message displays), click **Continue**.

6. If the application update changes the set of URLs that are handled by the application (that is, if servlet mappings are added, removed, or modified), make sure that the web server plug-in is regenerated and propagated to the web server.

Automatic synchronization: The automatic file synchronization of the node agent is disabled temporarily during the rollout process and then re-enabled afterwards if it was previously enabled. The Rollout Update feature works regardless of the automatic file synchronization setting. However, in production systems, the automatic synchronization is often disabled to give the administrator greater control over exactly when changes made to the cell configuration are distributed to the nodes.

Although the rollout update feature makes it easy to roll out an application to a cluster and keep the application continuously available, make sure that your application can handle the rollout. For example, assume that you have Version 1.0 of an application running in a cluster. The cluster consists of two application servers named server1 and server2, and HTTP session data is persisted to a database. When you roll out Version 2.0 of the application and server1 is stopped, the web server plug-in redirects users on server1 to server2. Then, when server1 is started again, starting Version 2.0 of the application, the plug-in distributes requests to server1 again. Now, if the application update incurred a change in the interface of any class that is stored in the HTTP session, when server1 tries to get these session objects from the database, it might run into a deserialization or class cast exception, preventing the application from working properly.

Another situation to consider is when the database structure changes between application versions, as when tables or column names change name or content. In that case, you might need to stop the entire application and migrate the database before you can deploy the new version. The Rollout Update feature is not suitable in this scenario.

So, it is important to understand the changes made to your application before rolling it out.

24.4 Application edition management and rollout

WebSphere Application Server V8.5 introduces the Edition control center, which enables management and operational control over application editions, including interruption-free application upgrades. An application edition is a version of an application composed of distinct versions of modules and bindings.

The terms version and edition distinguish between what occurs in your development and build environment from what occurs in your deployment and operational environment. A version is a successive generation of an interface, function, implementation, or entire application, and so forth. Version is a development and build concept. An edition is a successive deployment generation, for example, the deployment of a particular set of versioned artifacts. Edition is a deployment and operational concept.

The edition control center provides an application versioning model that supports multiple deployments of the same application in the Intelligent Management cell. Each deployment has a unique edition name. The edition control center allows you to select the edition to activate on an Intelligent Management cluster, so that you can perform a rollout of an application update or revert to a previous level.

24.4.1 Installing an application edition

Installing an application edition is similar to installing an application. When installing a new edition of an application, the Application edition and Edition description must be specified in the Select installation options window during the application installation wizard. This automatically creates an edition for the application being deployed (see Figure 24-6 on page 890).

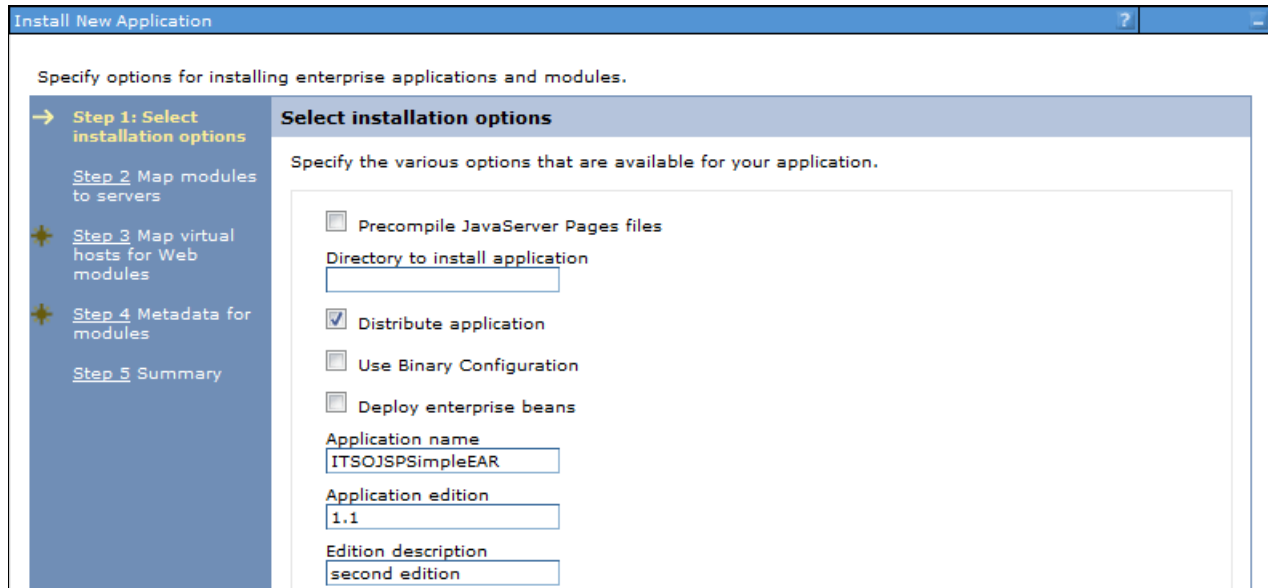


Figure 24-6 Installing an application edition

24.4.2 Activating an edition

Application editions can be in one of three states: Inactive, Validation, or Active. The installation of an edition does not make it automatically active. Activating an edition is different from starting an edition. An application edition must be in active state before it can be started. Similarly, the edition can only be deactivated when it is stopped.

To activate an edition, select **Applications** → **Edition control center** → *application_name* and then select the edition to activate. Click **Activate**.

Concurrent activation

Concurrent activation enables you to activate different editions of the same application on different servers or clusters. To use multiple editions concurrently, it is necessary to distinguish user requests from one another so that the requests can be sent to the application servers that are hosting the appropriate edition. For example, if you introduce a new edition of an application, you might want a select group of users to test the edition and not want all users to have access to the edition. The on demand router policies define the destination of a client request. To have concurrent activation of an edition, you must create a routing policy as explained in 24.4.3, “Creating routing policies for application editions” on page 891.

Note: Only one edition can be active in a single target simultaneously. If you want to have multiple editions of the same application concurrently activated, you must use different deployment targets. You also create the routing policies to allow the on demand router to decide to what edition to send the requests.

Figure 24-7 on page 891 shows an example of concurrent activation.

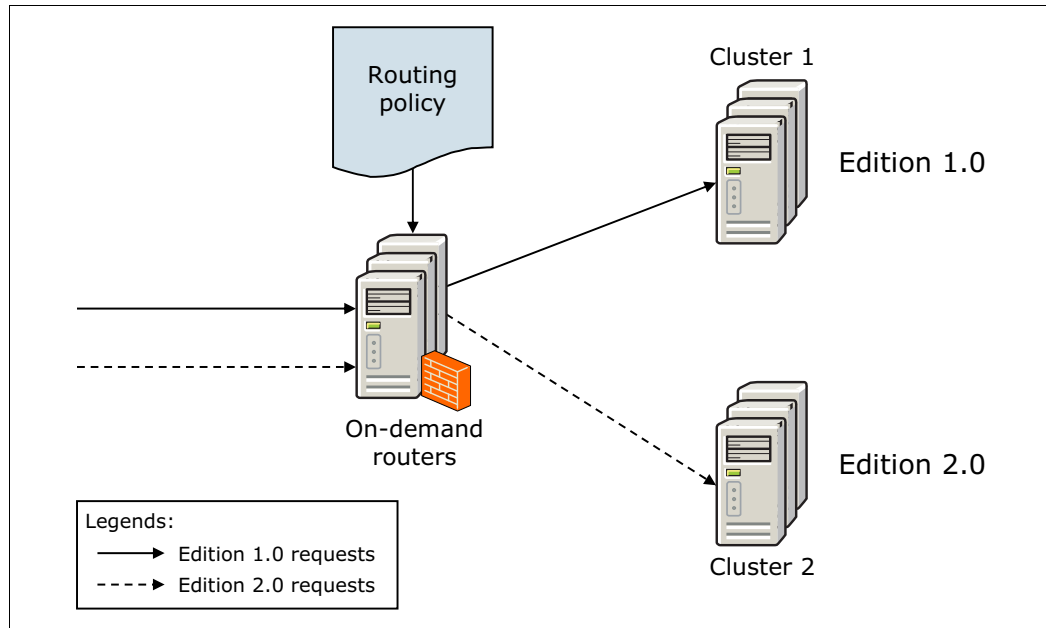


Figure 24-7 Concurrent activation

24.4.3 Creating routing policies for application editions

A routing policy allows you to create rules for the on demand router to decide to what edition to send a client request based on a set of criteria. This action is needed whenever you have multiple editions of the same application either in active/validate state or in active/active state.

Editions of an application usually have the same basic parameters used for routing requests, such as virtual host and context root. The routing policy needs extra information to be able to differentiate the client requests and route them to the correct edition. This information can be the IP address, the user ID or group, the time of the request, or many other parameters. For a listing of other parameters, visit the following information center website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=rwve_odrhttp

To create a routing policy for an edition:

1. From the WebSphere administrative console, go to **Applications** → **All Applications**.
2. Select the application for which you are creating the policy. Notice that the policy is created per application and not per edition, so your change affects all editions for the same application.
3. Go to the **Routing Policies** tab.
4. Expand **Work classes for HTTP requests** and then expand **Default_HTTP_WC**. This is the default work class for this application.
5. Add a new rule by clicking **Add Rule**.
6. Click **Build subexpression** to assist you in the definition of the rule. Figure 24-8 on page 892 shows an example of where the client IP address was used.

[All Applications](#) > [ITSOJSPSimpleEAR-edition1.1](#) > [New...](#)

Build a subexpression to copy and paste into the rule editor.

Select operand:

Operator:

Value:

Generated subexpression

Subexpression:

Figure 24-8 Subexpression builder

7. Copy and Paste the subexpression into the rule text box, and select the appropriate action to take when it matches. In this example, we selected the **Permit routing with affinity to** option. The other available options are listed in the following information center website.
http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=rwve_odrworkclass
8. Select the appropriate application edition to which to route the traffic. The window looks similar to Figure 24-9.

| Select | Order | Routing rule |
|--------------------------|-------|--|
| <input type="checkbox"/> | 1 | <p>Edit rule</p> <p>[<input type="button" value="Build subexpression"/>]</p> <p>If</p> <p><input type="text" value="clientipv4 = '192.168.64.1'"/></p> <p>Then <input type="text" value="Permit routing with affinity to"/></p> <p>Select edition name <input type="text" value="ITSOJSPSimpleEAR-edition1.1"/></p> <p><input type="button" value="Validate Rule"/> <input type="button" value="Cancel"/></p> |

Figure 24-9 Creating an edition routing rule

9. Click **Validate rule**.
10. Click **OK** and then **Save**.
11. Synchronize the nodes. The router policy is picked up automatically by the on demand router after the synchronization finishes.
12. Make sure you close and open all browser windows before doing the test.

24.4.4 Validating an edition

Validating an application in a realistic way or in a production environment is usually a complicated process. It can demand creating and setting up a whole new environment just for the sake of validation and still incur the risk of not having this validation environment synchronized with the production environment.

WebSphere Application Server V8.5 offers a simplified way of doing this task. By setting an application edition to validation state, it creates a clone of the target used by the previous edition and deploys the edition under validation. The new target (a cluster or stand-alone server) is cloned to target_name-Validation and is displayed as a regular server/cluster.

Note: To use application edition validation, you need to have a configured and working on demand router. The routing policy is used by this component to decide where to send client requests. You must also have an already active edition before starting the validation.

Figure 24-10 shows an example of the validation mode.

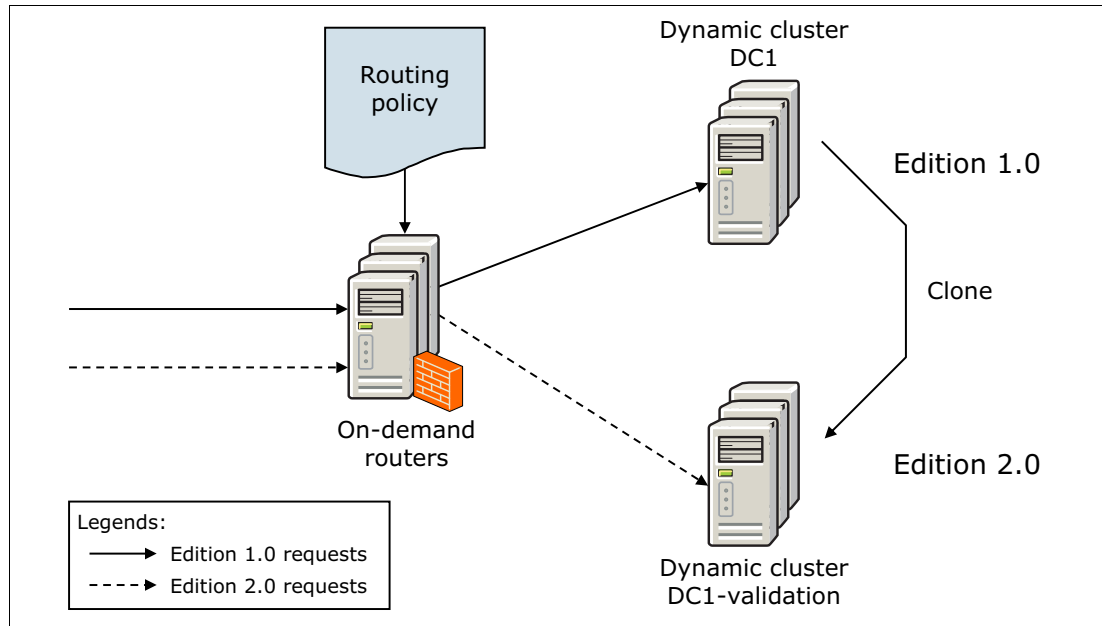


Figure 24-10 Validation mode

To validate an application edition:

1. After you install the new edition to the same target as the previous edition, select **Applications** → **Edition control center** → **application_name**.
2. Select the edition you want to validate, and click **Validate**. Wait for the deployment target to be cloned. Figure 24-11 on page 894 shows the steps executed.

```
Validate Edition

WPVVR0028: Validation started for edition 1.1 of application ITSOJSPSimpleEAR.

WPVVR0038: Validation: Cloned cluster dynaCluster as dynaCluster-Validation.

WPVVR0001: Application ITSOJSPSimpleEAR, Edition 1.1 - edition state set to ACTIVE.

WPVVR0020: Rollout: Synchronizing node was85-1Node01.

WPVVR0020: Rollout: Synchronizing node was85-2Node01.

WPVVR0031: Mapping module ITSOJSPSimple.war,WEB-INF/web.xml to new targets for ITSOJSPSimpleEAR-edition1.1.

WPVVR0020: Rollout: Synchronizing node was85-2Node01.

WPVVR0020: Rollout: Synchronizing node was85-1Node01.

WPVVR0030: Validation completed for edition 1.1 of application ITSOJSPSimpleEAR.

Manage Editions
```

Figure 24-11 Validation process

3. Select **Servers** → **All servers** to check that the cloned validation servers were created. If so, start the servers.
4. Start the application edition under validation in **Applications** → **All applications**, if not already started.
5. Create a new routing policy for the validation edition. This rule ideally applies to a specific set of clients, users, or groups that will have the traffic routed to the application under validation. See “Creating routing policies for application editions” on page 891 for more information about how to create the routing policy.
6. At this point you can access the application and validate it. Make sure you close and re-open all browser windows before accessing the application.
7. After the application is validated, you can either roll it out using the instructions from “Rolling out an edition” on page 894 or cancel the validation. To cancel a validation, go to **Application** → **Edition control center** → *application_name* and then select the edition under validation. Click **Cancel Validation**.
8. After the application is rolled out or after it is cancelled, edit the routing policy, and remove the rules created for the validation edition.

To learn more about edition validation, refer to the following information center website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=twve_appedval

24.4.5 Rolling out an edition

When you perform a rollout on an edition, you replace an active edition with a new edition. If the new edition is compatible with earlier versions, you can perform a rollout to replace the active edition without impacting existing clients. An interruption-free replacement is ensured by quiescing current application requests and either rerouting new requests to other servers or temporarily queuing the requests until the new edition is ready for service.

Figure 24-12 shows a rollout diagram.

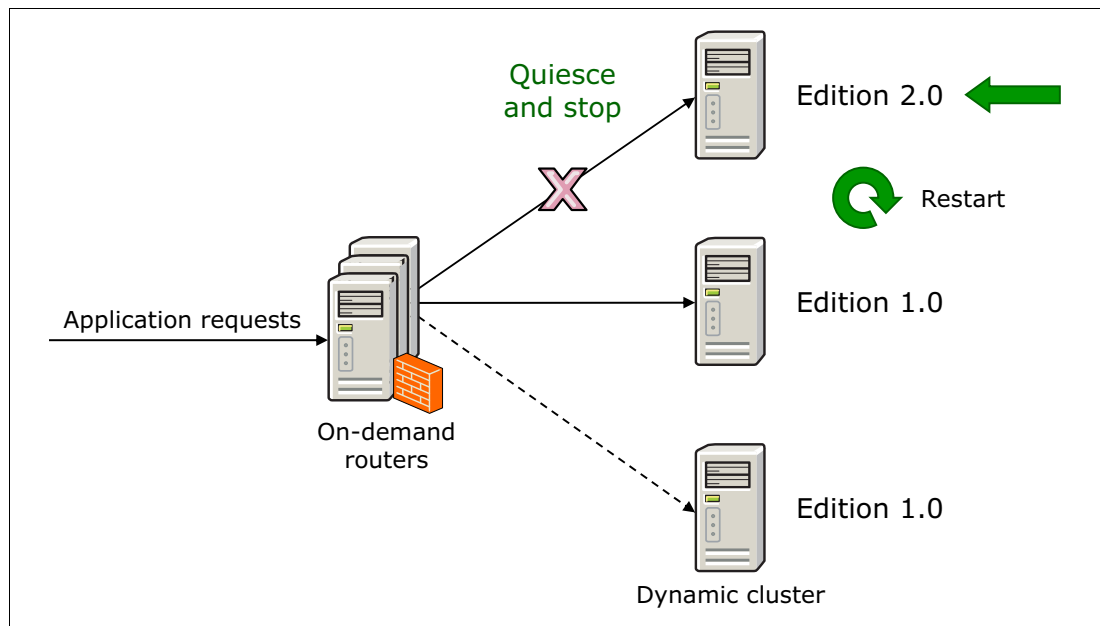


Figure 24-12 Rollout diagram

To perform a rollout on a new edition, you must first install the application edition with the new edition information. You also need to have the previous edition of the application active and started.

Check the information center to learn more about edition compatibility at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=cwve_appedcomp

Before starting an edition rollout, you need to understand a few concepts. These concepts are reviewed in the following paragraphs (rollout strategy, reset strategy, drainage interval, and performing the rollout).

Rollout strategy

The rollout strategy defines how the previous application edition will be replaced (in a group of servers or individually). The following list notes the available options:

- ▶ **Atomic:** Replaces one edition with another in half of the cluster at a time. This rollout type serves all user requests with a consistent edition of the application. While the first half of the cluster is offline and updated, application requests are routed to the second half of the cluster. Verify that half the cluster can handle the entire load during the rollout period.
- ▶ **Grouped:** Replaces one edition with another in a group of servers at a time. The size of this group is configurable so you can decide how many servers can be unavailable during the rollout.

Reset strategy

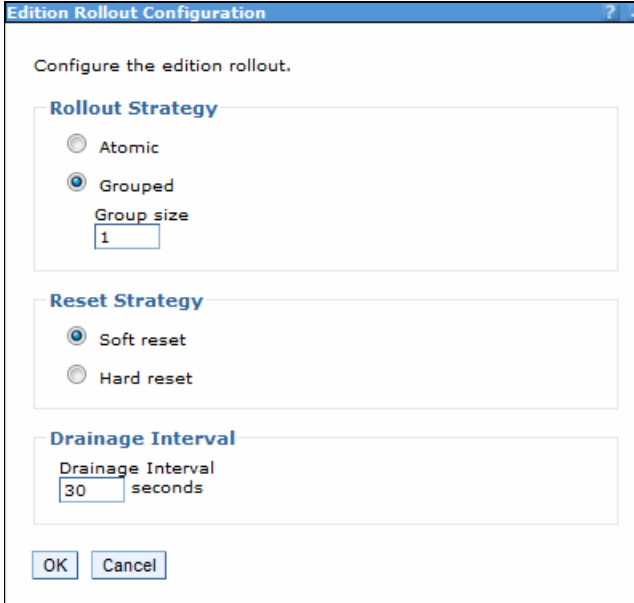
The reset strategy instructs the application edition manager how each deployment target loads the new edition into the server run time. The following list notes the available options:

- ▶ Soft: Resets the application by stopping or restarting the application in each server of the cluster as the next edition replaces the old edition in that server. The server stays up during this process.
- ▶ Hard: Recycles the entire application server as the next edition replaces the former edition, refreshing both process memory and any native libraries used by the application.

Drainage interval

The drainage interval gives the HTTP sessions time to complete before the application or server is reset. The drainage interval specifies the amount of time that the application edition manager waits before the reset strategy starts. The drainage interval allows requests with affinities and in-flight requests to complete. To prevent the loss of transient sessions, set the drainage interval to exceed the application session timeout interval. After the rollout starts, as each server updates, the server is marked as ineligible to begin any new sessions. If the drainage value is set to 0, the process does not wait for sessions to complete.

Figure 24-13 shows the window for all of the optional rollout configurations covered.



The image shows a dialog box titled "Edition Rollout Configuration". It contains three main sections: "Rollout Strategy", "Reset Strategy", and "Drainage Interval".

- Rollout Strategy:** Two radio buttons are present: "Atomic" (unselected) and "Grouped" (selected). Below "Grouped" is a text input field labeled "Group size" containing the value "1".
- Reset Strategy:** Two radio buttons are present: "Soft reset" (selected) and "Hard reset" (unselected).
- Drainage Interval:** A text input field labeled "Drainage Interval" contains the value "30", followed by the text "seconds".

At the bottom of the dialog box are two buttons: "OK" and "Cancel".

Figure 24-13 Edition rollout configuration panel

Note: Tune the SOAP connector properties to set the request timeout value for the deployment manager. Set the value to be greater than the total time required to perform a rollout on your system, and restart the deployment manager. If this property is not set it can cause the rollout process to fail when the request timeout value expires. Estimating the value to set is derived from the following formula:

$$\text{Request Timeout} = \text{RG} * (\text{DI} + \text{IQ} + \text{AR})$$

RG = number of groups to rollout

DI = drainage interval

IQ = internal quiesce timeouts (approximately five minutes)

AR = application or server restart times (approximately 10 minutes)

If you are performing the rollout using the administrative console, adjust the request timeout value by clicking **System administration** → **Deployment manager** → **Administration services** → **JMX connectors** → **SOAPConnector** → **Custom properties** → **requestTimeout**. The default value is 600 seconds and needs to be increased adequately. A value of zero disables the timeout.

Also set the session expiration for the administrative console to a value greater than the amount of time required for the entire rollout process to end. Refer to the information center for how to configure session expiration in the administrative console at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=cons_sessionto

More information about these concepts is in the information center at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=twve_appedroll

Performing the rollout

To perform an interruption-free rollout for a compatible application edition:

1. From the administrative console, go to **Applications** → **Edition control center**.
2. Select the application on which to perform the rollout.
3. Check the edition to be rolled out, and click **Rollout**.
4. Select the rollout strategy, reset strategy, and drainage interval based on the application requirements, and click **OK**, as shown in Figure 24-13 on page 896.
5. After the rollout completes, a window similar to Figure 24-14 on page 898 is displayed.

```

Rollout Edition
WPVR0010: Rollout started for edition 1.1 of application ITSOJSPSimpleEAR.
WPVR0014: Rollout: Edition 1.0 of application ITSOJSPSimpleEAR deactivated. Edition 1.1 activated.
WPVR0020: Rollout: Synchronizing node was85-1Node01.
WPVR0020: Rollout: Synchronizing node was85-2Node01.
WPVR0015: Rollout: Processing server was85-1Node01/staticMember1.
WPVR0515: Rollout: Disallowing new requests w/o affinity to server/application at was85-1Node01/staticMember1/ITSOJSPSimpleEAR-edition1.0.
WPVR0054: Rollout: Draining was85-1Node01/staticMember1/ITSOJSPSimpleEAR-edition1.0 (30 seconds)
WPVR0053: Rollout: Drain completed for was85-1Node01/staticMember1/ITSOJSPSimpleEAR-edition1.0.
WPVR0016: Rollout: Quiescing server/application at was85-1Node01/staticMember1/ITSOJSPSimpleEAR-edition1.0.
WPVR0018: Rollout: Stopping was85-1Node01/staticMember1/ITSOJSPSimpleEAR-edition1.0.
WPVR0020: Rollout: Synchronizing node was85-1Node01.
WPVR0022: Rollout: Starting was85-1Node01/staticMember1/ITSOJSPSimpleEAR-edition1.1.
WPVR0061: was85-1Node01/staticMember1/ITSOJSPSimpleEAR-edition1.1 started.
WPVR0020: Rollout: Synchronizing node was85-1Node01.
WPVR0020: Rollout: Synchronizing node was85-2Node01.
WPVR0015: Rollout: Processing server was85-2Node01/staticMember2.
WPVR0515: Rollout: Disallowing new requests w/o affinity to server/application at was85-2Node01/staticMember2/ITSOJSPSimpleEAR-edition1.0.
WPVR0054: Rollout: Draining was85-2Node01/staticMember2/ITSOJSPSimpleEAR-edition1.0 (30 seconds)
WPVR0053: Rollout: Drain completed for was85-2Node01/staticMember2/ITSOJSPSimpleEAR-edition1.0.
WPVR0016: Rollout: Quiescing server/application at was85-2Node01/staticMember2/ITSOJSPSimpleEAR-edition1.0.
WPVR0018: Rollout: Stopping was85-2Node01/staticMember2/ITSOJSPSimpleEAR-edition1.0.
WPVR0020: Rollout: Synchronizing node was85-2Node01.
WPVR0022: Rollout: Starting was85-2Node01/staticMember2/ITSOJSPSimpleEAR-edition1.1.
WPVR0061: was85-2Node01/staticMember2/ITSOJSPSimpleEAR-edition1.1 started.
WPVR0012: Rollout for edition 1.1 of application ITSOJSPSimpleEAR completed successfully.
Manage Editions

```

Figure 24-14 Rollout edition completion panel

24.4.6 Rolling back an edition

If after the application rollout you find an error and decide to go back, you can easily do this by rolling back to the former edition. The rollback process works the same way as the rollout. The only difference is that you select a previous working edition during the rollout procedure.

To rollback an edition:

1. From the administrative console, go to **Applications** → **Edition control center**.
2. Select the application that you want to rollback.
3. Check the previously known working edition, and click **Rollout**.

Select the rollout strategy, reset strategy, and drainage interval, based on the application requirements, and click **OK**, as shown in Figure 24-13 on page 896.

Note: The edition that you want to go back to needs to be installed. If the edition was deleted, you can install it again and then perform the rollback.

24.5 Hot deployment and dynamic reloading

Hot deployment and dynamic reloading characterize how application updates are handled when updates to the applications are made by directly manipulating the files on the server. In either case, updates do not require a server restart, although they might require an application restart. These features provide the following capabilities:

- ▶ Hot deployment of new components

Hot deployment of new components is the process of adding new components, such as WAR files, EJB JAR files, servlets, and JSP files to a running application server without having to stop and then restart the application server.

However, in most cases, such changes require the application itself to be restarted so that the application server run time reloads the application and its changes.

- ▶ Dynamic reloading of existing components

Dynamic reloading of existing components is the ability to change an existing component without the need to restart the application server for the change to take effect. Dynamic reloading can involve changes of the following types:

- Implementation of an application component, such as changing the implementation of a servlet
- Settings of the application, such as changing the deployment descriptor for a web module

To edit the application files manually, locate the binaries in use by the server (in most cases these binaries are in the application server `installedApps` directory). The application files can be edited manually on one or more of the nodes. These changes are overwritten the next time the node synchronizes its configuration with the deployment manager. Therefore, perform manual editing of an application's files only in the master repository, which is located on the deployment manager system.

Tip: If you are not familiar with updating applications by manipulating the server files directly, consider using the administrative console update wizard.

The following settings can affect dynamic reload:

- ▶ Reload classes when application files are updated

For application files to be reloaded automatically after an update, the **Override class reloading settings for Web and EJB modules** setting must be enabled, and the **Polling interval for updated files** setting must be greater than 0.

Click **Applications** → **Application Types** → **WebSphere enterprise applications**, and click the link for the application. In the **Detail properties** section, click the **Class loading and update detection** link.

- ▶ Application Server class loader policy

Set the application server's class loader policy to **Multiple**. If it is set to **Single**, you must restart the application server after an application update.

Click **Servers** → **Server Types** → **WebSphere application servers**, and click the server name. The setting is found in the **General Properties** section, under **Server-specific Application Settings**.

► JSP Reload options for web modules

A web container reloads a web module only when this setting is enabled, which is a default setting.

Click **Applications** → **Application Types** → **WebSphere enterprise applications**, and click the link for the application. In the Web Module Properties section, click **JSP and JSF options** and then select the **JSP enable class reloading** option. Enter a polling interval.

For more information about using hot deployment and dynamic reload, refer to the information center at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=trun_app_hotupgrade



Working with SCA applications

In this chapter, we discuss how to work with Service Component Architecture (SCA) applications. We introduce the basic elements of an SCA application and how to package and export SCA applications.

This chapter contains the following topics:

- ▶ SCA application introduction
- ▶ Preparing to use the sample application
- ▶ Packaging an SCA application for deployment
- ▶ Deploying an SCA application
- ▶ Additional resources for learning

Additional information about developing, testing, and deploying SCA applications are in the following publications:

- ▶ *Rational Application Developer for WebSphere Software V8 Programming Guide*, SG24-7835
- ▶ *Getting Started with WebSphere Application Server Feature Pack for Service Component Architecture*, REDP-4633

25.1 SCA application introduction

Support for SCA offers a simple and powerful way to construct applications based on service-oriented architecture (SOA). The support in WebSphere Application Server V8.5 uses the Apache Tuscany open-source technology to provide an implementation of the published SCA specifications. WebSphere Application Server supports the Open SOA Collaboration SCA specification (OSOA SCA 1.0) and the SCA OASIS programming model (OASIS SCA 1.1).

Figure 25-1 shows a technical overview of an SCA domain.

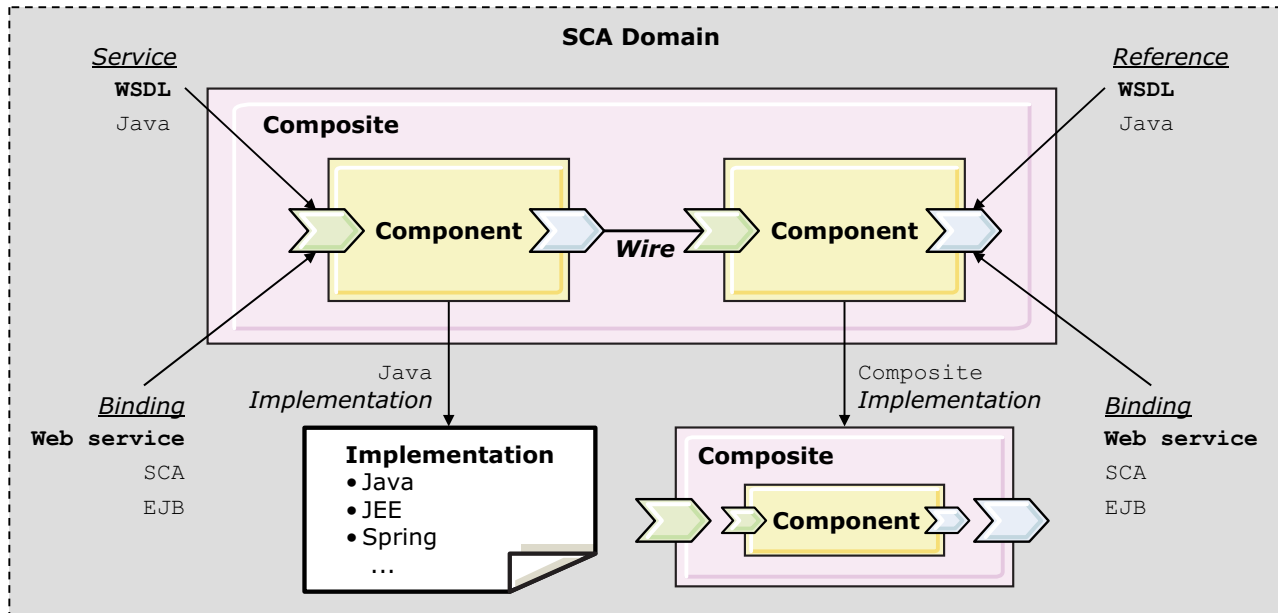


Figure 25-1 SCA technical overview

The functions of these components are noted in the following list:

- ▶ SCA *components* implement business functions in the form of services.
- ▶ The SCA component is a configured instance of an *implementation*, which is program code that implements one or more business functions, such as Java classes.
- ▶ SCA components are grouped into *composites*, which are the deployment unit for an SCA in WebSphere Application Server.
- ▶ An SCA *domain* consists of the definitions of composites, components, their implementations, and the nodes on which they run.
- ▶ Components that are deployed into a domain can directly *wire* to other components within the same domain. The SCA domain is typically the cell on multiple-server installations and the server scope on single-server installations.

You can find information about SCA specifications supported in WebSphere Application Server V8.5 at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=rovr_specs

25.1.1 SCA component

Components both provide and consume services. Figure 25-2 shows part of an SCA component.

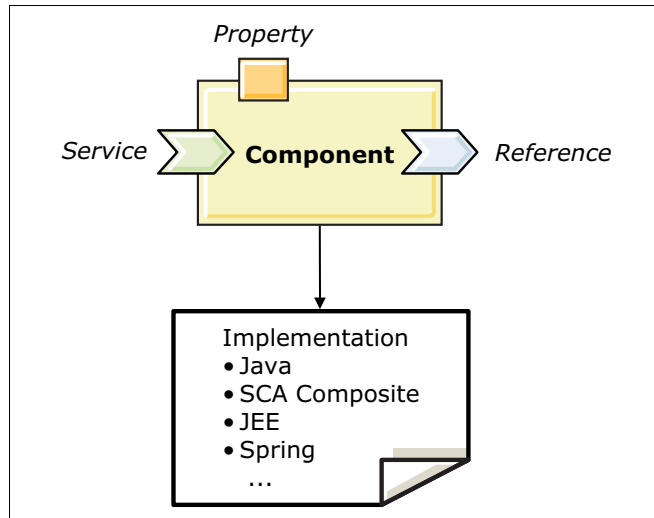


Figure 25-2 SCA component

An SCA component consists of the following parts:

- ▶ The service: The component provides a *service* or business function to its client.
- ▶ The reference: The component can have a *reference* to a service that is provided by another component.
- ▶ An implementation: The component contains a property that declares the implementation. The component reads the property value from the configuration file when the component is instantiated. In WebSphere Application Server V8.5, the service implementation includes the following components:
 - Java POJO
 - Java EE integration
 - Other SCA composites
 - Spring 2.5.5 containers

25.1.2 SCA composite

Assemblies of components are formally grouped into *composites*. A composite is the set of components and wires (that is, the assembly of services). It is the basic deployment unit for SCA in WebSphere Application Server. The components, assemblies, internal wires, and service and reference definitions are written in an open XML language called Service Component Definition Language (SCDL).

The composite provides a scoping mechanism that defines a local boundary for components but that can also hide services that are provided in components that are not intended for other SOA applications. When defined, a composite can be reused to provide the implementation for other components in a nested fashion.

Services and references in a composite are bound to specific protocols (such as web services) using bindings. The bindings are part of the SCDL definition, and the business logic (implementation) does not need this detail.

As illustrated in Figure 25-3, an SCA composite includes the following elements:

- ▶ SCA components: One or more SCA components wired together.
- ▶ Interfaces: Services and references have an *interface* definition that points to the location of the WSDL or Java that describes the interface. The *service interface* provides the information that clients use to call the service. A *reference interface* provides the information that the component needs to call another service.
- ▶ Bindings: Services and references are bound to specific protocols using *bindings*. The use of bindings removes the details of connection from the business logic.
- ▶ Policies and intents: Policy and quality of service intents can decorate services or references (called *interaction intents*) and can decorate components (called *implementation intents*).

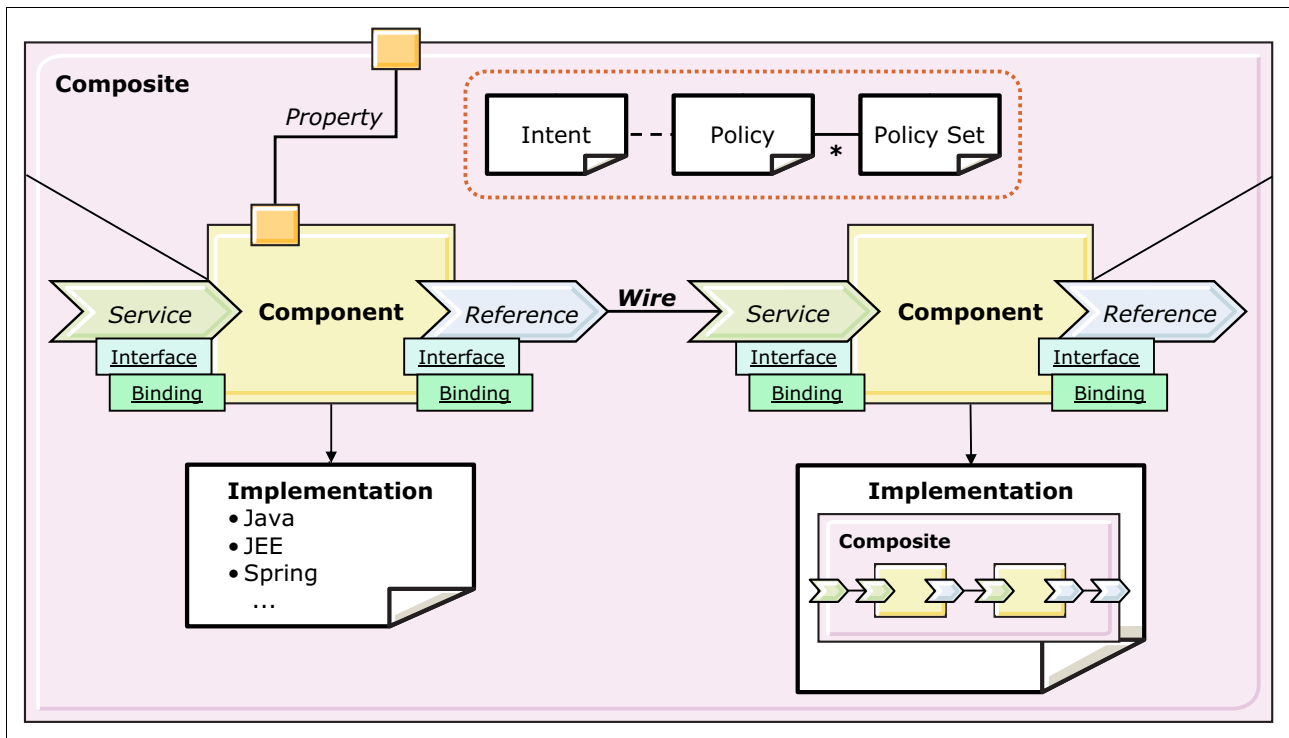


Figure 25-3 SCA composite

An application can contain one composite or several different composites. The components of a composite can run in a single process on a single computer or can be distributed across multiple processes on multiple computers. The components might all use the same implementation language or they can use different languages.

An SCA composite is typically described in a configuration file, the name of which ends in `.composite`. Figure 25-4 shows a composite definition, named `SupportFeeds.composite`, that is located in the project root folder of the `AtomContentFeed` composition unit in the `RAD8AtomFeeds` sample application.

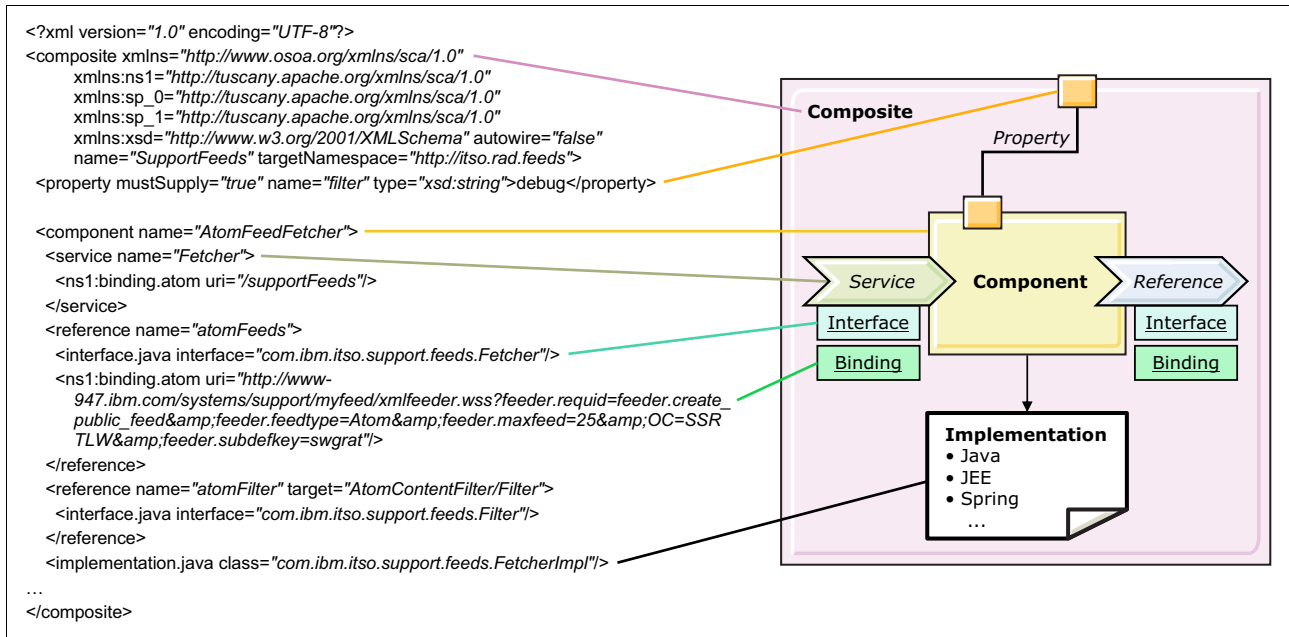


Figure 25-4 SCA composite file

Naming note: A composite file in a WAR file must be named `default.composite`. A composite file that is not in a WAR file can have any name.

25.1.3 SCA contribution

An SCA *contribution* contains artifacts that are needed for an SCA domain. Contributions are sometimes self-contained, in that all of the artifacts necessary to run the contents of the contribution are found within the contribution itself. However, the contents of the contribution can make one or many references to artifacts that are not contained within the contribution. These references might be to SCA artifacts or to other artifacts, such as Web Services Description Language (WSDL) files, XSD files, or to code artifacts such as Java class files.

Figure 25-5 shows composites in an sca-contribution.xml file in an SCA domain.

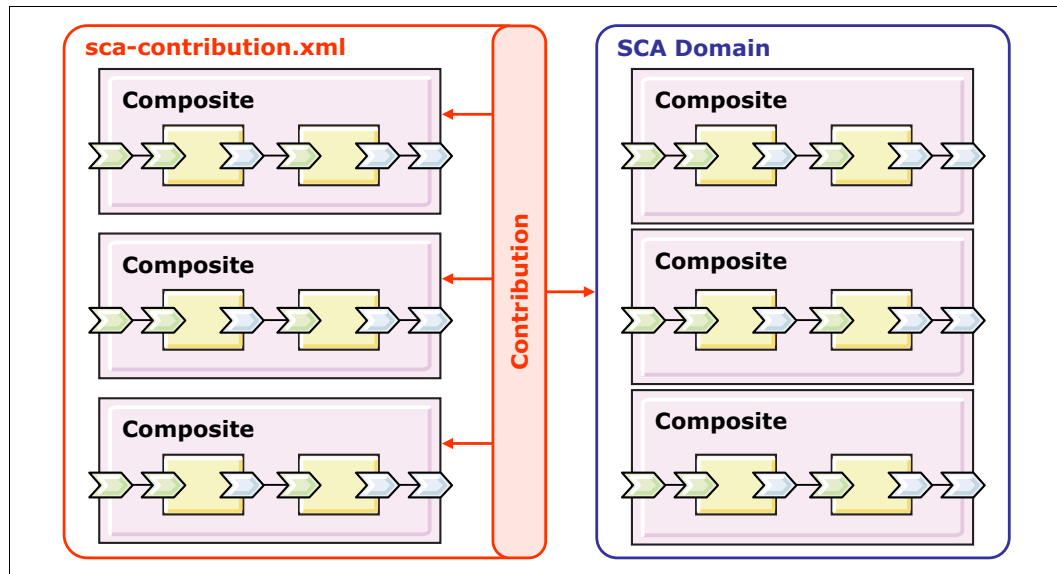


Figure 25-5 SCA contribution

An SCA contribution is typically described in a contribution file, named sca-contribution.xml in the META-INF directory. Example 25-1 shows the contribution file for the SupportFeeds composition unit.

Example 25-1 Contribution file

```
<?xml version="1.0" encoding="UTF-8"?>
<contribution xmlns="http://www.oxa.org/xmlns/sca/1.0">
  <deployable composite="ns1:SupportFeeds"
  xmlns:ns1="http://itso.rad.feeds"></deployable>
</contribution>
```

25.2 Preparing to use the sample application

The concepts in this chapter are illustrated using the RAD8AtomFeeds sample application, which is included with *Rational Application Developer for WebSphere Software V8 Programming Guide*, SG24-7835.

25.2.1 Downloading the application

To download the sample application:

1. Go to the following website:
<http://www.redbooks.ibm.com/abstracts/sg247835.html?open>
2. Click the **Additional Material** link.
3. Click the sg247835.zip file, and select **Save** to save the compressed file to your computer.
4. Extract the contents of the compressed file.

The SCA artifacts are in the 7835codesolution\sca directory.

25.2.2 Importing the application to the development tool

To use the sample application for our exercise, import both files into IBM Rational Application Developer by completing the following steps:

1. Start IBM Rational Application Developer.
2. To import the code, click **File** → **Import** and then expand the General section. Select **Existing projects into workspace**. Click **Next**.
3. Click **Browse** next to the Select Archive file field, and browse to the sca directory where you extracted the sample code. Select the **RAD8AtomFeeds.zip** file, and click **Open**.
4. Click **Select All** to select all projects in the file. Click **Finish**. If you are prompted to migrate the workspace, accept and complete the migration wizard.

25.2.3 Completing the service definition

The last action is to make sure the interface for the composite is complete by completing the following steps:

1. Open the composite by double-clicking **SCA 1.0 Content** → **Composites** → **http://itso.rad.feeds** → **SupportFeeds**.
2. Click the **Service** icon (green arrow) in the AtomFeedFetcher component and then select the **Properties** view.
3. Switch to (by clicking) the **Interface** tab and verify that the Interface is of type Java and that it has the value `com.ibm.itso.support.feeds.Fetcher`. If not, correct the fields, and save and close the composite.

These steps are illustrated in Figure 25-6.

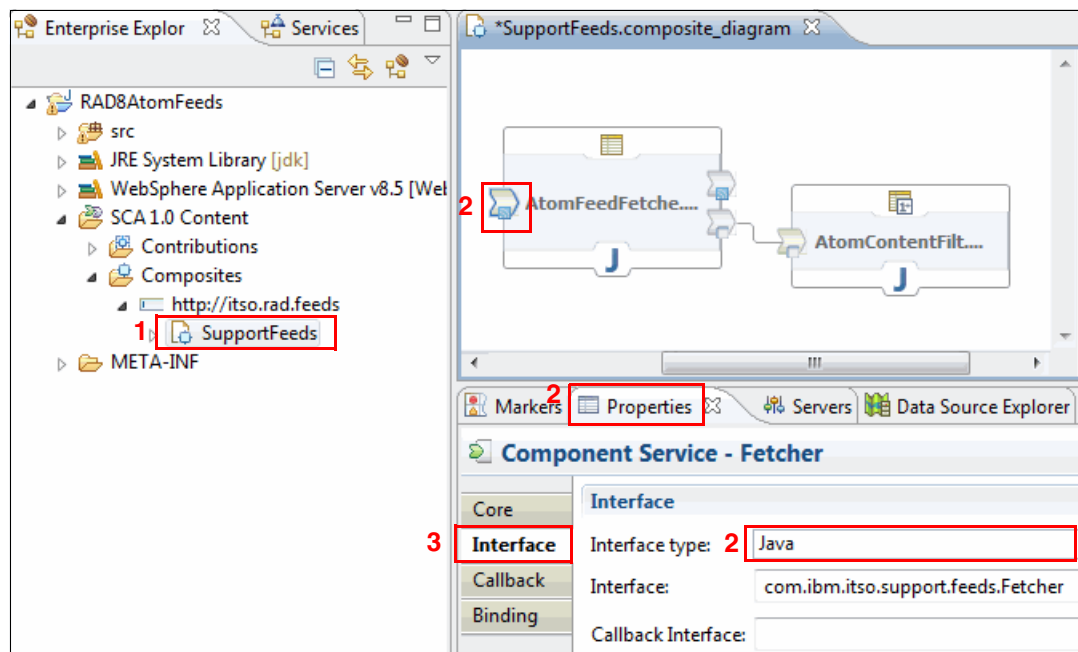


Figure 25-6 Open the SupportFeeds composite

25.3 Packaging an SCA application for deployment

Figure 25-7 shows the files that are contained in the sample SCA application package, as shown in the Enterprise Explorer view in IBM Rational Application Developer.

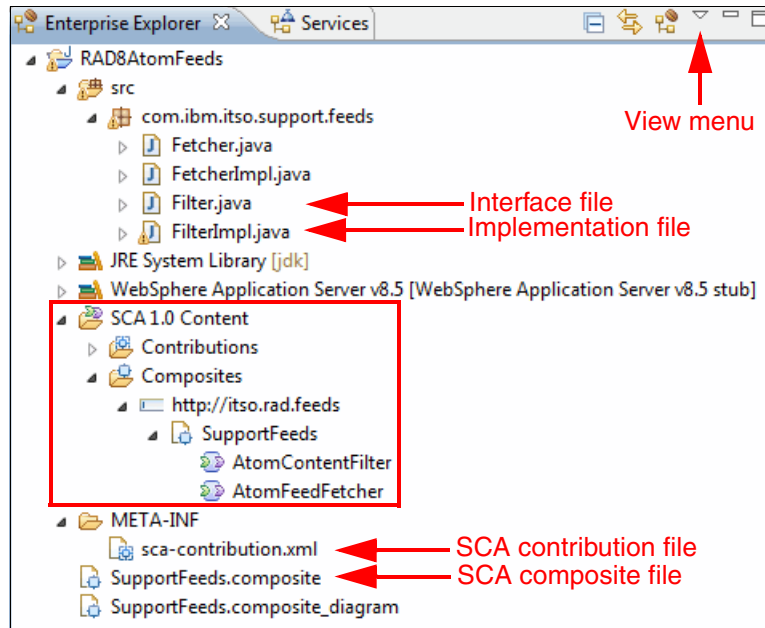


Figure 25-7 SCA application package

Tip: SCA applications are displayed in the Enterprise Explorer view of the IBM Rational Application Developer with features in a manner that makes working with the components easy. The SCA Content section (highlighted in Figure 25-7) contains entries for the contributions and composites, making them easily visible and selectable to open for editing. This is simply a structural view of the files. The SCA contribution file, SCA composite, and SCA composite diagram files (all located beneath the META-INF folder in Figure 25-7) are not normally visible as raw files.

We changed the view properties for the Enterprise Explorer view to show them to you in this format. So, you see these files twice, once by file name and once as an element under the SCA Content folder. Double-clicking **SCA 1.0 Content** → **Contributions** → **sca-contribution** opens the same file that opens if you double-clicked **META-INF** → **sca-contribution.xml**. The same is true for the composite.

To use these settings, click the **View** menu. Click **Customize view**. Click the **Filter** tab, and clear **SCA 1.0 resources**.

The composites are created by the developer as part of the application development process. The developer adds components to the composite, writes the implementation code for the components, and adds the interface files. These development activities are independent of deployment activities. The contribution file is related to packaging for deployment and must be created to deploy the application.

You can see the composite and components in Figure 25-8. Selecting any of the components in the composite shows the properties in the lower window. In Figure 25-8, the **AtomFeedFetcher** component is selected, and in the properties window, you can see the implementation class listed.

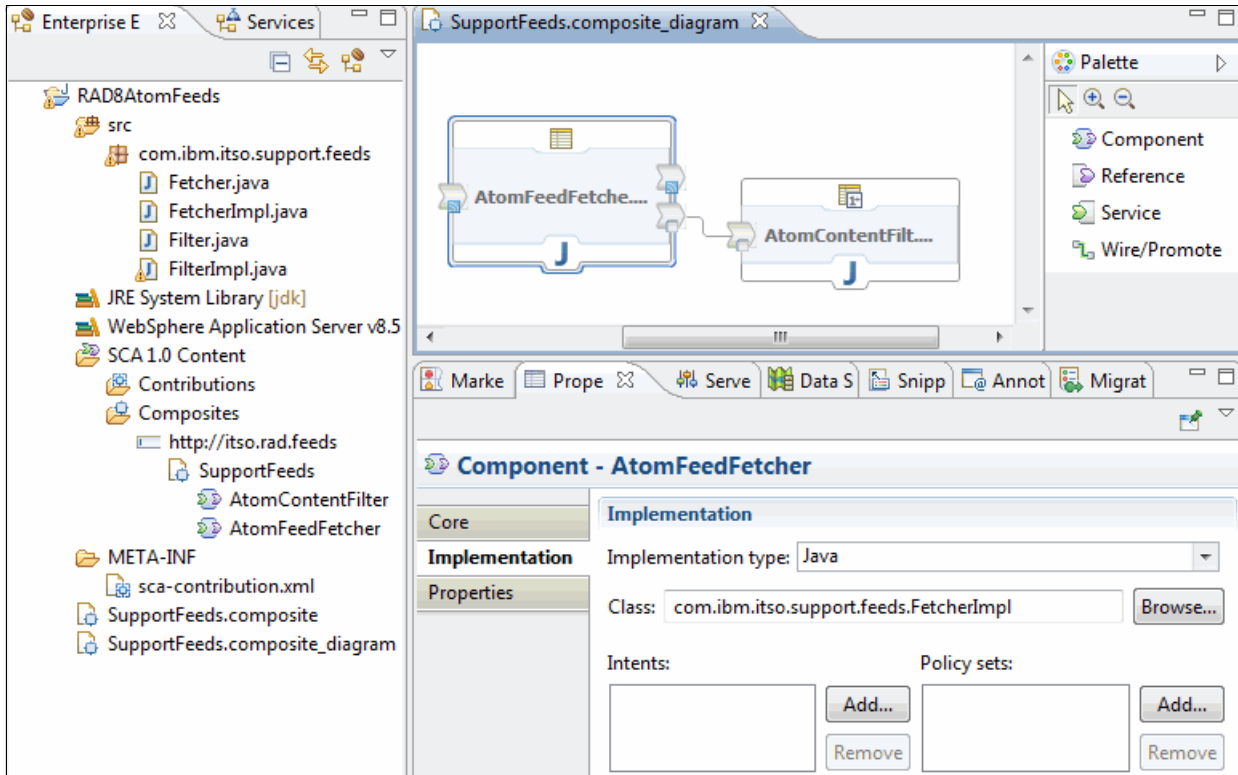


Figure 25-8 SupportFeeds composite

25.3.1 Creating the contribution

Before you can export the project for deployment, you need to create the contribution file. This is a simple process using the IBM Rational Application Developer.

Complete the following steps to create the contribution file:

1. In the Enterprise Explorer, expand **SCA 1.0 Content**. Right-click **Contributions**, and click **New** → **SCA 1.0 Contribution**. (Figure 25-9).

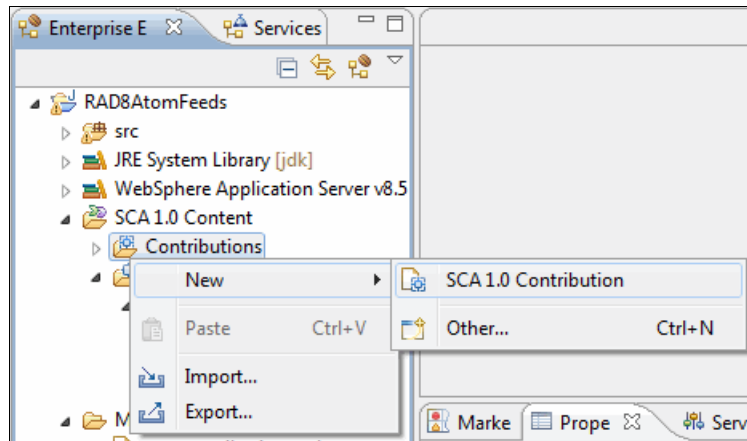


Figure 25-9 Contribution file creation

2. In the New Contribution Wizard window, select the composites to be deployed (Figure 25-10), and click **Finish**.

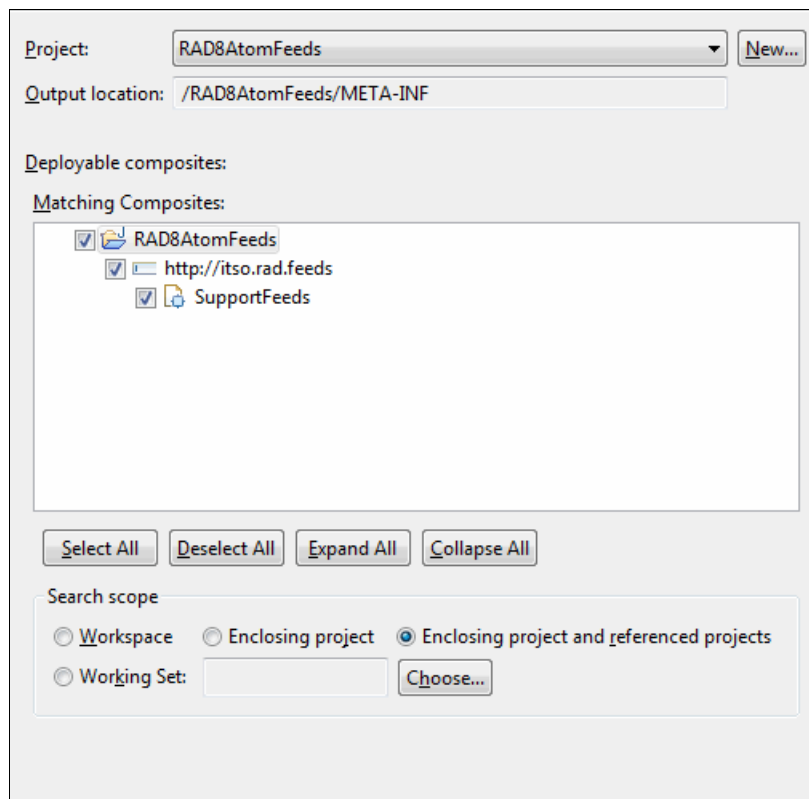


Figure 25-10 Add composites to the contribution

25.3.2 Exporting the SCA application for deployment

To export the contribution as an SCA archive file:

1. Right-click the **project**, and select **Export**, as shown in Figure 25-11.

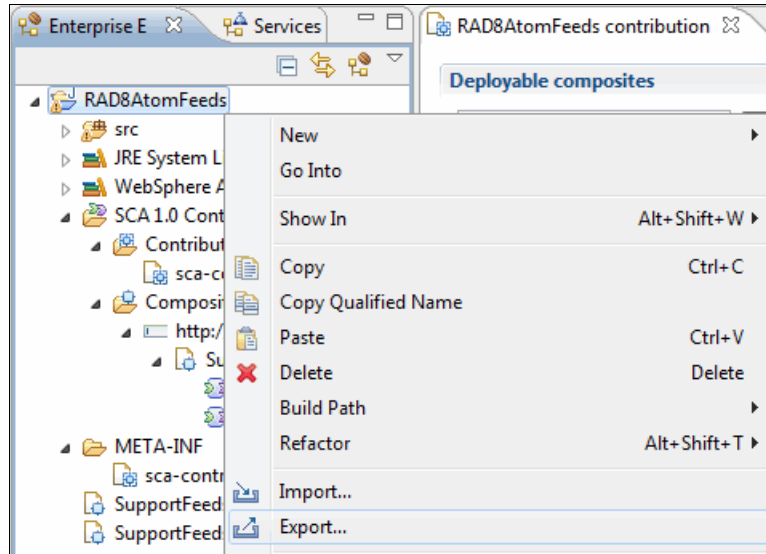


Figure 25-11 Exporting an SCA application

2. In the Export dialog box, select **Service Component Architecture 1.0** → **SCA Archive File**, as shown in Figure 25-12 and then click **Next**.

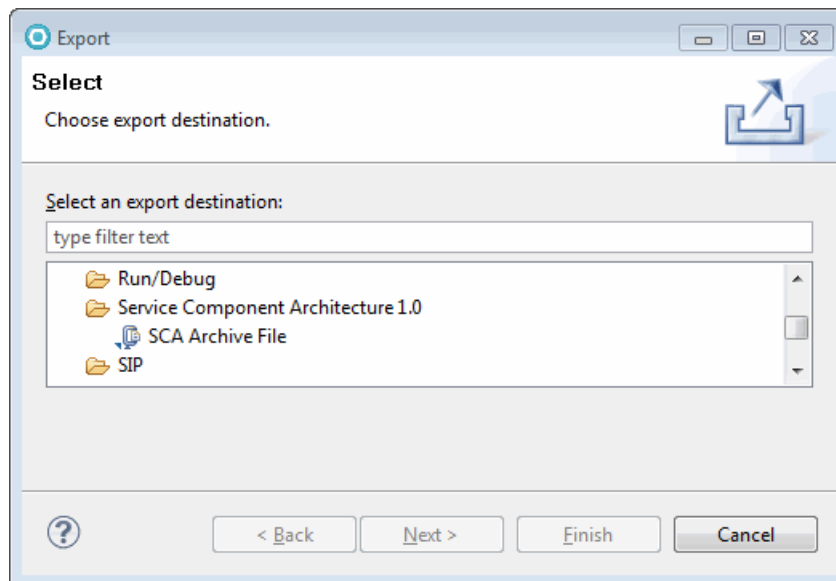


Figure 25-12 Selecting the archive file

3. Select the project to export (Figure 25-13), and click **Finish**.

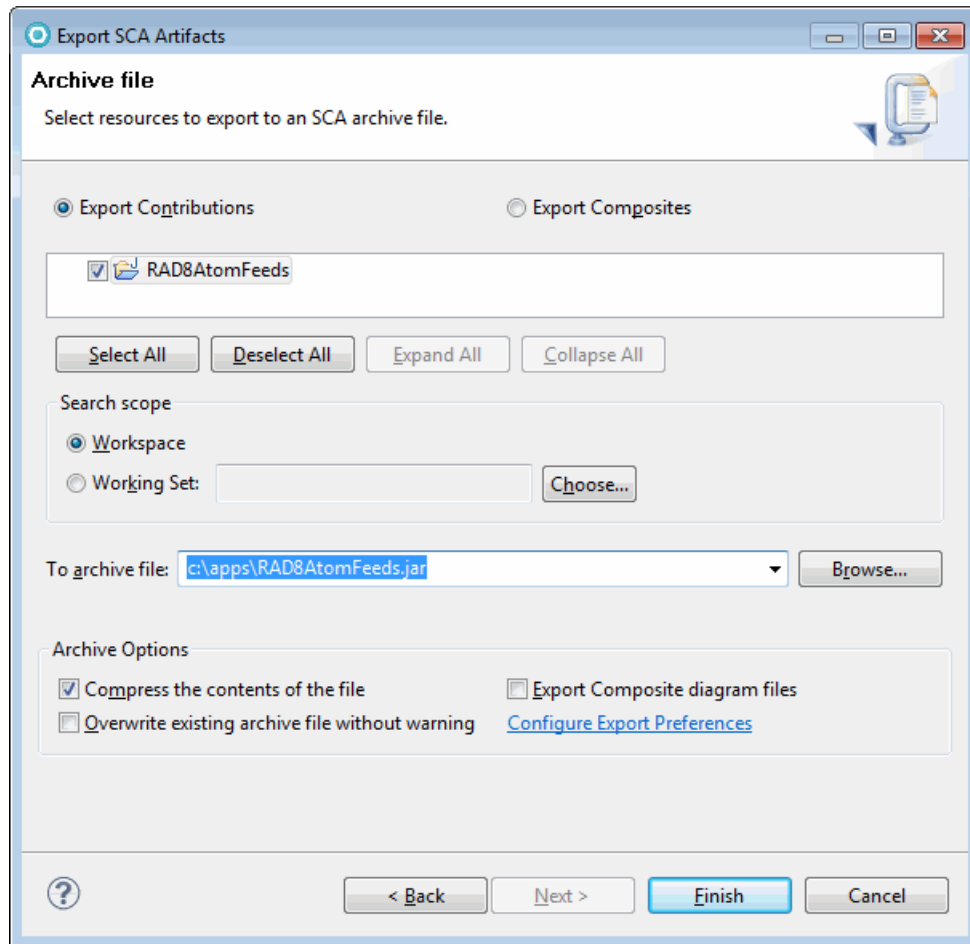


Figure 25-13 Exporting project

Now, you can deploy the RAD8AtomFeeds.jar SCA archive file.

25.4 Deploying an SCA application

A typical SCA solution consists of a combination of web, EJB, and SCA applications. The web and EJB modules are deployed as enterprise applications, and the SCA composites are deployed as assets in a business-level application.

This section describes how to deploy an SCA composite as an asset in a business-level application.

25.4.1 Importing the SCA archive file as an asset

The first step in deploying the application is to import the SCA archive file into the application server environment as an asset.

Complete the following steps to import an asset:

1. Click **Applications** → **Application Types** → **Assets**. In the Assets window, shown in Figure 25-14, click **Import**.



Figure 25-14 Import an asset

2. Select the SCA archive file, as shown in Figure 25-15 and then click **Next**.

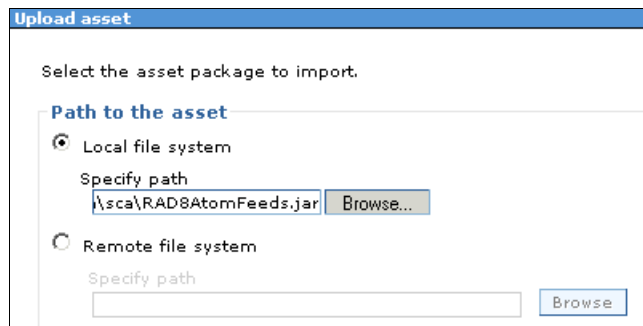


Figure 25-15 Identify the jar file location

3. Provide the required options. In this example, take the default values, as shown in Figure 25-16.

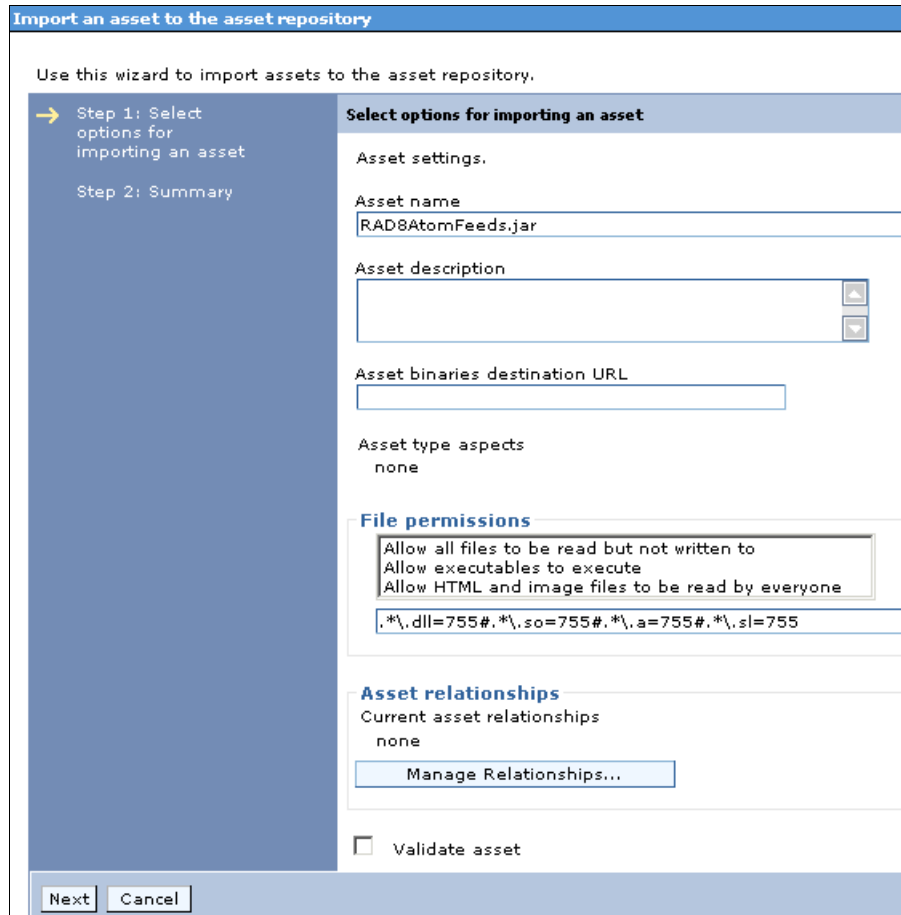


Figure 25-16 Options for importing an asset

The following lists explains the fields of note from Figure 25-16:

- Asset binaries destination URL: When the asset is imported, the binaries are extracted by default to *profile_root/installedAssets/asset_name/BASE*. You can change this location by specifying a new one here.
- File permissions: The default setting for the file permissions to be assigned to the extracted asset binaries is the *Allow executables to execute setting*. You can select another option or provide a customized setting.
- Asset relationships: Click the **Manage Relationships** button to specify assets to which this asset is related.
- Validate asset: Selecting this option enables validation of the references specified in the asset when it is imported. The asset is examined to ensure that references are defined in the scope of the deployment target of the asset. Some examples of these references are data sources or references contained in deployment descriptors (such as resource and resource environment references).

Click **Next**.

- In the Summary page, click **Finish** to import the file and save the modification to the master configuration. The new SCA archive file is now listed as an asset, as shown in Figure 25-17.



Figure 25-17 Asset imported

25.4.2 Creating a new business-level application

To create a new business-level application:

- Click **Applications** → **Application Types** → **Business-level applications**. In the Business-level applications window, shown in Figure 25-18, click **New**.

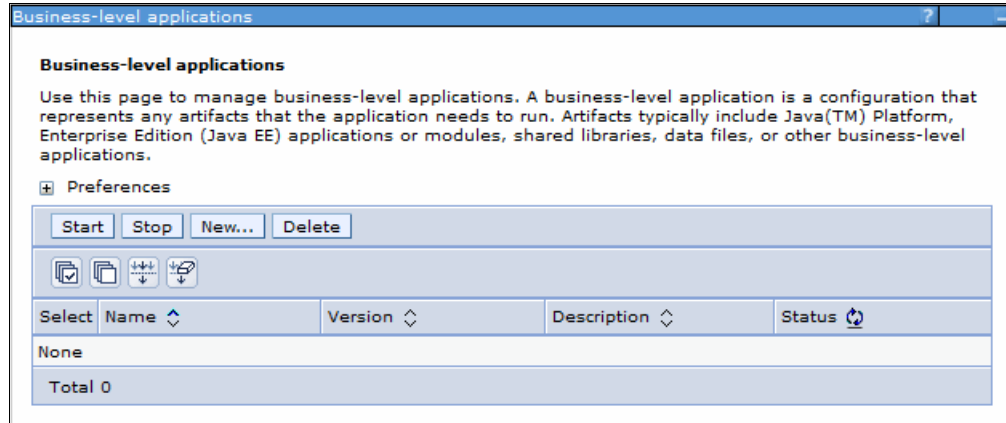


Figure 25-18 Business-level application window

2. Enter a new name for the business-level application, as shown in Figure 25-19. Click **Apply**. Save the change to the master configuration.

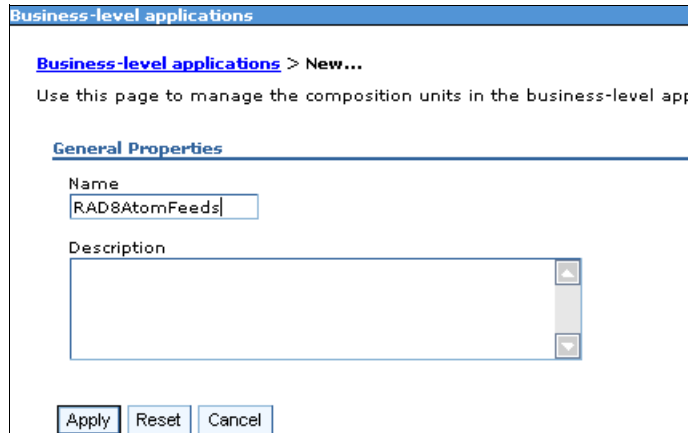


Figure 25-19 Create a new business-level application

25.4.3 Adding the new asset to the business-level application

Add the new asset, as a composition unit, to the RAD8AtomFeed business-level application by completing the following steps:

1. Click **Applications** → **Application Types** → **Business-level applications** and then click **RAD8AtomFeeds**, as shown in Figure 25-20.

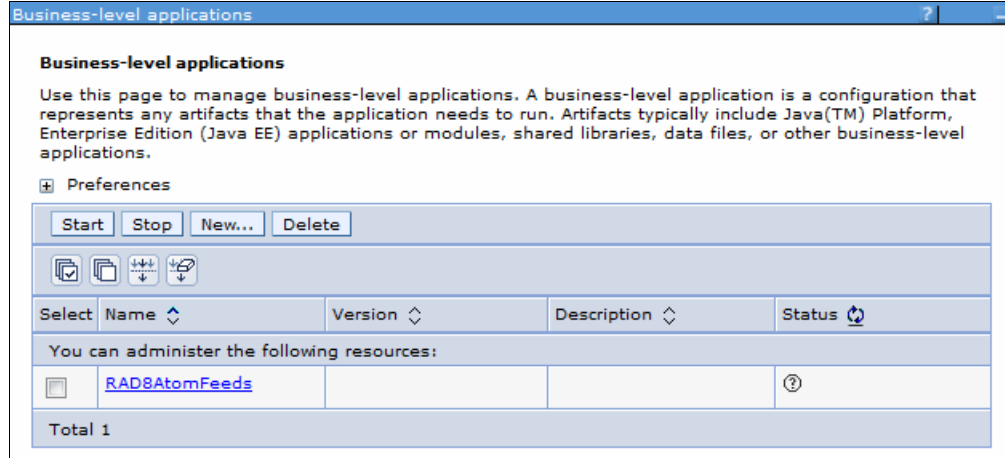


Figure 25-20 Click the business-level application

2. In the Deploy assets section, click **Add** → **Add Asset**, as shown in Figure 25-21.

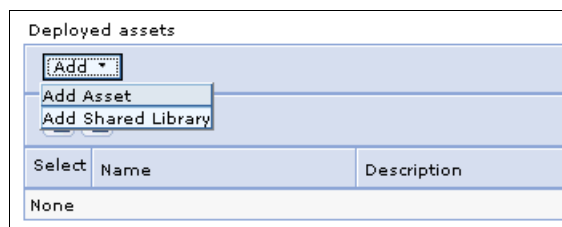


Figure 25-21 Add asset

- Choose the asset that you want to deploy to this business-level application, as shown in Figure 25-22, and click **Continue**.



Figure 25-22 Choose the asset

- Review the options for the composite unit, as shown in Figure 25-23. In this example, we use the default values.

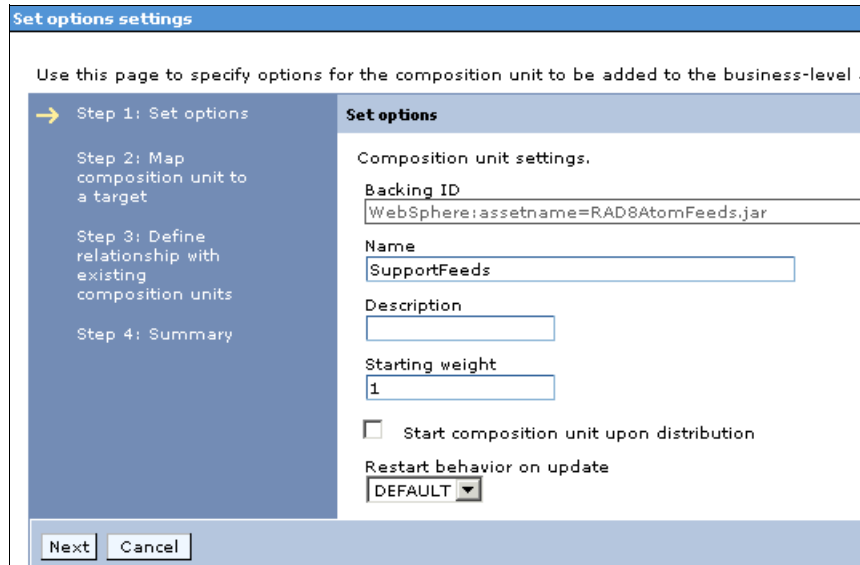


Figure 25-23 Options for the composite unit

The fields displayed in Figure 25-23 are described in the following list:

- Backing ID: Displays the unique identifier for the composition unit that will be registered in the application domain. You cannot change this field.
- Name: Name for the composition unit.
- Starting weight: Specifies the order in which composition units are started when the server starts. The composition unit with the lowest number is started first.
- Start composition unit upon distribution: Specifies whether to start the composition unit when it is distributed to other locations. This setting applies only to assets and shared library composition units.
- Restart behavior on update (of the composition unit):
 - ALL: Restarts the composition unit after the entire composition unit is updated.

- **DEFAULT:** Restarts the composition unit after the part of the composition unit is updated.
- **NONE:** Does not restart the composition unit after the composition unit is updated.

Click **Next**.

5. The next window allows you to select the target server for deployment. Select the server in the Available column, and click the **right arrow** to move it to the Selected column. Click **Next**.

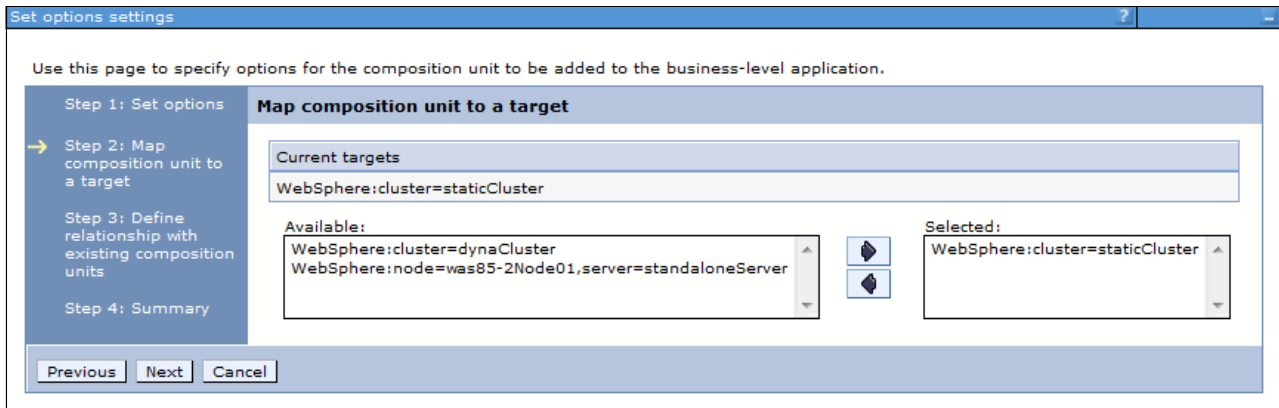


Figure 25-24 Map the composition unit to a target

6. The next window allows you to manage relationships with an existing composition unit. A relationship declares a dependency of this composition unit to another asset deployed as a shared library. In this example, we have none to declare. Click **Next**.

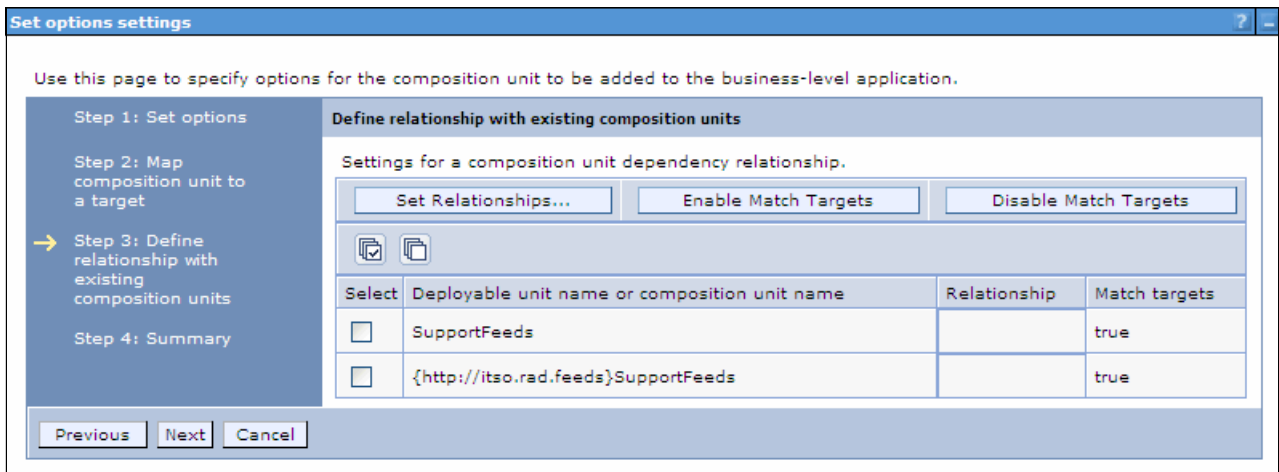


Figure 25-25 Define relationships

7. The last window is a summary window. Review your selections and then click **Finish**.
8. Save the update to the master configuration.

25.4.4 Starting and verifying the business-level application

The last step is to start the business-level application and to verify that it is available by completing the following steps:

1. Click **Applications** → **Application Types** → **Business-level applications**, choose the **RAD8AtomFeeds** application and then click **Start**. The application starts, as shown in Figure 25-26.

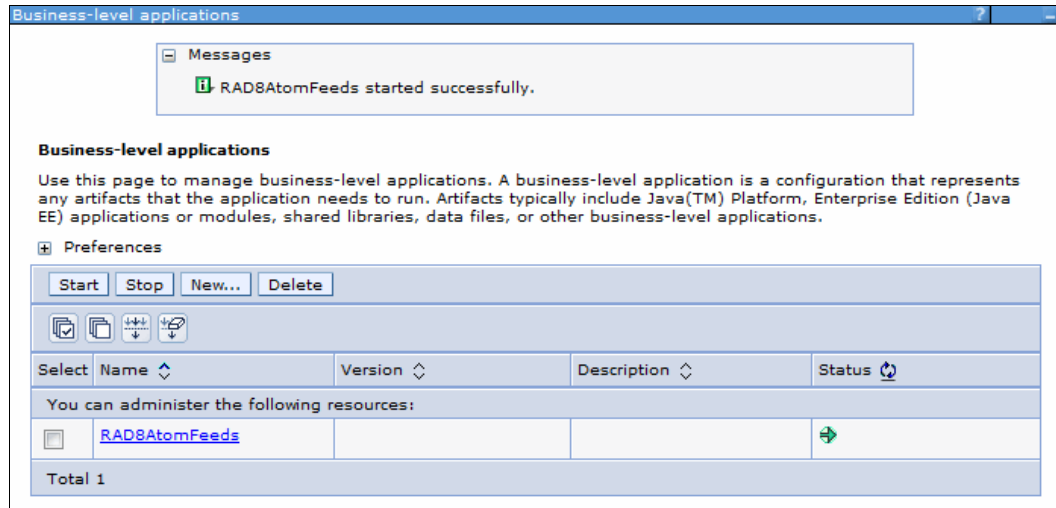


Figure 25-26 Business-level application started

2. Verify the offered service from a web browser by entering the following URL:
`http://hostname:port/supportFeeds`
Typically, the *hostname* is the localhost and the *port* has the value 9080.

25.5 Additional resources for learning

Deploying SCA composites in the information center:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=welc6tech_sca_dep

Samples and Tutorials with SCA using IBM Rational Application Developer:

http://publib.boulder.ibm.com/infocenter/radhelp/v9/index.jsp?topic=%2Fcom.ibm.sca.tools.doc%2Ftopics%2Fsca_tools_intro.html



Working with OSGi applications

This chapter provides information about WebSphere Application Server V8.5 OSGi capabilities. We describe what an OSGi application is, the OSGi lifecycle, and how to package, deploy and manage an OSGi application on a WebSphere Application Server.

This chapter includes the following topics:

- ▶ OSGi overview
- ▶ Enterprise OSGi
- ▶ Using the sample application
- ▶ Packaging OSGi applications
- ▶ Deploying OSGi applications
- ▶ Administrating OSGi applications

A detailed explanation of the OSGi framework is out of the scope of this publication. Refer to the following sources for more information about that topic:

- ▶ The home page of the OSGi alliance:
<http://www.osgi.org>
- ▶ Eclipse Equinox OSGi implementation:
<http://eclipse.org/equinox>
- ▶ The WebSphere Application Server V8.5 Information Center:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=/com.ibm.websphere.osgi.nd.doc/topics/ca_blueprint.html

26.1 OSGi overview

Open Services Gateway initiative (OSGi) is a framework based on Java technology. OSGi enables you to write modular, dynamic, and versioning applications. WebSphere Application Server V8.5 uses the Eclipse Equinox as the framework for OSGi applications and OSGi V4.2.

You can design and build applications from coherent, versioned, reusable OSGi modules that are accessed only through well-defined interfaces. The new features of WebSphere Application Server V8.5 support for OSGi applications includes the following list:

- ▶ An OSGi application can contain Enterprise JavaBean (EJB) that allows direct access and invokes an enterprise bean.
- ▶ A blueprint configuration file can specify a user role for security access to the methods of the bean.

The OSGi framework in WebSphere Application Server provides support for each of the layers of the OSGi Architecture:

- ▶ The Modules layer
- ▶ The Life-cycle layer
- ▶ The Services layer

We discuss each of them in following sections.

26.1.1 OSGi application model

This section covers OSGi deployment, metadata, and the dependencies and capabilities of OSGi through versioning.

OSGi bundles and bundle archives

The unit of deployment in OSGi is a bundle. A bundle is a standard JAR or WAR file with additional metadata in the manifest file. Because the standard Java environment ignores these additional properties, you can also use an OSGi bundle with classic Java applications.

There are several bundles and bundle archives within an OSGi application:

- ▶ Enterprise bundle archives

An *enterprise bundle archive* file contains a set of OSGi bundles that are deployed as a single OSGi application and isolated from other OSGi applications. Bundles that belong to an OSGi application can reference other bundles that are in the shared bundle repository. The external bundles referenced do not have to be included within the application as long as the originating bundles import the required packages and the external bundles export these packages as well.

- ▶ Composite bundles

A composite bundle actively groups shared bundles together into a *composite bundle archive* file. This grouped bundle acts as a single bundle from the user perspective. It provides one or more packages at *exact* versions (not a version range) to an OSGi application. When the OSGi framework resolves a package to a bundle within a composite bundle archive, it has the affinity to resolve the remainder of the packages within the same composite bundle archive.

► Application bundles and shared bundles

Application bundles are instance-specific or *isolated*, and each instance of an application includes its own instance of the bundle. Shared bundles are *shared* or not instance-specific, and a single instance of a package or service from a shared bundle can be used by many applications.

► Web application bundles

A *web application bundle* is a bundle that contains a web application and that can be deployed in an OSGi container.

► EJB bundles

An EJB bundle is a bundle that contains EJBs and that can be deployed in an OSGi container. An EJB bundle is an OSGi bundle version of an EJB JAR file. It is a new feature introduced in WebSphere Application Server V8.5.

For more information about OSGi bundles and bundle archives, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=ca_bundles

OSGi Bundle Manifest file

The metadata for an OSGi application is defined in manifest files. An OSGi bundle contains a bundle manifest. A composite bundle archive contains a composite bundle manifest. An enterprise bundle archive contains an application manifest. An enterprise bundle archive asset contains a deployment manifest which is generated automatically when the enterprise bundle archive file is imported as an asset.

In this section we focus on the OSGi bundle manifest located under META-INF/MANIFEST.MF. For other manifest information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=ca_manifest

We now check each element of the example manifest shown as Example 26-1.

Example 26-1 An example of bundle manifest file, META-INF/MANIFEST.MF

```
Manifest-Version: 1.0
Bundle-ManifestVersion: 2
Bundle-Name: MyLibrary bundle
Bundle-SymbolicName: com.sample.mylibrary
Bundle-Version: 42.0.0
Bundle-Activator: com.sample.mylibrary.Activator
Import-Package: org.osgi.framework;version="[1.0.0,2.0.0)"
Export-Package:
com.sample.mylibrary.stringops;version=23.2.1,com.sample.mylibrary.integerops;version=5.0.0
Meta-Persistence: entities/persistence.xml,
lib/thirdPartyEntities.jar!/META-INF/persistence.xml
Web-ContextPath: /contextRoot
Export-EJB:
Bundle-Blueprint: /blueprint/*.xml
```

The following list explains the elements of the bundle manifest file from Example 26-1 on page 923:

▶ **Bundle-ManifestVersion**

This header must be set to the numeric value of 2. This value indicates that the bundle is written to revision 4 or later of the OSGi specification, rather than previous revisions.

▶ **Bundle-SymbolicName** and **Bundle-Version**

These two headers define the identity of the module. Every bundle in an OSGi system has a unique identity that is determined by its symbolic name and version. The **Bundle-Version** header is optional. If the header is not present, the bundle version defaults to 0.0.0.

▶ **Bundle-Activator**

This header notifies the bundle of lifecycle changes.

▶ **Import-Package**

This header defines the packages that are visible to a bundle. A bundle always has access to all `java.*` packages, all the packages inside the bundle, and depending on the OSGi framework configuration, the `javax.*` packages that are part of the JDK. All other packages must be imported. Every package import carries a version range that defines the accepted versions of the dependencies. This version range is entirely independent of the bundle version.

▶ **Export-Package**

This header declares the packages that are visible outside the bundle. Only the specified visible packages can be used by other bundles. Every exported package carries a version, which defaults to 0.0.0 if unspecified.

▶ **Meta-Persistence**

If your application uses the Java Persistence API (JPA) and this bundle is a persistence bundle, the bundle manifest also contains a **Meta-Persistence** header. This header lists all the locations of `persistence.xml` files in the persistence bundle. When this header is present the default location, `META-INF/persistence.xml`, is added by default.

▶ **Web-ContextPath**

This header identifies this bundle as a web application bundle. The value specifies the default context from which the web content is hosted.

▶ **Bundle-Blueprint**

This header specifies the location of the blueprint descriptor files in the bundle.

▶ **Export-EJB**

This header is new in WebSphere Application Server V8.5. It identifies this bundle as an EJB bundle. This header causes any enterprise beans in the bundle to be loaded and run by the EJB container. The value of this header declares the enterprise beans that you want to export as OSGi services.

The **Export-EJB** header can have any of the following values:

- A single space character: Export all enterprise beans in the bundle.
- A comma-separated list of the class names of the enterprise beans that you want to export. If an enterprise bean is not included in this list, it is still loaded and run, but not exposed in the OSGi service registry.
- NONE: Do not export any enterprise beans.

An exported enterprise bean is registered in the OSGi service registry with the following service properties:

- `ejb.name`: The name of the enterprise bean.
- `ejb.type`: The EJB type with the value of this property as either *Stateless* or *Singleton*.
- `service.exported.interfaces`: The enterprise bean is registered with this property only if it has a remote interface. The value is the EJB interface name.

For more information about the Export-EJB header, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=ra_bundle_mf

OSGi versioning

OSGi versioning is the foundation for the OSGi modules *dependencies* and *capabilities*. Every bundle or export package (capabilities) has a *version number* in a specific format, and every import package (dependencies) statement has a *version range*.

This versioning provides the ability to write modules today that can interoperate with current libraries and with future versions of those libraries but that will fail to resolve against incompatible future versions of those libraries.

Note: OSGi distinguishes between bundle versions and package versions. Versioning a bundle does not version the packages and vice versa.

The OSGi semantic versioning follows this format:

`<major>.<minor>.<micro>.<qualifier>`

- ▶ The `<qualifier>` carries no semantics and is often used to denote build numbers. Such as in Rational Application Developer, the qualifier can be replaced by a build time stamp during export.
- ▶ A change in the `<major>` denotes a breaking API change, for example, the parameters of a method changed.
- ▶ A change in the `<minor>` denotes a backwards compatible API change, for example, adding a new method to an interface. Existing clients can function both with the old and new versions. Implementations can be updated to support the new method.
- ▶ A change in the `<micro>` denotes a bug fix, and no change to the API is allowed.

In addition to single versions, OSGi defines the concept of version ranges. You can refer to the version ranges that were used on the Import-`Package` statements in Example 26-1 on page 923. Version ranges come in the following forms:

- ▶ `[1.0.0,2.0.0)`
Defines a version range from 1.0.0 (inclusive) to 2.0.0 (exclusive). A square bracket denotes an inclusive endpoint, whereas a round bracket denotes an exclusive endpoint.
- ▶ `1.0.0`
Defines an open version range of 1.0.0 and continues through higher versions. Do not confuse open version ranges with single versions. Whether the version is a single version or a range is determined by the context. For example, the `Export-Package` and `Bundle-Version` statements have single versions, where as the `Import-Package` and `Application-Content` statements have version ranges.

Consideration: Despite operating productively in most development scenarios, the OSGi version policy is not appropriate in all cases. For example, OSGi versions are conceived as linear, that is, v1.7 must contain all the features of v1.6 (otherwise, a breaking change occurs and the version is actually v2.0). As a consequence, you can introduce new features only on the latest version within one major package version. You cannot introduce features at earlier versions. This limitation can be a problem with large products that maintain a stable API. Stable APIs have infrequent major version changes, but you still need to combine the product with their multiple minor versions that are supported and actively extended.

Also note that it is up to the developer to adjust package versions.

OSGi Class Loader

One key advantage of OSGi is its class loader, which uses the metadata in the manifest file to resolve the classes and resources. When bundles are installed into the OSGi Framework, their metadata is processed. The OSGi Framework works out all the dependencies and calculates the independent required class path for each bundle in the following ways:

- ▶ Each bundle provides visibility only to Java packages that it explicitly exports.
- ▶ Each bundle declares its package dependencies explicitly.
- ▶ Packages can be exported at specific versions and imported at specific versions or from a specific range of versions.
- ▶ Multiple versions of a package can be available concurrently to different clients.

For more information, refer to chapter 22.5, “OSGi class loaders” on page 810.

26.1.2 OSGi bundle lifecycle

All artifacts in OSGi are equipped with support of dynamics in the form of defined lifecycles. Figure 26-1 on page 927 shows the status of an OSGi bundle lifecycle.

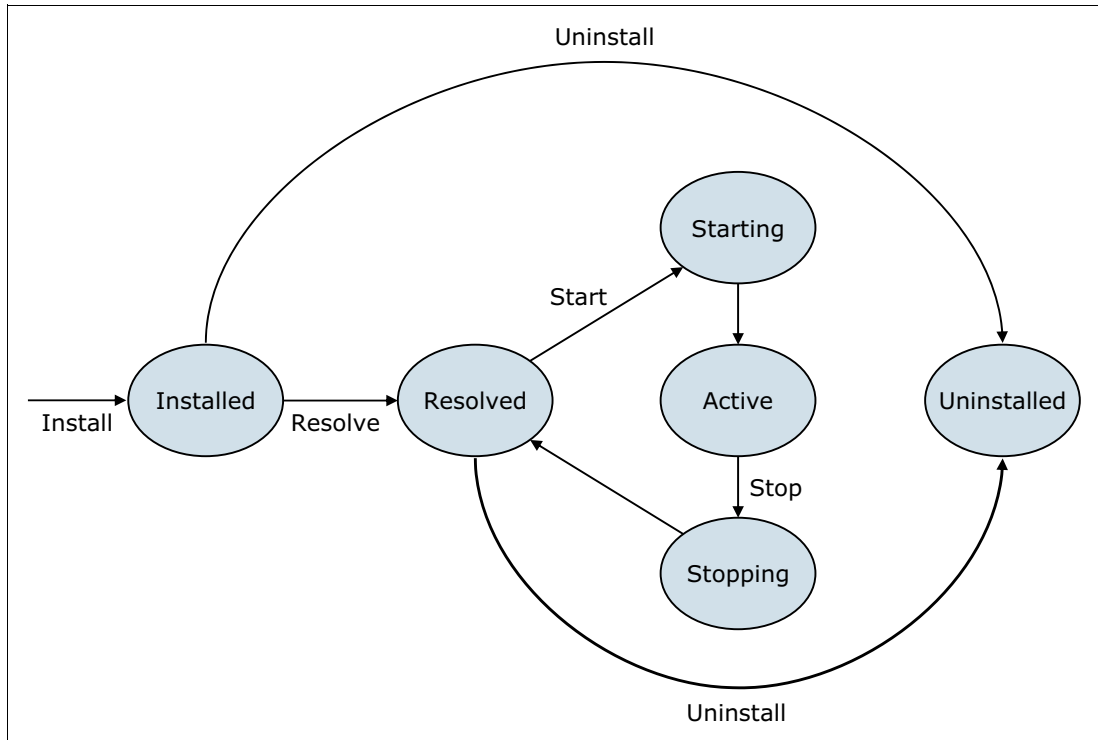


Figure 26-1 OSGi bundle lifecycle

The following list describes some the stages of the OSGi bundle lifecycle:

- ▶ A bundle is first *installed* into an OSGi runtime environment (called the OSGi framework).
- ▶ A process called *resolution* determines whether all of the bundles' dependencies can be satisfied.

If successful, the bundle moves to the *resolved* state and can be started. In this state, classes and resources in the bundle can be used by other bundles. The lifecycle layer ensures that bundles are started only if all their dependencies are resolved. It can reduce the occurrence of `ClassNotFoundException` exceptions at run time.

- ▶ It briefly goes through the *starting* state where the bundle's startup class, called an *Activator*, is executed. At that time, the bundle transitions into the *active* state. The bundle activator is specified in the bundle manifest.

The bundle activator allows an OSGi bundle to be more than just a provider of classes by actively executing tasks, registering and consuming services, and so on with other tasks.

- ▶ At the end of its life, a bundle can be uninstalled from the framework again.
However, when uninstalling a bundle, OSGi ensures that any packages that the bundle provides to other bundles remain available. OSGi verifies if there any states that still need to be resolved or are active.

Bundles can be installed, started, stopped, and uninstalled independently of each other and independent from the lifecycle of the application server.

OSGi bundles are not started unnecessarily but only when explicitly requested or first needed. Even when started explicitly, a bundle author can defer activation to when the first class is loaded from the bundle. This is called *lazy activation*.

Updates

Bundle and service lifecycles, along with the event notification support that OSGi defines around them, gives developers the tools to build truly dynamic applications. However, even with this support, it remains far from trivial to write code that can cope appropriately with a truly dynamic environment on which services can come and go at any time.

The OSGi model allows you to perform live updates of the module during which time the environment normally runs the applications. This method forces OSGi to cope with a bundle that might disappear and then re-appear or be present and active more than once for some interval. This method minimizes down time of server side applications but alternatively also requires a good OSGi design.

26.1.3 OSGi Service

OSGi also introduces a service registry layer used for collaboration between bundles. An OSGi service is published to the service registry under one or more Java interface names with any optional metadata stored as custom properties.

Bundles publish services to the service registry, and other bundles can discover them by looking up a service in the service registry. Bundles can filter service registry searches by using the interface name and custom properties.

OSGi Applications in WebSphere Application Server usually interact with the OSGi service registry through a Blueprint module definition. POJO bean components, that are described in the Blueprint module definition, can be registered as services through a `<service>` element or can have service references injected into them through a `<reference>` element.

Services are fully dynamic, and typically have the same lifecycle as the bundle that provides them. Provider bundles can be stopped and started causing POJO services to be registered and de-registered, independently of the lifecycle of the consuming bundles. However, bundles can choose to publish and retract services dynamically due to changes in the environment, for example, due to one required service going away or coming back.

26.2 Enterprise OSGi

OSGi was originally targeted to support Java Platform, Standard Edition, but Version 4.2 of the OSGi specification introduces additional support for Java Platform, Enterprise Edition.

These extensions allow developers to write an OSGi enterprise application using methods that are similar to writing Java EE applications. The following Java EE technologies are integrated with the OSGi framework:

- ▶ Web Application Specification

Defines how to support the Servlet 3.0 and JSP 2.1 specifications in OSGi. Bundles that are built on this support are called *web application bundles*. Web application bundles must be marked by the **Web-ContextPath** bundle manifest header.

For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=ca_wab

- ▶ JNDI Services Specification

Defines how OSGi bundles can access `javax.naming` services and how JNDI can be used to access the OSGi service registry.

► JPA Service Specification

Defines the basic support for unmanaged JPA in an OSGi bundle, called a *persistence bundle*, which must be marked by the Meta-Persistence manifest header. In particular, the specification defines packaging requirements around persistence bundles and provider selection or integration with the JPA run time.

► Blueprint Container Specification

Based upon the Spring dynamic modules project, Blueprint provides a light-weight, XML-based POJO injection model with special support for the OSGi service registry. The Blueprint XML files define and describe how the components are instantiated and are wired together to form a running module. A Blueprint XML file is used for the following actions:

- Declare beans by using the bean element.
- Declare a service element to define the registration of a service in the OSGi service registry.
- Declare a reference element to find services in the service registry or the reference-list element to find multiple matching services.

For more information about the Blueprint Container, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=/com.ibm.websphere.osgi.nd.doc/topics/ca_blueprint.html

The specifications allow developers to write an OSGi enterprise application using methods that are similar to writing Java EE applications. Because the OSGi model is non-invasive to the classic Java EE model, both applications can be used in the same run time.

26.3 Using the sample application

In this section, we use an ITSOBank sample application to explain the OSGi packaging model. To work with the OSGi application, we use IBM Assembly and Deploy Tools for WebSphere Administration v8.5. However, you can use any other tool that supports the OSGi framework.

26.3.1 Downloading the application

Complete the following steps to access and use the sample application:

1. To download the sample application, go to the following website:

<http://www.redbooks.ibm.com/abstracts/sg247835.html?Open>

2. Click the **Additional Material** link.
3. Click the **sg247835.zip** file, and select **Save** to save the compressed file to your computer.
4. Extract the compressed file, and find the OSGi_ITSO_example.zip file. This file is located in the 7835codesolution/osgi directory.

26.3.2 Importing the application to the development tool

To use the sample `OSGi_ITSO_example` application for our exercise, import the projects files into IBM Assembly and Deploy Tools for WebSphere Administration by completing the following steps:

1. Start IBM Assembly and Deploy Tools for WebSphere Administration.
2. Click **File** → **Import**, and select **Existing projects into workspace** under the General section. Click **Next**.
3. Click **Browse** next to the Select root directory, and point to the root folder where you extracted the `OSGi_ITSO_example.zip` file. Click **OK**.
4. The process discovers the five projects shown in Figure 26-2. Click **Select All** to select all of the projects. Select **Copy projects into workspace**.

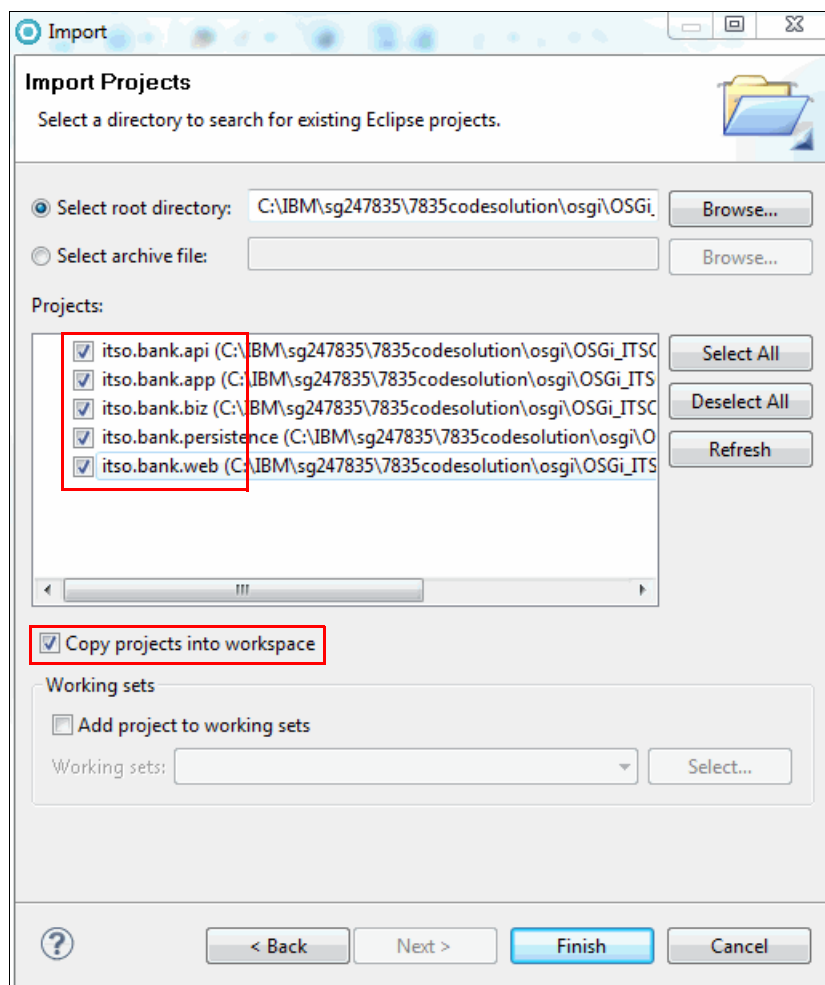


Figure 26-2 Import OSGi applications

5. Click **Finish**.
6. The IBM Assembly and Deploy Tools for WebSphere Administration promotes a Workspace Migration window. Click **Select All**, and make sure the check boxes of all five projects are selected, and click **Next**. In the Migration Project Resource window, click **Next**. In the Undefined Server Runtime window, as shown in Figure 26-3 on page 931, select the WebSphere Application Server V8.5 as **New Server Runtime** and accept all default settings and then click **Next**. In the Complete Migration Startup window, click

Finish. After the migration process complete, a dialog confirms that the migration completed successfully.

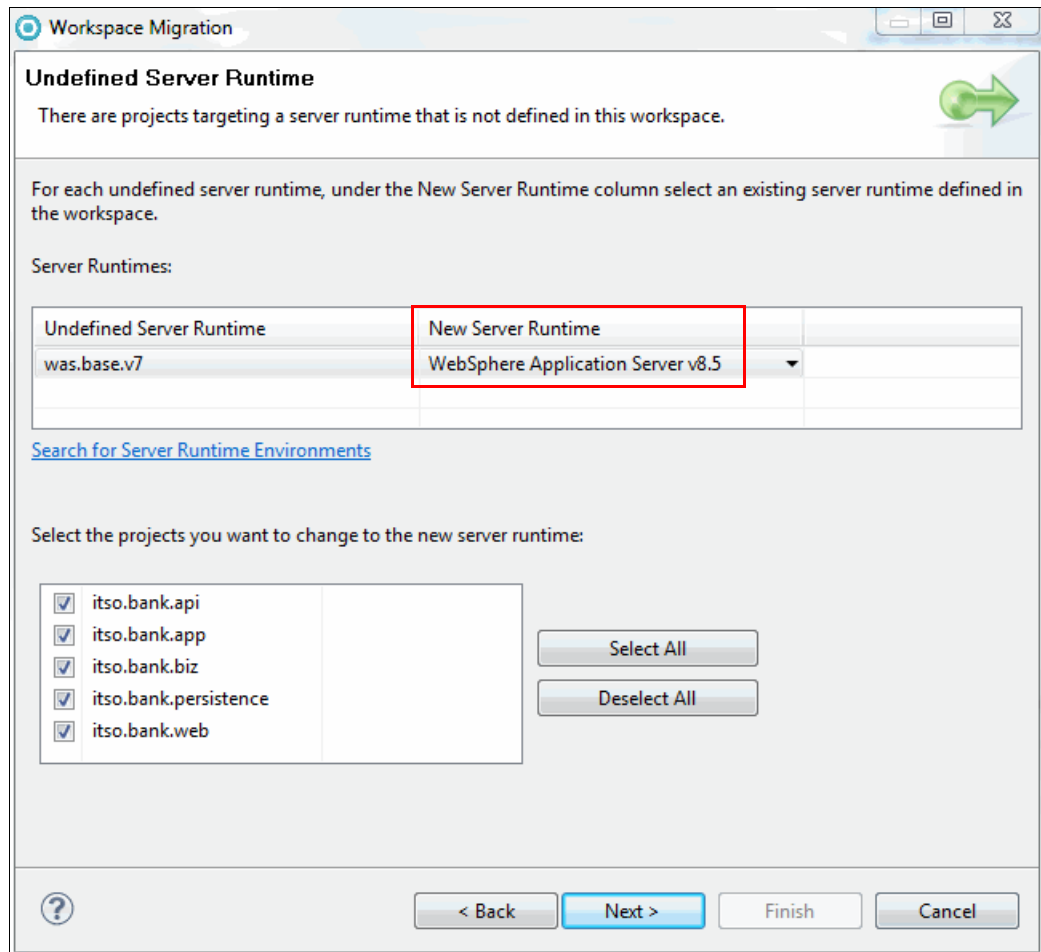


Figure 26-3 Select WebSphere Application Server V8.5 as new server run time

7. After the projects are loaded to your workspace, copy all the JAR files used by `itso.bank.web` bundle from the `OSGi_ITS0_example\jar files` directory. Then, in the Enterprise Explorer view of the workspace, expand the `itso.bank.web` application, and paste the files into the `WEB-INF/lib` directory. Figure 26-4 on page 932 shows the result. Note that this method is only one way to supply libraries to the web project.

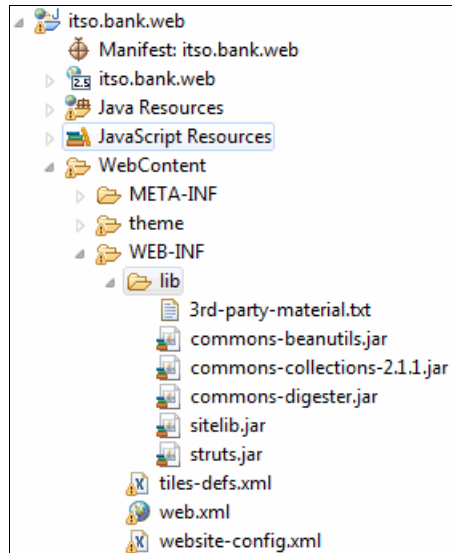


Figure 26-4 Copy the jar files used by itso.bank.web bundle

After importing the projects, you can use the Enterprise Explorer perspective to preview the ITSOBank application projects, as shown in Figure 26-5.

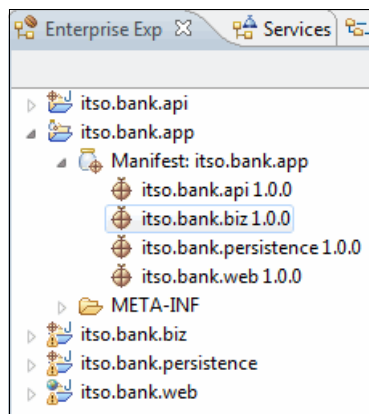


Figure 26-5 Sample itso.bank OSGi application

For more information about developing an OSGi application, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=thread_ta_dev_devdepeba

Note: To have a fully operational OSGi ITSOBank application, additional configuration is required, such as configuring the data source on the server. Refer to 23.6, “Preparing the runtime environment for the application” on page 846 and 23.6.5, “Creating a DB2 JDBC provider and data source” on page 853 for more information.

26.4 Packaging OSGi applications

This section covers the packaging of OSGi applications. There are several areas that can be affected by OSGi packaging, and this section considers patterns, importing, and how class loaders interact.

26.4.1 Common OSGi patterns

Consider using the following common OSGi patterns that have special support in the packaging of OSGi applications:

- ▶ Web content is packaged in web application bundles.
- ▶ Business logic is written as POJOs.
- ▶ Bundles share interfaces and services rather than concrete implementations.
- ▶ Other bundles depend on services.
- ▶ Persistence is achieved through JPA managed persistence.
- ▶ Declarative transactions are provided by a custom Blueprint extension.

26.4.2 Sample application packaging

The ITSOBank is a simple application, but it presents a good approach about how to design OSGi modules. This application is built from four bundles that follow a three-tier architecture of front end, business logic, and back end, as shown in Figure 26-6.

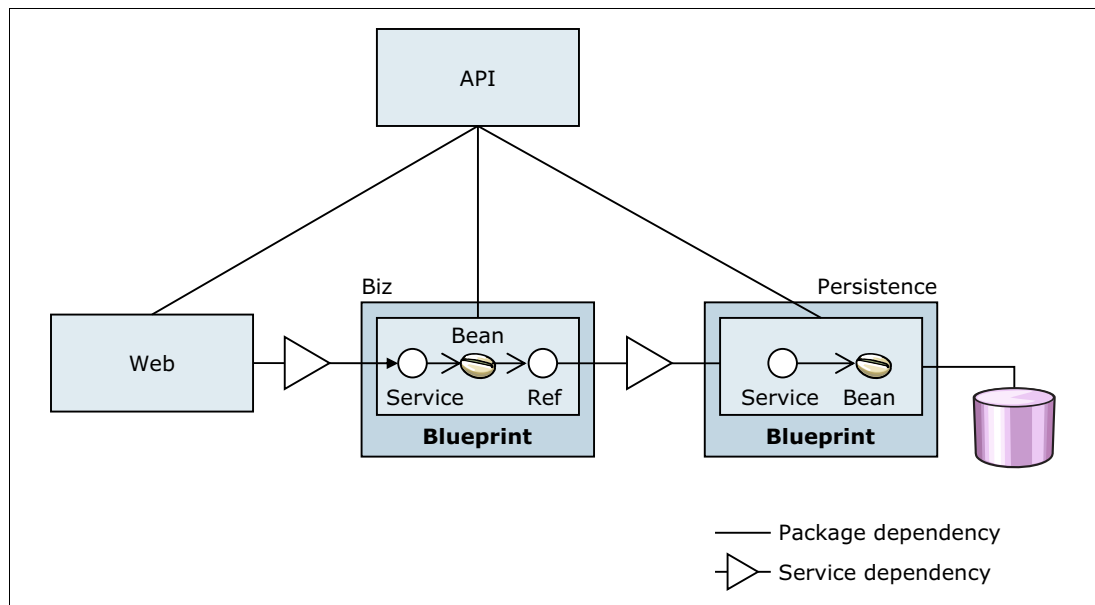


Figure 26-6 ITSOBank OSGi sample application architecture

If an OSGi application provides or requires any external services and references, these are explicitly made available by declaring them in an application manifest. The application manifest describes modularity at the application level in a similar way to OSGi headers in a bundle manifest file that define modularity at the bundle level.

A single main application project, `itso.bank.app`, holds a descriptor `META-INF/APPLICATION.MF` file, as shown in Example 26-2, which configures bundles that are used by the application.

Example 26-2 APPLICATION.MF of itso.bank.app application

```
Application-Name: itso.bank.app
Application-SymbolicName: itso.bank.app
Application-ManifestVersion: 1.0
Application-Version: 1.0.0.qualifier
Manifest-Version: 1.0
Application-Content: itso.bank.api;version="1.0.0",
```

```
itso.bank.biz;version="1.0.0",  
itso.bank.persistence;version="1.0.0",  
itso.bank.web;version="1.0.0"
```

The other projects define each bundle as follows:

- ▶ `itso.bank.api`
An API bundle that contains the interfaces that connect the web logic to the business logic and the business logic to persistence.
- ▶ `itso.bank.biz`
A business bundle that contains the business logic and acts as an intermediary between the presentation logic and the database.
- ▶ `itso.bank.persistence`
A persistence bundle that encapsulates the JPA-based database access pattern.
- ▶ `itso.bank.web`
A web bundle for servlets, JSPs, and static content that delegates the actual business functionality to the business bundle.

The three-tier application split is a standard pattern, but in an OSGi model, it contains a number of OSGi specific twists.

All of the implementation layers are connected through interfaces only. OSGi helps to make sure that concrete classes are not visible. Thus, individual module developers can only choose to use the provided interfaces. It is a best practice to minimize class dependencies and to ensure unit testability. To obtain concrete implementation of the service interfaces, module developers can use the OSGi service registry.

As a second step, we use a Blueprint to provide the fine-grained dependency injection that separates the wiring of business beans to and from services from their implementation.

Finally, there is the API bundle, which warrants a closer look. In a traditional Java EE style development, we might have included the interfaces in the bundles that also provide the implementation. Alternatively, in OSGi development, we focus more on deciding on packaging and module boundaries. The key criterias define responsibilities and frequency of change across the entire module lifecycle.

Providing the service or entity interfaces and implementing the interfaces are two separate concerns. For example, with the persistence interface (besides a JPA-based persistence implementation mechanism), we can investigate other mechanisms that accesses a no-SQL persistence store or a flat file. Separating the concerns earlier in the process pays off, rather than packaging the interfaces in each separate implementation.

Furthermore, the frequency of change also distinguishes interfaces and implementation. Interfaces change less often because the cost for change is high and because client code can be broken easily. Alternatively, the hidden implementation classes change more frequently without requiring interface changes, in particular, as result of defects. Thus, separate packaging conceptually means we can update the implementation without changing the interface bundles. Replacing interface classes can never be done seamlessly, but replacing a service implementation can.

26.4.3 Exporting OSGi applications

To export an OSGi application from the IBM Assembly and Deploy Tools for WebSphere Administration, complete the following steps from the Enterprise application view:

1. Select the main OSGi application (`itso.bank.app` in this case), right-click `itso.bank.app` and then click **Export** → **Export**.
2. From the context window, click **OSGi Application (EBA)** application type.
3. Enter the location of the exported file in the To EBA file field. The `.eba` extension is added automatically to this file. Click **Finish**. Figure 26-7 shows the OSGi application export window.

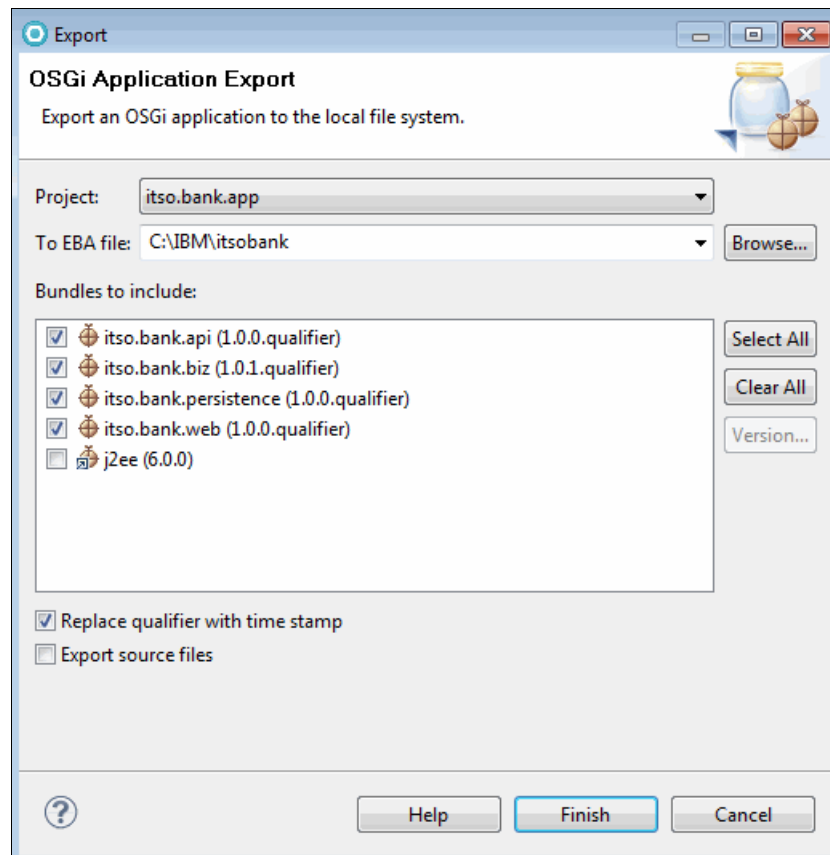


Figure 26-7 Exporting an OSGi application

An OSGi package for deployment is built and validated. In this case, a file called `c:\IBM\itsobank.eba` is created. You can inspect this file for its content and generated descriptor files. Notice that if you select the **Replace qualifier with time stamp** option, the generated versions and bundles name qualifier part is a time stamp, as shown in the following examples:

- ▶ Application-Version: `1.0.0.201206141321`
- ▶ `itso.bank.biz_1.0.0.201206141321.jar`

When generating an application without using this option, the files are saved with the following names:

- ▶ Application-Version: `1.0.0.qualifier`
- ▶ `itso.bank.biz_1.0.0.qualifier.jar`

You can also use other tools to build your OSGi applications, such as Ant or Maven, and automate them with `wsadmin` scripts. To learn more about these technologies, refer to the following web sites:

- ▶ <http://ant.apache.org>
- ▶ <http://maven.apache.org>

Additional considerations when exporting OSGi applications

Because of the complexities of the OSGi class loader, give special consideration to importing an exported package. When a package is exported, the package can be used by a separate bundle before the exported bundle is even started. This situation can be a problem when singletons and static fields are used because an imported class is loaded by a separate class loader than the class loader that is used by the bundle. This creates separate instances of the service object.

For example, suppose bundleA exports a service object that is responsible for generating sequential order numbers. Also, suppose that three other bundles import and use that service object for generating order numbers. Those three other bundles use the *framework class loader* and share the same instance of the service object. If bundleA also imports the package, it also uses the same instance. However, if bundleA does not import the package, the *bundle class loader* instantiates a new instance of the service object and possibly produces a duplicate list of order numbers.

So the practices noted indicate that you typically must import any packages that you export to reduce the number of copies of that package in memory and to ensure that the object instances come from the same class loader.

For more information about OSGi class loaders, refer to 22.5, “OSGi class loaders” on page 810.

26.5 Deploying OSGi applications

You can deploy an OSGi application by adding an enterprise bundle archive asset to a business-level application. The deployment can be done using the administrative console or `wsadmin`. Installing through the administrative console is useful when dealing with complex application structures involving multiple versions and sharing. Using the administrative console for the first installation is also useful for capturing script commands for future use with `wsadmin`.

OSGi applications are packaged in enterprise bundle archive files. Deploying applications packaged this way involves completing the following steps:

1. Import the enterprise bundle archive file as an asset.
2. Add that asset to the business-level application as a composition unit.

26.5.1 Importing the enterprise bundle archive file as an asset

To import the enterprise bundle archive file as an asset:

1. In the administrative console, click **Applications** → **Application Types** → **Assets**. Click **Import**, and select the `itsobank.eba` file exported in 26.4.3, “Exporting OSGi applications” on page 935. Click **Next**.
2. In the wizard, as shown in Figure 26-8 on page 937, accept all the defaults and then click **Next**.

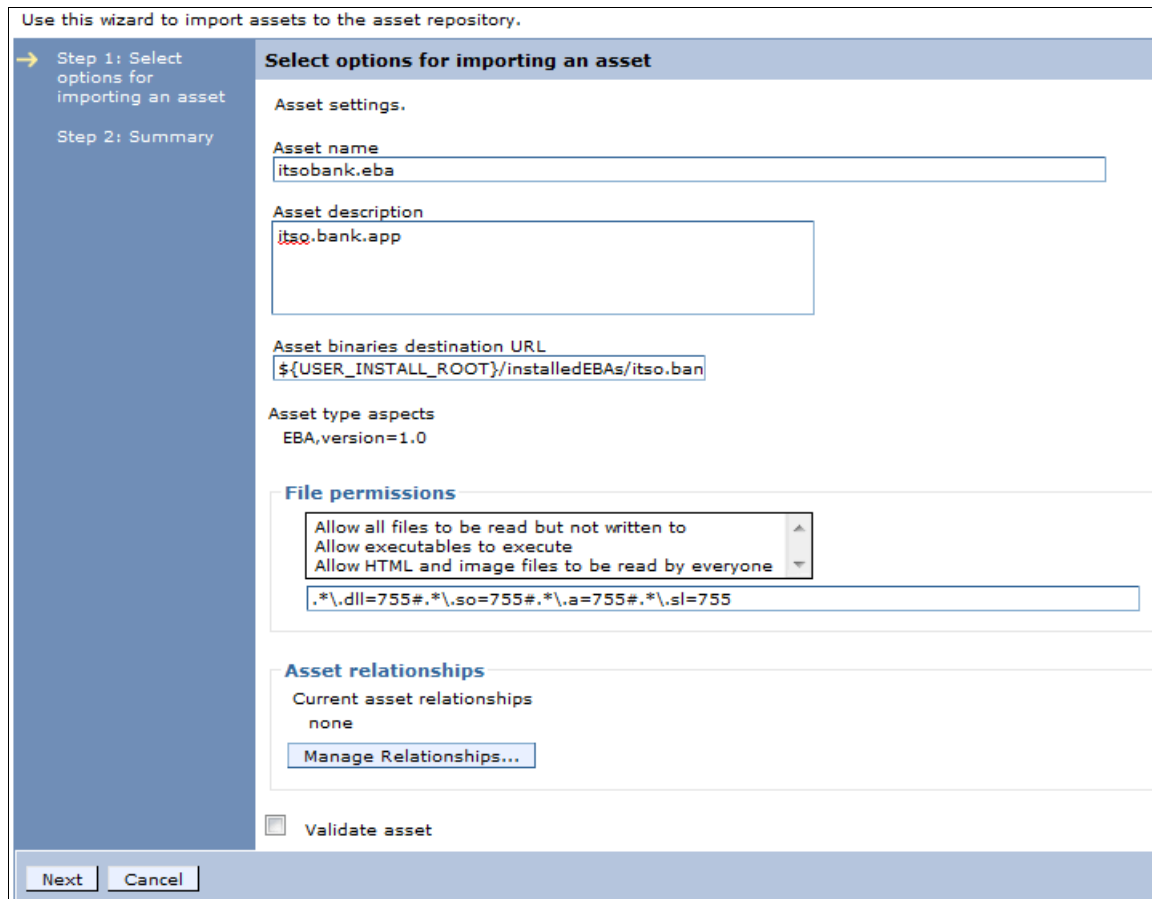


Figure 26-8 Import asset wizard

3. In the summary step, click **Finish** to import the asset. Enterprise bundle archive asset conversion and resolution warnings or other information might display in the upper-left corner.
4. After processing is complete and without error, save your changes. For OSGi applications, provisioning happens when importing the asset. As a result of provisioning, bundles might need to be retrieved from a bundle repository (internal or external). This process needs to be complete before the asset can be added to a business-level application. However, downloads are triggered only after the asset import is saved. Thus, saving the changes after importing the asset is required.

26.5.2 Adding the enterprise bundle archive asset to the business-level application

To add the enterprise bundle archive asset to the business-level applications:

1. Click **Applications** → **Application Types** → **Business-level applications**.
2. Click **New** to create a new empty business-level application. Enter a name for the application. ITSOBank Systems is used in this example, as shown in Figure 26-9 on page 938. Click **Apply**.

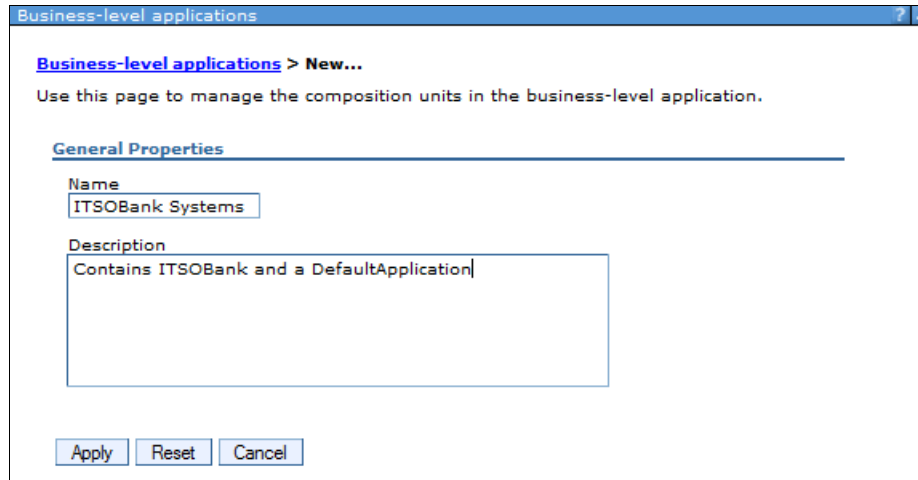


Figure 26-9 Create a new Business-level application

In the business-level application detail panel, click **Add** → **Add Asset** in the Deployed Assets section, as shown in Figure 26-10.

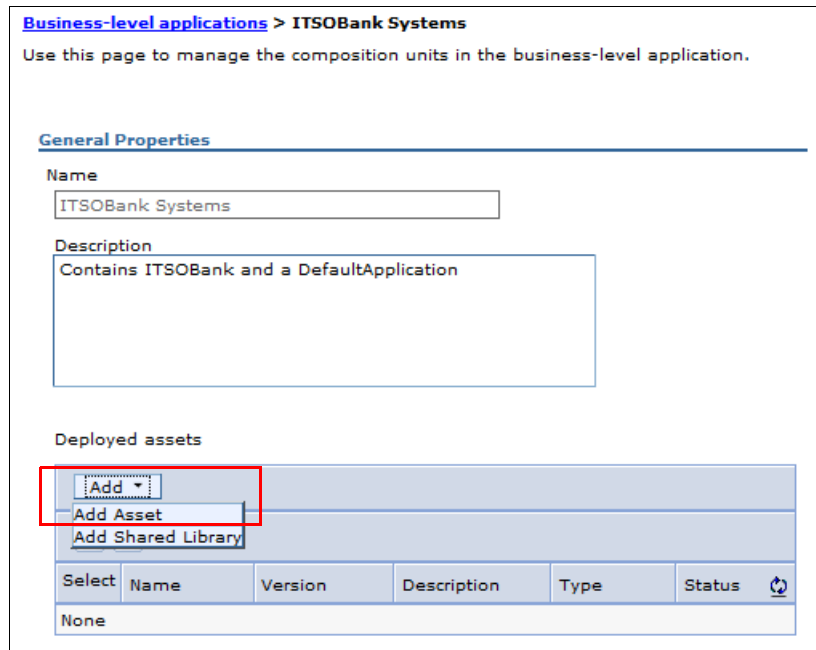


Figure 26-10 Add asset to business-level application

- Choose the asset that was created in 26.5.1, “Importing the enterprise bundle archive file as an asset” on page 936 from the list of assets, as shown in Figure 26-11. Click **Continue**.

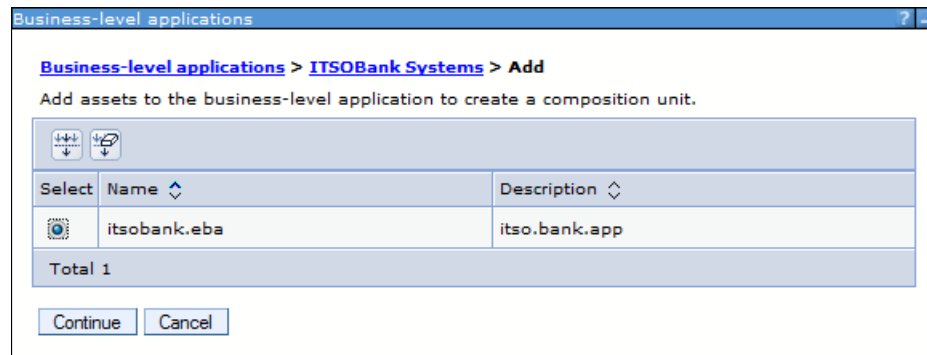


Figure 26-11 Add deployed asset

- A wizard starts the new composition unit that will be created. In the wizard, the first step is to set the options for the composition unit, as shown in Figure 26-12.

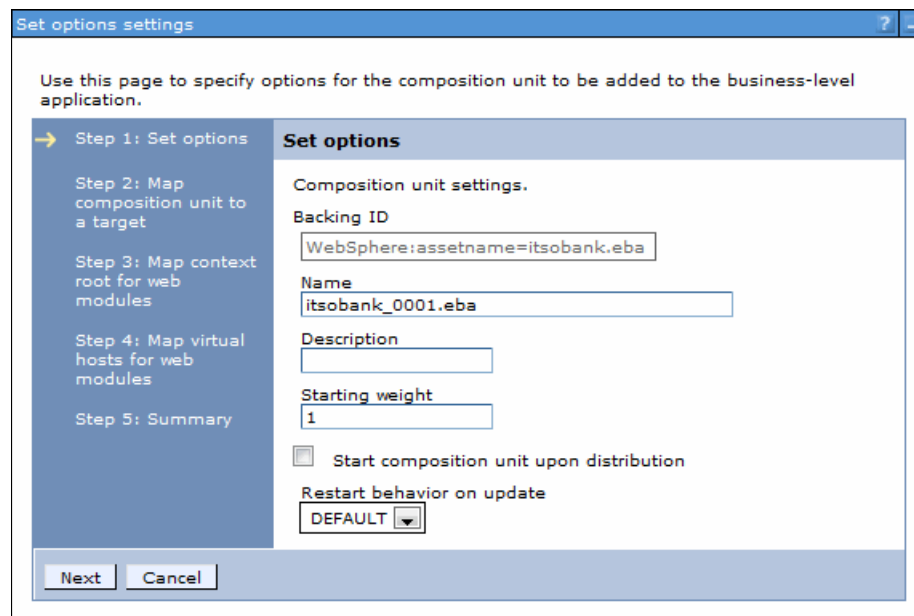


Figure 26-12 The add asset wizard

In the asset wizard, the following fields are displayed. The following list describes the fields:

- Backing ID: Displays the unique identifier for the composition unit that is registered in the application domain. You cannot change this setting.
- Name: Name for the composition unit.
- Starting weight: Specifies the order in which composition units are started when the server starts. The composition unit with the lowest number is started first.
- Start composition unit upon distribution: Specifies whether to start the composition unit when it is distributed to other locations. This setting applies only to assets and shared library composition units.

- Restart behavior on update (of the composition unit):
 - ALL: Restarts the composition unit after the entire composition unit is updated.
 - DEFAULT: Restarts the composition unit after the part of the composition unit is updated.
 - NONE: Does not restart the composition unit after the composition unit is updated.
- 5. Click **Next**.
- 6. While still in the asset wizard, its second step is to map the composition unit to a target. Select the server on which the `itsobank.app` application will run and then click **Next**.
- 7. The third step within the asset wizard is to map the context root for the web modules. Select the *Context Root* to which the `itsobank.app` is to be mapped (default is `/itsobank`) and then click **Next**.
- 8. The fourth step within the asset wizard is to map the virtual hosts for the web modules. Select the desired virtual host and then click **Next**.
- 9. In the Summary page of the wizard, click **Finish**.
- 10. **Save** the changes.

Now you can start the business-level application. There are other wizard steps depending on the bundles and archives contained in your OSGi application. For more information about this topic, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-iserics&topic=thread_ta_dev_acu_console

Note: There are some restrictions for OSGi application deployment:

- ▶ An enterprise bundle archive file can be imported into only one asset.
- ▶ An enterprise bundle archive asset can be added to only one business-level application.
- ▶ One or more composite bundle extensions can be added to a composition unit.

You can also use `wsadmin` to deploy the OSGi application, it is the common method used in a production environment. For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-iserics&topic=ta_dep

26.6 Administrating OSGi applications

In this section, we focus on the OSGi application update, adding a new bundle to the bundle repository, and OSGi application security. For more information about the OSGi application administration, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-express-dist&topic=ta_admin

26.6.1 Updating OSGi applications

After you import your OSGi application as an asset, newer versions of the bundles or composite bundles that the asset uses might become available. You can configure the deployed asset to use an updated version, specific bundle version, or to pull in the latest

compatible version. WebSphere Application Server does not update the asset to a newer version automatically.

Before you update an OSGi application, ensure that the new update bundles have a valid and updated version value. If the new bundle version is the same as the old bundle version, the OSGi run time treats the original version and the new version interchangeably (despite potentially different build numbers).

In this example, we update the ITSO Bank OSGi application described in 26.4, “Packaging OSGi applications” on page 932.

Adding the updated bundle or composite bundle to a bundle repository

To update the ITSO Bank application, the new bundle is added to the internal OSGi bundler repository by completing the following steps:

1. In the administrative console, as shown in Figure 26-13, click **Environment** → **OSGi bundle repositories** → **Internal bundle repository**.

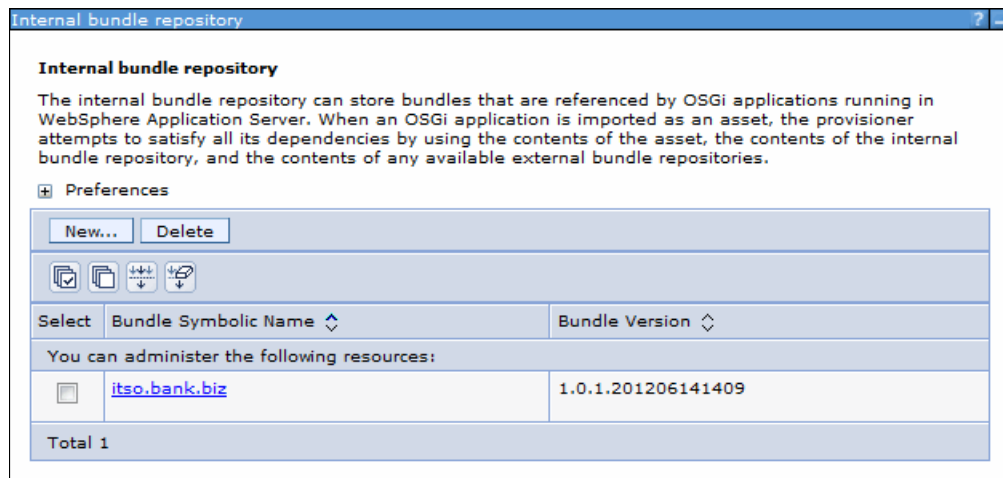


Figure 26-13 Add new bundle to the Internal bundle repository

2. Click **New**, and in the Specify path field, enter the path to the new bundle. You can choose to specify a path on your local computer or on the remote network manager environment.

In this case, we use the itso.bank.biz bundle Version 1.0.1 that was exported from IBM Assembly and Deploy Tools for WebSphere Administration. This bundle is a single JAR file. The results are also previously displayed in Figure 26-13.

3. Click **OK**, and save the configuration.

For information about administering the bundle repository, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-express-dist&topic=ta_admin_obr

Updating bundle versions in a deployed OSGi application

After the new bundle is added to the repository, configure the ITSO Bank application to use it explicitly, using the following steps:

1. Click **Applications** → **Application Types** → **Assets**, and choose the ITSO Bank application.
2. Under the Additional Properties section, click the **Update bundle versions in this application** link.

- From the new form, choose the new version of the itso.bank.biz bundle, as shown on Figure 26-14. Click **Preview**.

Assets > itsobank.eba > Update bundle versions in this application
Update the versions of the bundles that comprise this application.

Application bundle content

| Symbolic Name | Content Type | Sharing | Deployed Version | New Version |
|-----------------------|--------------|----------|--------------------|--------------------|
| itso.bank.api | Bundle | Isolated | 1.0.0.201206141321 | No preference |
| itso.bank.biz | Bundle | Isolated | 1.0.0.201206141321 | 1.0.1.201206141409 |
| itso.bank.persistence | Bundle | Isolated | 1.0.0.201206141321 | No preference |
| itso.bank.web | Bundle | Isolated | 1.0.0.201206141321 | No preference |

Preview Cancel

Figure 26-14 Update bundle version within application

Select the version: The drop-down menus list all the available versions plus the No preference option. Selecting a specific version instructs the provisioning system to use exactly that version for the update. Selecting **No preference** instructs the provisioning system to find the highest version that works with the other selections.

With both selections, the new version must fall into the range that is specified in the application manifest. For example, the following application content specifies that only bundle versions from 1.0.0 but less than 2.0.0 are admissible during provisioning and during updates:

Application-Content: itso.bank.biz;version="[1.0.0,2.0.0)"

- If no errors occur during processing, a message displays that selected bundle versions can be resolved, as illustrated in Figure 26-15. Click **Create**.

Assets > itsobank.eba > Update bundle versions in this application > Preview
A preview of the result of the proposed changes to the bundle versions in this application.

The selected bundle versions can be resolved, so you can now create a new deployment with the proposed bundle versions. The new deployment will not affect any composition units for this asset until the composition units are updated to use the new deployment.

Application bundle content

| Symbolic Name | Deployed Version | New Version |
|-----------------------|--------------------|--------------------|
| itso.bank.api | 1.0.0.201206141321 | 1.0.0.201206141321 |
| itso.bank.biz | 1.0.0.201206141321 | 1.0.1.201206141409 |
| itso.bank.persistence | 1.0.0.201206141321 | 1.0.0.201206141321 |
| itso.bank.web | 1.0.0.201206141321 | 1.0.0.201206141321 |

Create Cancel

Figure 26-15 updating the OSGi bundle on the WebSphere Application Server

At this time, the configured bundles are downloaded from repositories, in this case from an internal OSGi repository on WebSphere Application Server. The OSGi run time must cache the bundles locally so that applications are not affected by changes to the bundle repositories. This process of downloading the new versions to create local copies can take a small amount of time with external repositories or large bundles, but is mostly instantaneous when working with the internal bundle repository. You can check the status of downloads from the asset detail window or the composition unit detail window.

5. **Save** your changes to the master configuration.

Checking the status of the OSGi composition unit and update

You can check the update status of the OSGi composition unit in the administrative console:

1. Click **Applications** → **Application Types** → **Business-level applications** → **application_name** → **composition_unit_name**.
2. Check the deployment status. It is displayed under General Properties and then OSGi application deployment status, as illustrated in Figure 26-16.



Figure 26-16 OSGi application deployment status

The OSGi composition unit status: There are four distinct deployment statuses for an OSGi composition unit:

- ▶ Using the latest OSGi application deployment.
- ▶ New OSGi application deployment not yet available because it requires bundles that are still downloading.
- ▶ New OSGi application deployment available.
- ▶ New OSGi application deployment cannot be applied because bundle downloads have failed.

3. If you plan to update the composition unit at this time, you can update the OSGi composition unit so that the business-level application uses the newer configuration. Click **Update to latest deployment** under **Business-level applications** → **ITSOBank application** → **ITSOBank eba asset**.

For more information about updating OSGi applications (for example, using `wsadmin` scripting), refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=thread_ta_admin_maint

26.6.2 Securing OSGi applications

Securing OSGi applications in WebSphere Application Server is similar to securing enterprise applications. The following options are available for use:

► Using application security with OSGi applications:

- Modify the security role to user or group mapping.

In the administrative console, Click **Applications** → **Application Types** → **Business-level applications** → *application_name* → **[Deployed assets] Add** → **Add Asset** → *asset_name* → **Wizard step: Map security roles to users or groups**

- Use application security with web application bundles.

Define security constraints in the web.xml file for a web application bundle.

- Configure bean security in the Blueprint XML file.

This is new feature introduced in WebSphere Application Server V8.5. You configure security by defining one or more `<access-constraint>` elements, inside the `<bean>` element of Blueprint XML file of your OSGi application, as shown in Example 26-3.

Example 26-3 An example of bean security of an OSGi application

```
<bean id="secureBean1" class="com.sample.secureBeanImpl">
  <sec:access-constraint role="ROLE1" />
  <sec:access-constraint method="getID" role="ROLE2" />
</bean>
```

The two levels of security configuration are:

- Configuring bean-level security: In Example 26-3, the methods of the secureBean1 bean are accessible only by users that are assigned the role called ROLE1.
 - Configuring method-level security: In Example 26-3, the **getID** method of the secureBean1 bean is accessible only by users that are assigned ROLE2.
- Use application security with EJB bundles.

Enforce any bean method security settings in the ejb-jar.xml file for an EJB bundle.

► Use Java 2 security with OSGi applications.

Similar as Java 2 security in enterprise applications, WebSphere Application Server allows you to have a permissions.perm file in the META-INF directory of the OSGi application. This permissions.perm file applies fine-grained control of the permissions for each bundle.

For more information about securing OSGi application, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-iseries&topic=ta_sec



Working with Service Mapping

This chapter provides information about WebSphere Application Server V8.5.5 service mapping capabilities.

This chapter includes the following topics:

- ▶ Service mapping overview
- ▶ Local mapping service
- ▶ Administration for target service clients
- ▶ Event emissions

For additional information, see the following topic in the WebSphere Application Server Information Center:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=csmwas_servicemappingintro

27.1 Service mapping overview

Service mapping allows you to route and transform requests and responses between service clients and service providers. Based on the service mapping definition, the application can choose which service provider will handle the request or response.

Prior to WebSphere® Application Server V8.5.5, a service client can connect to a service provider only when it is configured with the endpoint address of the service provider. To reroute a request to a different service provider, you must configure the service client manually. The service provider and service client must also use exactly the same interface specifications for the service calls to function.

Figure 27-1 shows the interface with static routing. In this figure, a specific service client is configured to interact with a specific service provider. When a new service provider is introduced, the service client must be configured manually to send requests to the new service provider.

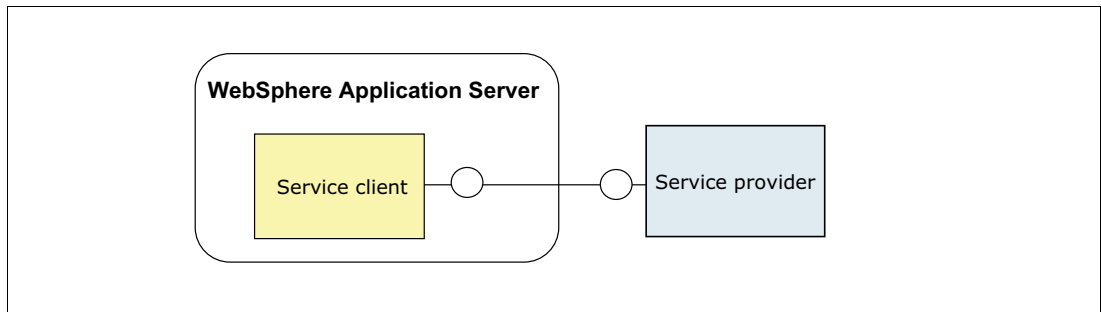


Figure 27-1 Static routing

Note: In Figure 27-1 and subsequent diagrams, interfaces are indicated by shapes: circles, squares, triangles, and hexagons. An identical shape on both the service client and the service provider means that their interface types are the same.

In V8.5.5, the service mapping feature, illustrated in Figure 27-2, is introduced to allow these interactions between service clients and service providers to be more dynamic. By using the service mapping feature, requests and responses can be intercepted, transformed, and rerouted.

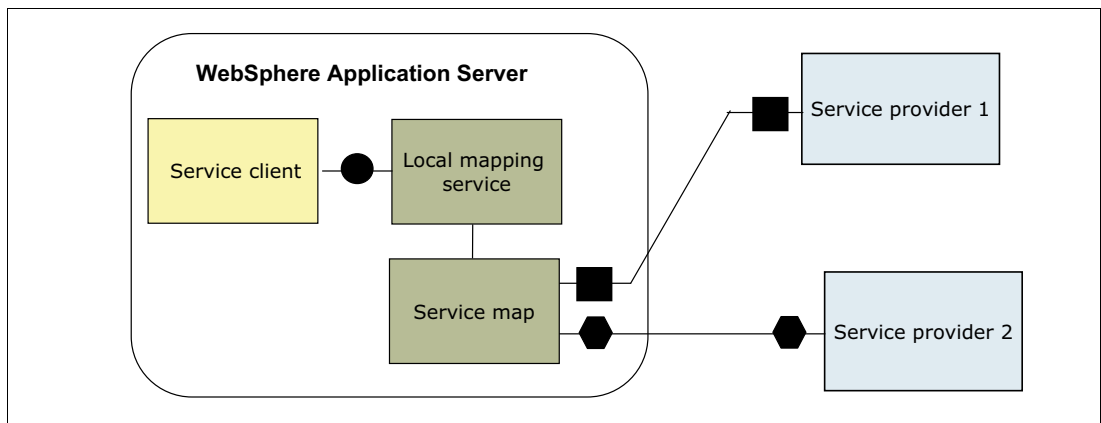


Figure 27-2 Service mapping with routing and transformation

As shown in Figure 27-2 on page 946, service mapping involves the following two objects:

- ▶ A service map: Defines the conditions for routing and transforming requests and responses that are intercepted by the local mapping service.
- ▶ A local mapping service: Intercepts requests and responses between service clients and service providers.

27.1.1 Service maps

A *service map* contains the information that a local mapping service uses to transform and route requests and responses between service clients and service providers. You develop a service map by using the Service Mapping Editor in Rational Application Developer. Then, you export that service map for use in the WebSphere Application Server run time.

The service map is packaged into a service map library, which is stored as a .slibzip file. This file is composed of XML files (WSDL, XSD, and SVRCMAP). The .slibzip file can also contain MAP files that are produced by the DFDL editor in Rational Application Developer, which defines the message-level mappings. You can check the service map file content by using the administrative command `inspectServiceMapLibrary` for the AdminTask object; see Example 27-1.

Example 27-1 inspectServiceMapLibrary command

```
wsadmin.sh -lang jython
```

```
wsadmin>AdminTask.inspectServiceMapLibrary('-source /tmp/labwork.slibzip')
```

```
'{srcvmapFile=labwork/NewServiceMap.srvvmap, name=NewServiceMap,
sourceService={name=source, namespace=http://test, portType=MyServiceDelegate,
binding=MyServicePortBinding}, targetServices=[{name=target,
namespace=http://test, service=MyEnhancedServiceService,
port=MyEnhancedServicePort, binding=MyEnhancedServicePortBinding,
portType=MyEnhancedServiceDelegate,
endpointURL=http://localhost:9080/DemoJAXWSProject/MyEnhancedServiceService}]}'
wsadmin>
```

Installing a service map

You can install a service map by using the administrative commands. Example 27-2 shows how to install a service map by using the command line.

Example 27-2 Installing a service map by using the command line

```
wsadmin>AdminTask.installServiceMap('[-sourceLibrary C:\code\labwork.slibzip
-sourceServiceMap labwork/LabWorkMap.srvvmap -name LabWorkMap -description
"Service map for lab" -deploymentTargets
WebSphere:cell=IBM-VIH218VES91Cell01,node=IBM-VIH218VES91Node01,server=server1
-targetServiceEndpoints [[target
http://localhost:9080/DemoJAXWSProject/MyEnhancedServiceService]]')
```

```
wsadmin>AdminConfig.save()
```

```
wsadmin>AdminTask.listServiceMaps()
```

```
'LabWorkMap'
```

```
wsadmin>AdminTask.showServiceMap('LabWorkMap')
```

```
'{name=LabWorkMap, description=Service map for lab,  
blaname=WebSphere:blaname=LabWorkMap, deploymentTargets=WebSphere:node=IBM-VIH218VES91Node01,server=server1,  
sourceAvailableServices=[MyServiceService], sourceNamespace=http://test, source  
AvailablePorts={MyServiceService=[MyServicePort]},  
sourcePortType=MyServiceDelegate}'
```

The AdminTask object provides the following commands for administering the service map:

- ▶ installServiceMap
- ▶ uninstallServiceMap
- ▶ listServiceMaps
- ▶ showServiceMap
- ▶ inspectServiceMapLibrary

To deploy a service map to WebSphere Application Server by using the administrative console, carry out the following steps:

1. Navigate to **Service integration** → **Service mapping** → **Service maps**. The Service map collection panel is displayed.
2. Click **Install**. Then, the Service maps - Install panel is displayed.
3. Select a service map library file (*.slibzip) from either the local file system or from a remote file system (where your WebSphere Application Server has been installed). Click **Next**.
4. The wizard for installing service maps from the service map library is displayed.

The first step of the wizard is Configure service map properties. If the service map library contains a single service map, the Filename field will be automatically filled with the name of the service map file. If the service map library contains multiple service maps, choose the service maps you want to install by selecting them from the Filename drop-down menu and ensure that the “Install this service map” check box is checked. By default, the “Install this service map” check box is selected for every service map in the service map library. If you do not want to install a specific service map, select the service map from the Filename drop-down menu and clear the check box.

You can specify a name and a description for each service map that you want to install. You can also override the target service’s endpoint address on this panel.

Figure 27-3 on page 949 illustrates how to configure the service map properties for a service map library file that includes multiple service maps.

Click **Next**.

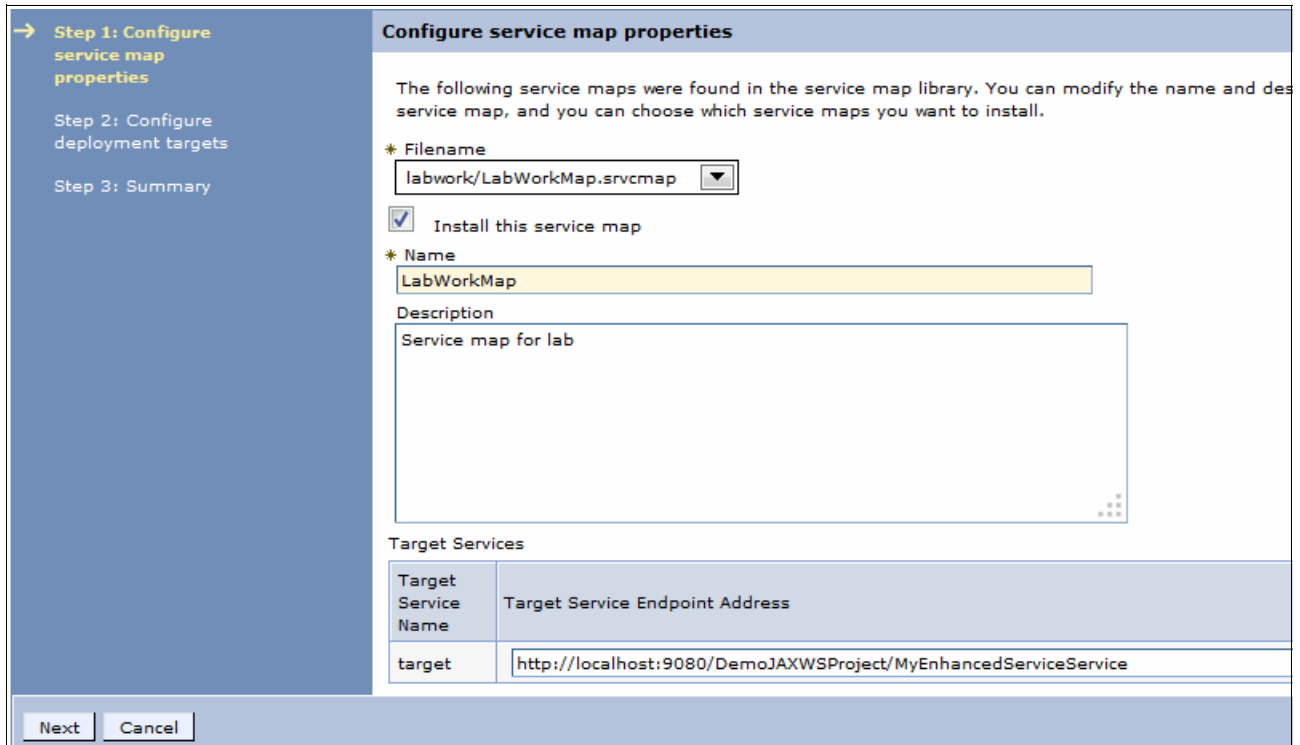


Figure 27-3 Configure service map properties

5. The next step in the wizard is Step 2: Configure deployment targets. If you are using a network deployment environment with several clusters or managed servers, these clusters or servers are listed in this panel. You must choose the deployment targets to which you want the service maps to be installed. A service map must be deployed to the same deployment target as the client whose service requests are to be intercepted. Click **Next**.
6. The Summary panel is displayed, showing the name and description of the service maps and their installation status. The deployment targets, associated with the service maps that are to be installed, are also shown (Figure 27-4).

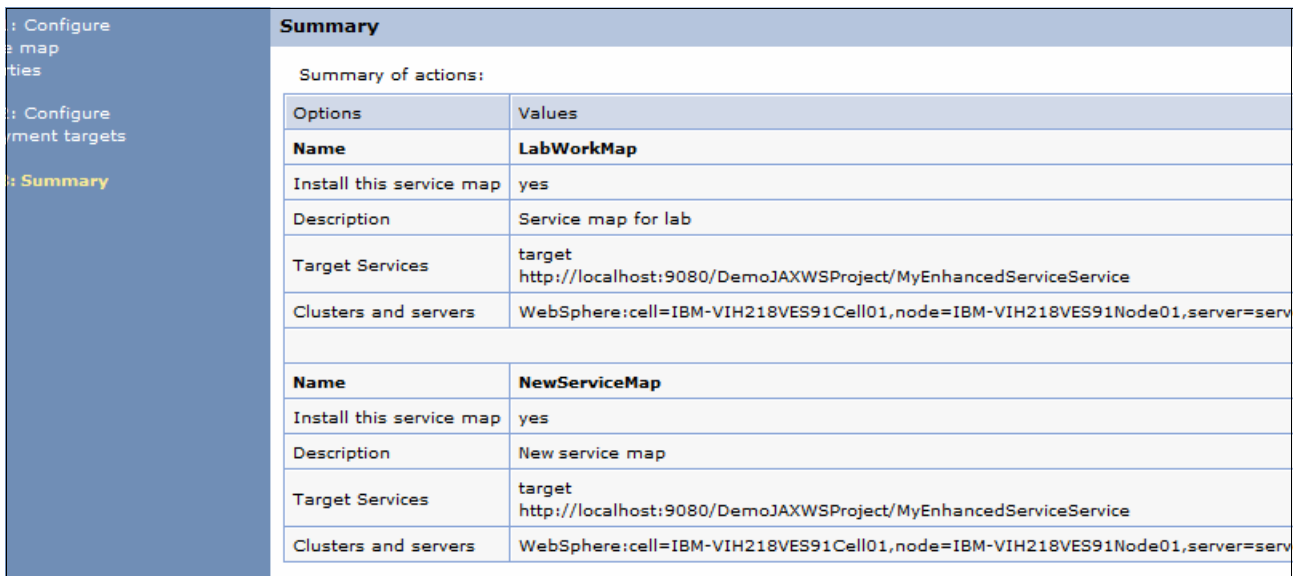


Figure 27-4 Install service map - Summary

7. Click **Finish** and save your changes to the master configuration.

27.2 Local mapping service

The *local mapping service* intercepts requests and responses between service clients and service providers. Each local mapping service must have a service map attached to it for the service mapping to work. If a local mapping service does not have an associated service map, the request is routed to the original target service specified by the service client.

27.2.1 Creating a local mapping service

To create a local mapping service, perform the following steps:

1. From the console, navigate to **Service integration** → **Service mapping** → **Local mapping services**. Then, click **New**.
2. Enter the name of the local mapping service and the description, and then, click **Next**.
3. Choose the service map that you want to attach to the local mapping service, as shown in Figure 27-5.

The screenshot shows a configuration window with a sidebar on the left containing four steps: 'Step 1: Specify local mapping service details', 'Step 2: Attach a service map (optional)' (highlighted with a blue arrow), 'Step 3: Specify details for interception of client requests', and 'Step 4: Summary'. The main content area is titled 'Attach a service map (optional)'. It contains a paragraph explaining that a service map defines request routing and transformation, and that it must be created using the Service Mapping Editor in IBM Rational Application Developer. Below this is another paragraph stating that a service map should be chosen from the 'Installed service maps' list. At the bottom of this section is a dropdown menu labeled 'Installed service maps' with 'NewServiceMap' selected. At the very bottom of the window are three buttons: 'Previous', 'Next', and 'Cancel'.

Figure 27-5 Attaching a service map

If you choose to attach a service map, in the next panel, you only need to enter the endpoint address of the service provider.

In this example, the NewServiceMap is selected. Click **Next**.

4. In the following panel, you must specify the intercepted provider details as shown in Figure 27-6 on page 951, and then, click **Finish**. Because a service map was selected in the previous panel, only the endpoint access information needs to be entered. The local mapping service only intercepts requests sent from Java API for XML Web Services (JAX-WS) service clients that try to access a service by using the supplied endpoint address.

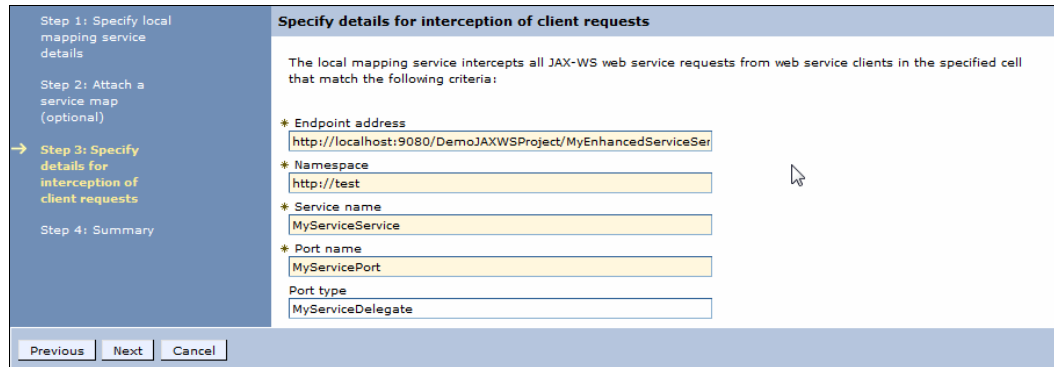


Figure 27-6 Client request intercept information

5. The summary will appear as shown in Figure 27-7.

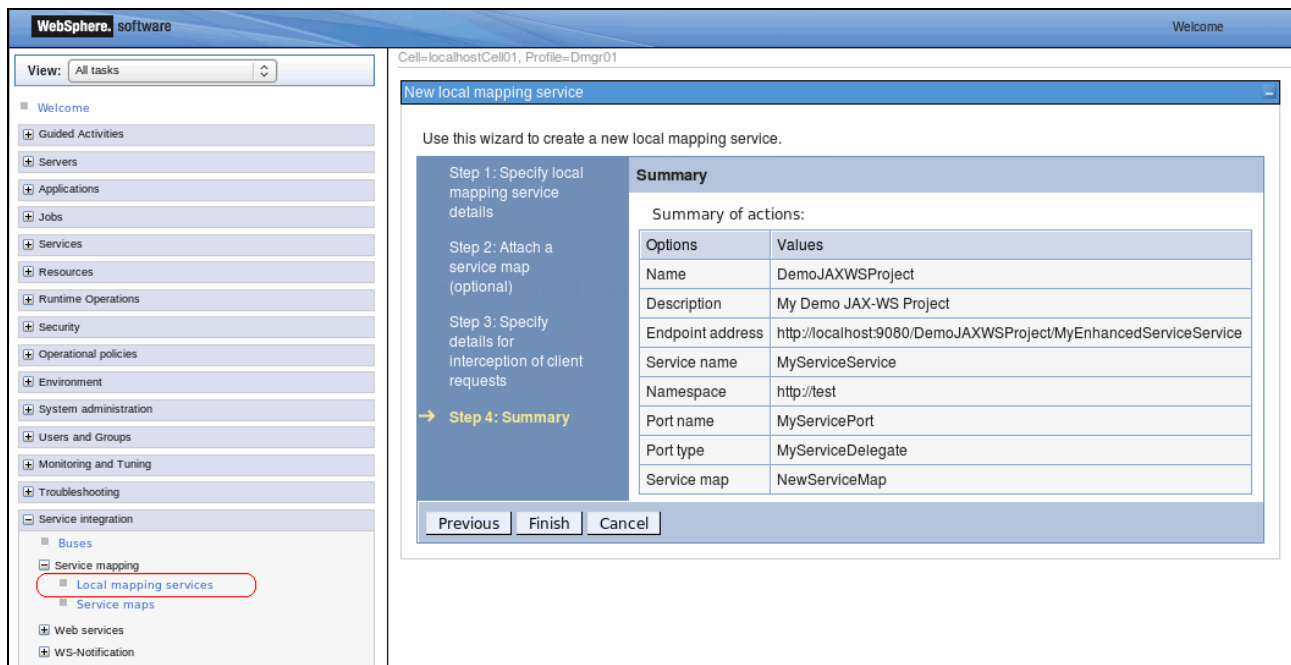


Figure 27-7 Create local mapping service

You also can create a local mapping service by using administrative commands with the `wsadmin` tool (see Example 27-3):

```
AdminTask.createLMService('[-name DemoJAXWSProject -description "My Demo JAX-WS Project" -attachSMName NewServiceMap -consumedService [-serviceName MyServiceService -portName MyServicePort -namespace http://test -targetEndpoint http://localhost:9080/DemoJAXWSProject/MyEnhancedServiceService -portType MyServiceDelegate]]')
```

Example 27-3 Creating local service mapping with commands

```
wsadmin.sh -lang jython
```

```
wsadmin>AdminTask.createLMService('[-name DemoJAXWSProject -description "My Demo JAX-WS Project" -attachSMName NewServiceMap -consumedService [-serviceName MyServiceService -portName MyServicePort -namespace http://test -targetEndpoint
```

```

http://localhost:9080/DemoJAXWSProject/MyEnhancedServiceService -portType
MyServiceDelegate]]')
'DemoJAXWSProject'
wsadmin>
wsadmin>AdminConfig.save()
''

wsadmin>
wsadmin>AdminTask.showLMService('DemoJAXWSProject')
'{name=DemoJAXWSProject, description=My Demo JAX-WS Project,
consumedService=[{targetEndpoint=http://localhost:9080/DemoJAXWSProject/MyEnhanced
ServiceService, namespace=http://test, serviceName=MyServiceService,
portName=MyServicePort, portType=MyServiceDelegate}],
attachedServiceMap=NewServiceMap, serviceStatus=LocalMappingServiceState.STARTED}'

```

For more information about creating local mapping services, see the following topics in the WebSphere Application Server Information Center:

- ▶ http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-n-d-mp&topic=tsmwas_createlmserviceusingadminconsole
- ▶ http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-n-d-mp&topic=tsm_createlmservice

27.2.2 Starting and stopping a local mapping service

A local mapping service can be started and stopped. The default state for a local mapping service, when it is created, is the started state. The started state is the default, even if no service map is attached to that local mapping service. Table 27-1 shows the different states for local mapping services, and their behavior. The behavior is dependent on whether that local mapping service has a service map attached to it.

Table 27-1 Local mapping service behavior

| Local mapping service status | Attached to a service map? Yes/No | Behavior |
|------------------------------|---|---|
| Started | No | The request is routed to the original target service specified by the service client. |
| Stopped | No | The service is not available. An error is returned. |
| Started | Yes | The service map is applied. |
| Stopped | Yes | The service is not available. An error is returned. |
| Unknown | Yes, but the service map has not yet been saved to the master configuration | The request is routed to the original target service specified by the service client. |

27.3 Administration for target service clients

When a service map is installed on WebSphere Application Server, a business-level application (BLA) is automatically generated. See Figure 27-8 on page 953.

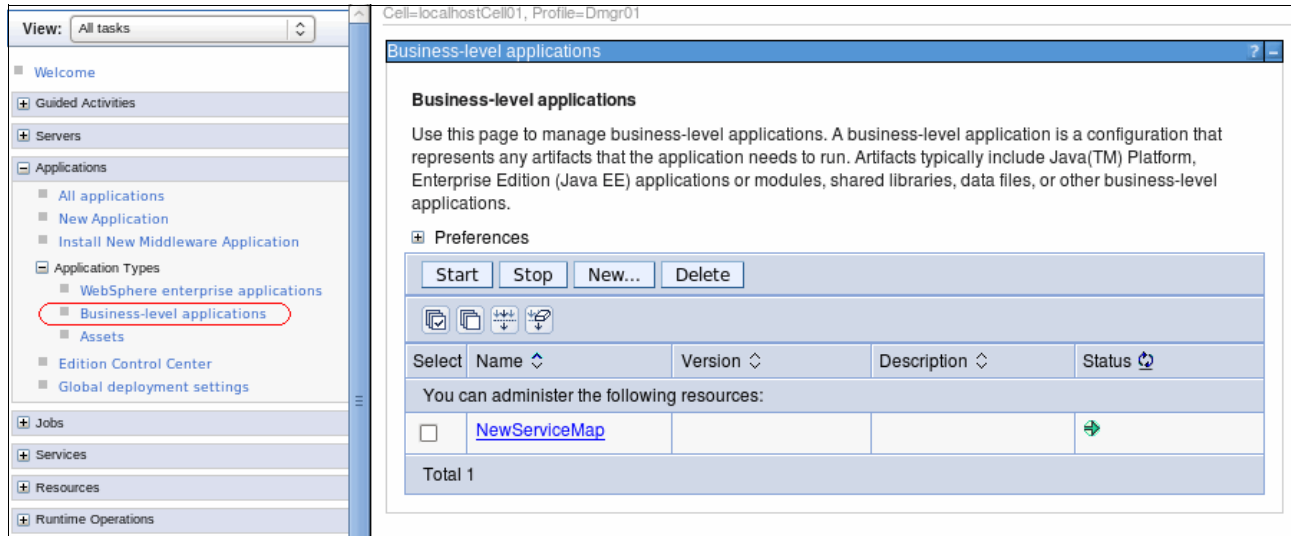


Figure 27-8 Service map BLA application

The BLA contains the following two composition units, which link to two assets that are automatically generated at the same time as the BLA:

- ▶ Enterprise application: The service map enterprise application is used to make outgoing calls to the target services that are defined in the service map. The name of the generated enterprise application is the name of the service map with the letters App appended.
- ▶ Enterprise bundle archive (EBA) asset: The service map EBA asset contains the files that make up the service map. It is used for target service selection and message transformation.

To see the BLA composition units, click the BLA application name in Figure 27-8. The BLA composition units will be displayed as shown in Figure 27-9 on page 954.

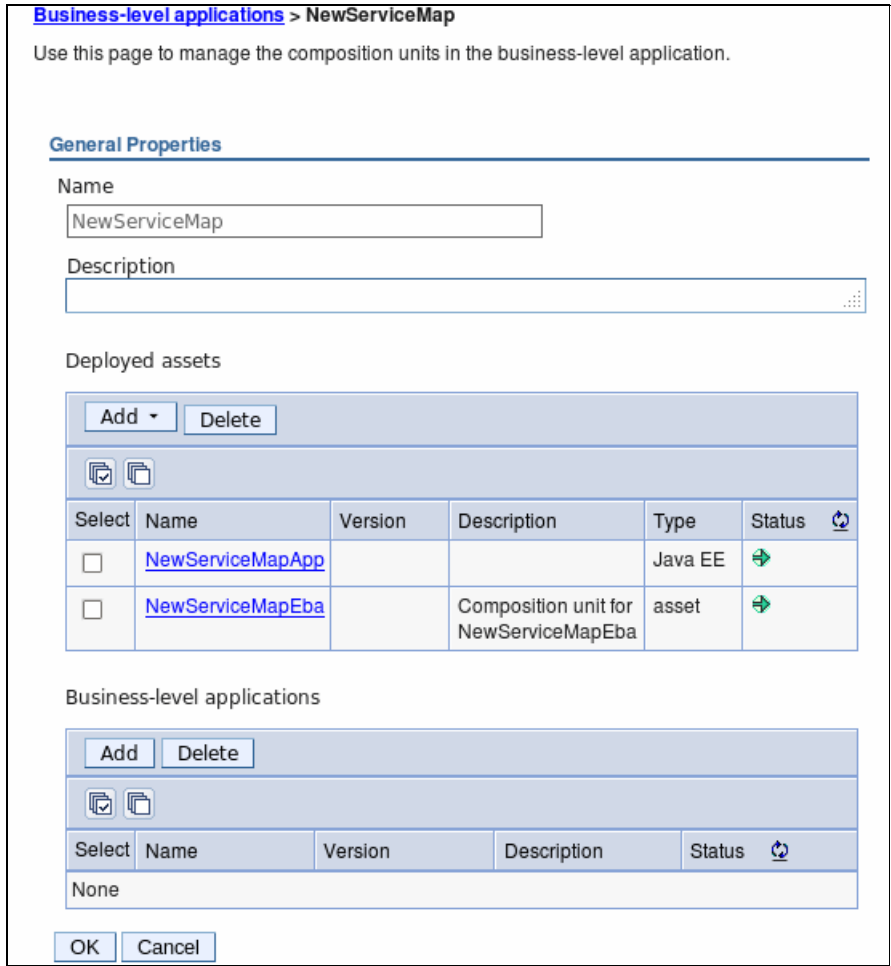


Figure 27-9 Service map BLA composition units

The enterprise application and EBA asset are standard WebSphere Application Server assets that can be managed by an administrator. One example of administration is applying policy sets to configure WS-Security for the request that is sent to the target service.

The assets can be deployed to specific WebSphere Application Server clusters and servers. To see the new enterprise applications, navigate to **Applications** → **Application Types** → **WebSphere enterprise applications**, as shown in Figure 27-10.

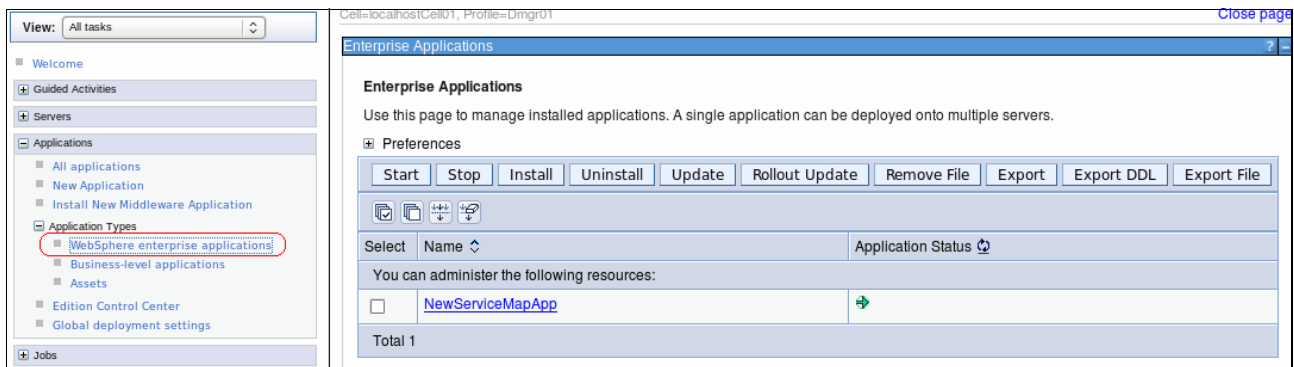


Figure 27-10 Service map - New application

27.3.1 Policy sets and bindings

You can specify the policy set and policy set binding that a service map uses by referencing the web services client provided in the generated enterprise application. To do this from the administrative console, click the name of the new application that was generated when you created a service map. In this scenario, click **NewServiceMapApp** in the list of enterprise applications and select **Service client policy sets and bindings** in the Web Services Properties. The resulting panel will display the Service client policy sets and bindings, as shown in the Figure 27-11.

Enterprise Applications > NewServiceMapApp > Service client policy sets and bindings

Define policy and binding configuration for the application, its service clients, endpoints, or operations. These settings are also used as default settings for the service references of the service clients. To customize the settings for service references, navigate to the list of Service clients under Services or on the Application configuration page. Access a Policies Applied link to indicate whether to use and how to acquire policies from the service provider. Complete the attachment by providing system-specific configuration when you assign the appropriate binding.

Preferences

Attach Client Policy Set | Detach Client Policy Set | Assign Binding

| Select | Application/Service/Endpoint/Operation | Attached Client Policy Set | Policies Applied | Binding |
|---|--|----------------------------|----------------------|----------------|
| You can administer the following resources: | | | | |
| <input type="checkbox"/> | NewServiceMapApp | None | None | Not applicable |
| <input type="checkbox"/> | MyEnhancedServiceService | None | None | Not applicable |
| <input type="checkbox"/> | MyEnhancedServicePort | None | None | Not applicable |
| <input type="checkbox"/> | getGreeting | None | None | Not applicable |
| Total 4 | | | | |

Figure 27-11 Service map policy sets and bindings

For more information about policy sets and bindings, see the following topic in the WebSphere Application Server Information Center:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=usca_psattach

27.3.2 Override target service URLs

You can override the target service endpoint URL used by the service map. The endpoint URL is acquired by the service map through referencing the web services client provided in the generated enterprise application. To override the URL, use the administrative console and follow these steps:

1. Open the configuration for the application. In this scenario, click **NewServiceMapApp**, as shown in Figure 27-8 on page 953.
2. In the next panel, click **Manage Modules** to display the Enterprise JavaBeans (EJB) module, as illustrated in Figure 27-12 on page 956.

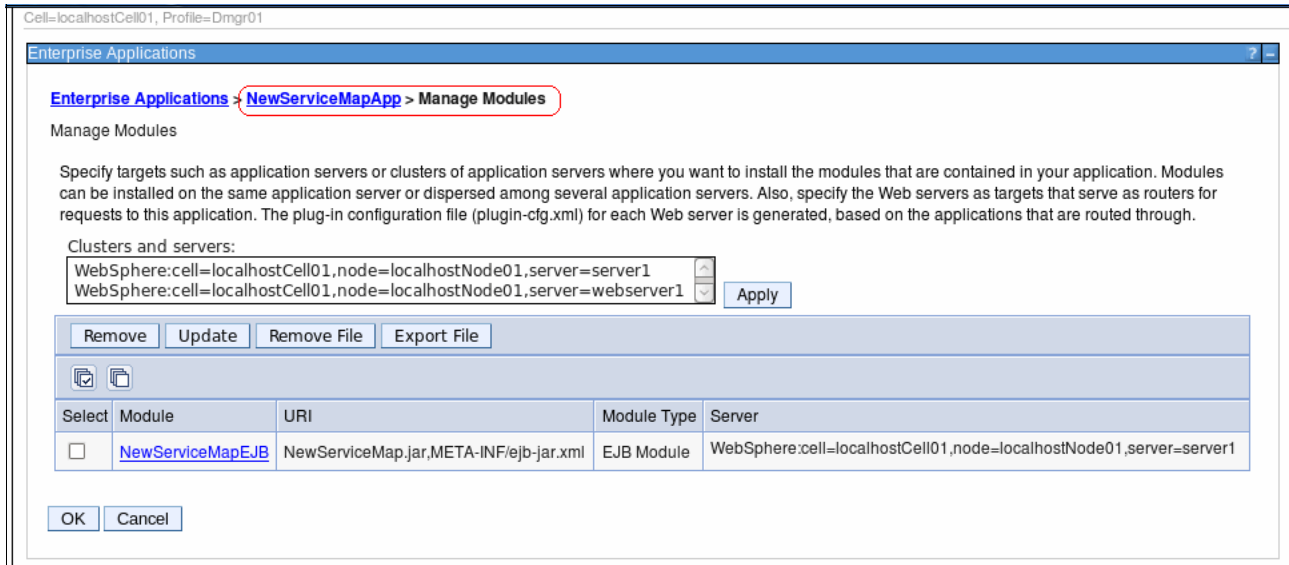


Figure 27-12 Service map - Manage modules

3. Click the module name. In this scenario, click **NewServiceMapEJB**, as shown in Figure 27-12.
4. In the next panel, click **Web services client bindings** to display the Web Services Description Language (WSDL) file name, preferred port mappings, and port information.
5. Click **Edit** in the Port Information column as shown in Figure 27-13.

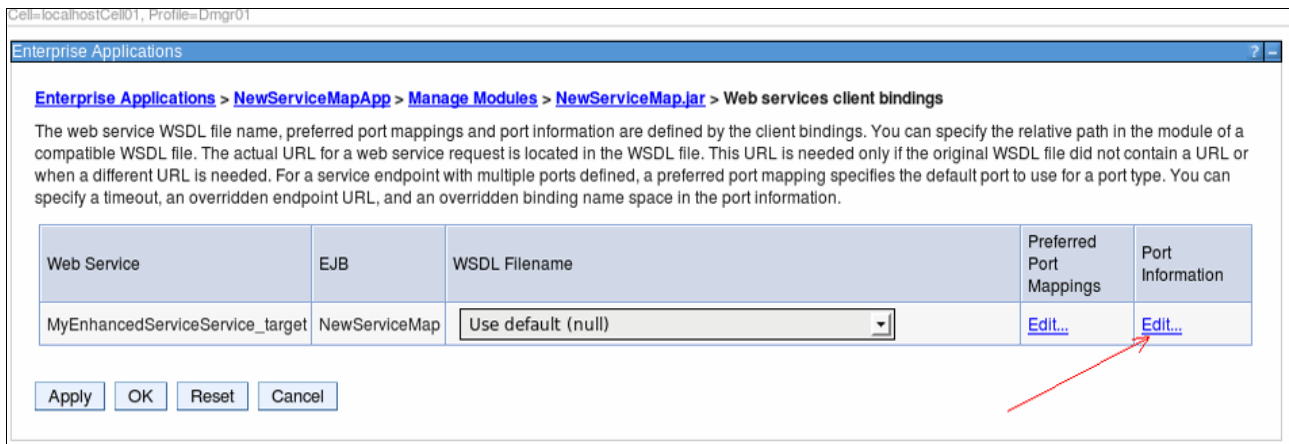


Figure 27-13 Service map Port Information

6. The Port Information panel appears and you can update the following options (Figure 27-14 on page 957):
 - Request Timeout
 - Endpoint URL
 - Binding Namespace

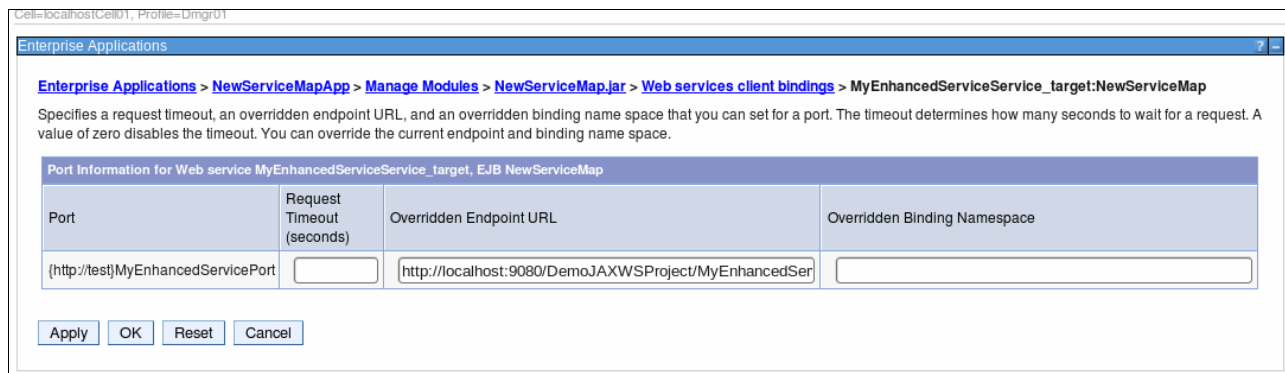


Figure 27-14 Service map - Override URL

27.4 Event emissions

Service mapping allows you to emit events when service requests and responses are intercepted by a local mapping service. Events are configured on local mapping services and published to a Java Message Service (JMS) topic on a specified JMS provider. Events can contain details about the request. Subscribers to the JMS topic use this event information to monitor the business status of their systems.

To enable local mapping services to emit events, you must configure an event point. The *event point* controls properties of the event, including the following attributes:

- ▶ When the events are emitted
- ▶ The event contents, including whether message data is included
- ▶ Topic to which the events are published to

Important: To include message data in events, you must create the WebSphere variable `SM_EVENT_POLICY_PERMIT_MESSAGE_DATA` and set its value to `true`.

Use the `createLMServiceEventPoint` command for the `AdminTask` object to create an event point on an existing local mapping service (Example 27-4).

Example 27-4 Create an event point using the `createLMServiceEventPoint` command

```
AdminTask.createLMServiceEventPoint('-lmService StockQuoteService
-connectionFactory myJmsCF')
```

```
AdminTask.enableLMServiceEventPoint('-lmService StockQuoteService')
```

```
AdminConfig.save()
```

```
AdminTask.showLMService('StockQuoteService')
```

```
{name=StockQuoteService, description=Stock Quote JAX-WS service,
consumedService=[{portType=sqType, namespace=http://test, serviceName=sqservice,
targetEndpoint=http://localhost:9080/StockQuote/sqservice, portName=sqport}],
serviceStatus=LocalMappingServiceState.STARTED, eventPoint={enabled=true,
mustSend=false, sendAtCommit=false, eventData=NONE, connectionFactory=myJmsCF}}
```

Important: The event point is disabled by default. Use the `enableLMServiceEventPoint` command to enable it; see Example 27-4 on page 957.

The emitted events are in an XML format. If the event point has been configured to include request data within the event, that data is based on the state of the SOAP request when it is intercepted. The event can contain unencrypted request data because the local mapping service intercepts service requests before any WS-Security or transport-level encryption.

27.4.1 Schema explanation

The `eventPointData` element contains information about how and when the event was generated, including the following attributes:

- ▶ `eventSourceAddress`: Contains the originating application, service, local mapping service, and operation name of the service request, response, or fault that was intercepted.
- ▶ `eventName`, in the `eventIdentity` element, is set to the following values:
 - `CLIENT_PREMAP_REQUEST` when a service request is intercepted by the local mapping
 - `CLIENT_POSTMAP_RESPONSE` when a service response or fault is returned to the originating service client
 - `CLIENT_POSTMAP_RESPONSE_FAULT` when a fault is returned to the originating service client
- ▶ `creationTime`, in the `eventSequence` element: Contains the time when the web service request, response, or fault was intercepted.
- ▶ Transaction ID in the `eventCorrelation` element has the following attributes:
 - The `localTransactionId` attribute: A unique ID generated for the web service request, response, or fault that was intercepted.
 - The `parentTransactionId` attribute: A unique ID that can be used to correlate events emitted for the web service request and the returned web service response or fault.
 - The `globalTransactionId` attribute: The WebSphere Application Server global transaction ID.
 - If the event point has been configured to include message data in the event, the `bitstream` element contains the message data in Base-64 encoded form.

Example 27-5 shows a published event.

Example 27-5 Published event

```
<ns1:event
xmlns:ns1="http://www.ibm.com/xmlns/prod/websphere/servicemapping/event/2013/06">
<ns1:eventPointData>
<ns1:eventData ns1:productVersion="" ns1:eventSchemaVersion=""
ns1:eventSourceAddress="DemoJAXWS/DemoLM/getGreeting">
<ns1:eventIdentity ns1:eventName="CLIENT_POSTMAP_RESPONSE_FAULT" ns1:severity=""
ns1:priority="" ns1:successDisposition=""/>
<ns1:eventSequence ns1:creationTime="2013-04-09T11:04:05.204+01:00"
ns1:counter="0"/>
<ns1:eventCorrelation
ns1:localTransactionId="50c69d27-fb5c-48d6-881b-8430c2520f7c"
ns1:parentTransactionId="3cb52402-b953-424a-a6f2-28b39eef5877"
ns1:globalTransactionId=""/>
</ns1:eventData>
```

```

</ns1:eventPointData>
<ns1:bitstreamData>
<ns1:bitstream ns1:encoding="base64Binary"/>
</ns1:bitstreamData>
</ns1:event>

```

For more information about the service mapping event schema, see the following topic in the WebSphere Application Server Information Center:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=rsmwas_servicemappingeventschema

27.5 Securing a service map

A new role, named `ServiceMapUser`, is available to secure a service map. It allows the users or groups that access target services to also access the service map.

To secure a service map, from the administrative console, navigate to **Applications** → **WebSphere Enterprise Applications**.

Select the service map enterprise application, in this example, **NewServiceMapApp**. Then, click **Security role to user/group mapping** to display the Security role to user/group mapping panel as shown in Figure 27-15. You can then map users, groups, or special subjects to the `ServiceMapUser` role to allow web service client applications to call the service map enterprise application.

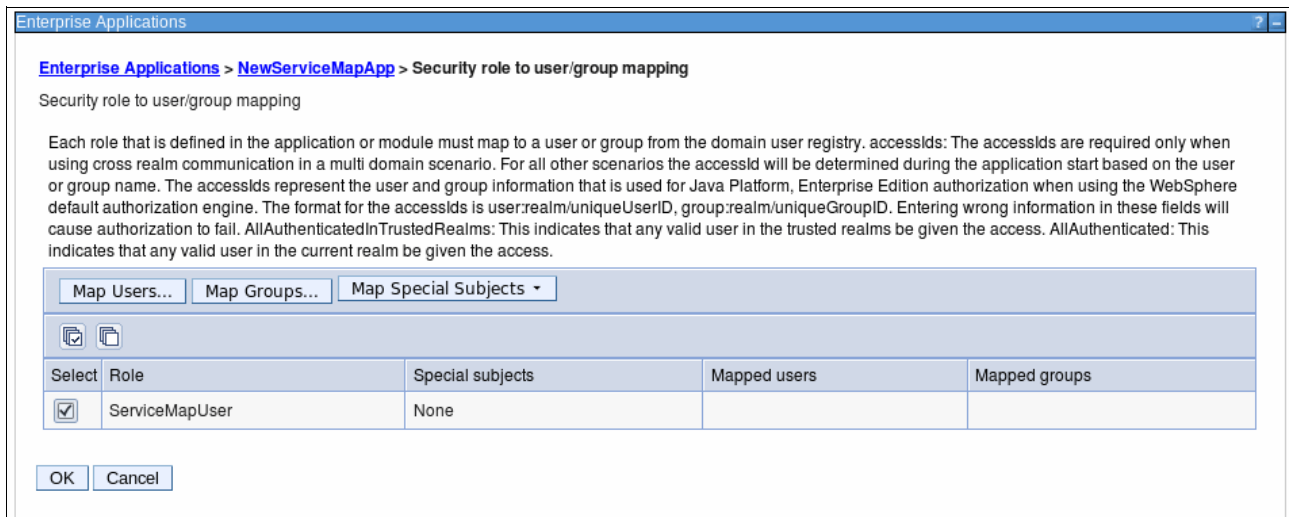


Figure 27-15 Service map security

For more information about role-based authorization, see the following topic in the WebSphere Application Server Information Center:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=cjr0450_



Session management

Session support allows a web application to maintain state information across multiple user requests. In this chapter, we discuss how you can manage the sessions with WebSphere Application Server.

This chapter includes the following topics:

- ▶ Session overview
- ▶ Session management configuration
- ▶ Storing session information
- ▶ Session affinity
- ▶ Session management tuning
- ▶ Stateful session bean failover

28.1 Session overview

In many web applications, user choices or actions determine where the user is sent next, how the application behaves, or what the page displays. For example, if the user clicks a checkout button on a site, the next page must contain the user's shopping choices and information. The Java servlet specification provides a mechanism for servlet applications to maintain a user's state information. This mechanism, is known as a *session*. Sessions allow applications, running in a web container, to keep track of individual users.

A servlet distinguishes users by their unique session IDs. The session ID is stored as a cookie or alternatively can be conveyed to the servlet by URL rewriting.

28.1.1 Session identifiers

WebSphere Application Server passes the user an identifier known as a *session ID*, which correlates an incoming user request to a session object that is maintained on the server. The session ID arrives with each request.

Note: In accordance with the Servlet 2.3 API specification, the session management facility supports session scoping by web modules. Only servlets in the same web module can access the data associated with a particular session. Multiple requests from the same browser, each specifying a unique web application, result in multiple sessions with a shared session ID. You can invalidate any of the sessions that share a session ID without affecting the other sessions.

There are three approaches used in WebSphere Application Server for tracking sessions:

- ▶ Cookies
- ▶ URL rewriting
- ▶ SSL session identifiers (deprecated)

Deprecated feature: Session tracking using the SSL ID is deprecated in WebSphere Application Server 7.0. You can configure session tracking to use cookies or URL rewriting.

Cookies

WebSphere Application Server session support generates a unique session ID for each user and returns this ID to the user's browser with a cookie, as illustrated in Figure 28-1. The default name for the session management cookie is *JSESSIONID*.

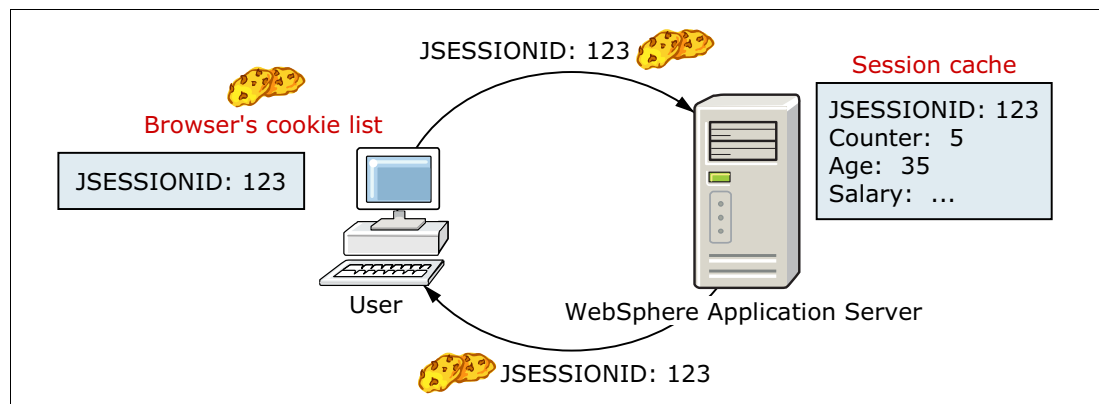


Figure 28-1 Cookies overview

A cookie consists of information that is embedded as part of the headers in the HTML stream passed between the server and the browser. The browser holds the cookie and returns it to the server whenever the user makes a subsequent request. By default, WebSphere defines its cookies so they are destroyed if the browser is closed.

The web application developer uses the HTTP request object's standard interface to obtain the session, as shown in Example 28-1.

Example 28-1 Get the HTTP session object

```
//Suppose HttpServletRequest request has been initiated.  
HttpSession session = request.getSession(true);  
String sessionID = session.getId();
```

WebSphere places the user's session identifier in the outbound cookie when the servlet completes its execution, and the HTML response stream returns to the user.

URL rewriting

A typical usage of URL rewriting is configuring session tracking for Wireless Application Protocol (WAP) devices. Because most WAP devices do not support cookies, you can configure these devices to use URL rewriting to track sessions. URL rewriting requires the developer to perform the following actions:

- ▶ Use special APIs to encode the URLs.
- ▶ Set up the site page flow to avoid losing the encoded information.

Program session servlets to encode URLs

URL rewriting works by storing the session identifier in the page that is returned to the user. WebSphere Application Server encodes the session identifier as a parameter on URLs that are encoded programmatically by the web application developer. Example 28-2 shows a web page link with URL encoding.

Example 28-2 Web page link with URL encoding

```
<a href="/store/catalog;$jsessionid=DA32242SSGE2">
```

When the user clicks this link to move to the `/store/catalog` page, the session identifier is passed in the request as a parameter.

If the servlet returns HTML directly to the requester, without using *JavaServer Pages (JSP)*, the servlet calls the API, as shown in Example 28-3, to encode the returning content.

Example 28-3 URL encoding from a servlet

```
//Suppose HttpServletResponse response has been initiated.  
out.println("<a href=\"");  
out.println(response.encodeURL ("/store/catalog"));  
out.println("\>catalog</a>");
```

The pages using redirection, servlet, or JSP must encode the session ID as part of the redirection, as shown in Example 28-4.

Example 28-4 URL encoding with redirection

```
//Suppose HttpServletResponse response has been initiated.  
response.sendRedirect(response.encodeRedirectURL ("http://myhost/store/catalog"));
```

Supplying a servlet or JSP file as an entry point

The entry point to an application, such as the initial window presented, might not require the use of sessions. However, if the application requires session support, after a session is created, all URLs are encoded to perpetuate the session ID for the servlet requiring the session support. The following syntax shows how you can embed Java code within a JSP file. JSP calls a similar interface to encode the session ID:

```
<% response.encodeURL ("/store/catalog"); %>
```

Note: WebSphere Application Server inserts the session ID into dynamic pages but cannot insert the user's session ID into static pages, .htm, or .html.

28.1.2 Session invalidation

When the user no longer needs the session object (user logs off), the sessions belonging to that user can be invalidated. The invalidating process removes a session from the session cache and from the persistent store.

WebSphere Application Server offers the following methods for invalidating session objects:

- ▶ **On Demand:** You can explicitly use the `invalidate()` command in application code to immediately invalidate the session object. If the session object is accessed by multiple threads in a web application, be sure that none of the threads still have references to the session object.
- ▶ **Automated or Periodic:** The session manager treats a session as a candidate for invalidation if it has not been accessed for a period that is longer than the specified session timeout. The session manager has an invalidation process thread that runs periodically to invalidate sessions that are eligible for invalidation. For more information, refer to the website:

<http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=cprsinvalidation>

Note: In a distributed environment, one cluster member can be randomly chosen to act as the invalidator for the entire cluster. The cluster member can invalidate the session regardless of the session in which that cluster member was created.

- ▶ **Scheduled invalidation:** In a distributed environment, you can set specific times for the session management facility to scan for invalidated sessions instead of relying on the periodic invalidation timer. For more information, refer to the website:
- http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=cprs_skin

28.1.3 Session listeners

Session listener classes are defined to listen for state changes of a session and its attributes. This *listening* allows control over interactions with sessions, so that programmers can monitor creation, deletion, and modification of sessions. With the help of listeners, programmers can perform initialization tasks when a session is created or can clean up tasks when a session is removed. It is also possible to perform some specific tasks for a session attribute when the attribute is added, deleted, or modified.

The following Table 28-1 on page 965 shows the listener interfaces and how they are used to monitor the events that are associated with the HttpSession object.

Table 28-1 Listener interfaces and their methods

| Target | Event | Interface | Method | Comments |
|-----------|-----------|-------------------------------|------------------------|---|
| session | create | HttpSessionListener | sessionCreated() | To monitor creation and deletion, including session timeout. |
| | destroy | HttpSessionListener | sessionDestroyed() | |
| session | activate | HttpSessionActivationListener | sessionDidActivate() | To monitor changes of session attributes, such as add, delete, and replace. |
| | passivate | HttpSessionActivationListener | sessionWillPassivate() | |
| attribute | add | HttpSessionAttributeListener | attributeAdded() | To monitor sessions that are made active or that are made passive. |
| | remove | HttpSessionAttributeListener | attributeRemoved() | |
| | replace | HttpSessionAttributeListener | attributeReplaced() | |

For more information, see the Java EE specifications at the following website:

<http://www.oracle.com/technetwork/java/javaee/overview/index.html>

28.1.4 Session security

You can integrate sessions and security in WebSphere Application Server. When session security (security integration) is enabled, the session manager checks the user ID of the HTTP request against the user ID of the session held within WebSphere Application Server. This check is done as part of the processing of the request.getSession() function. If the check succeeds, the session data is returned to the calling servlet or JSP. If the check fails, WebSphere throws the `com.ibm.websphere.servlet.session.UnauthorizedSessionRequestException`.

The session security integration is enabled by default. To view or edit the security integration setting, refer to section 28.2, “Session management configuration” on page 966.

The identity or user name of a session can be accessed through the `com.ibm.websphere.servlet.session.IBMSession` interface. An unauthenticated identity is denoted by the user name *anonymous*.

Session management security uses the following rules:

- ▶ Sessions in unsecured pages are treated as accesses by the anonymous user.
- ▶ Sessions created in unsecured pages are created under the identity of that anonymous user.
- ▶ Sessions in secured pages are treated as accesses by the authenticated user.
- ▶ Sessions created in secured pages are created under the identity of the authenticated user. They can only be accessed in other secured pages by the same user. To protect these sessions from use by unauthorized users, they cannot be accessed from an unsecure page. Do not mix access to secure and unsecure pages.

For more information about the session security, refer to the following website:

http://fred.rtp.raleigh.ibm.com:8680/help/index.jsp?topic=/com.ibm.websphere.nd.doc/ae/rprs_secg.html

28.2 Session management configuration

There are three levels of session management configuration:

- ▶ Web container (the default level): Configuration at this level is applied to all web modules within the server.
- ▶ Application: Configuration at this level is applied to all web modules within the application.
- ▶ Web module: Configuration at this level is applied only to that specific web module.

When you configure session management at the web container level, all applications and the respective web modules in the web container normally inherit that configuration. However, you can set up different configurations individually for specific applications and web modules that vary from the web container default.

28.2.1 Session management properties

With the exception of the *Overwrite session management* parameter, the session management properties are the same at each configuration level. The following list describes the parameters available:

- ▶ *Overwrite session management* determines whether these session management settings are used for the current module or inherited from the parent object. Only the application level or the web module level have such parameters.
- ▶ *Session tracking mechanism* lets you select from cookies, URL rewriting, and SSL ID tracking. Selecting cookies leads you to a second configuration page that contains further configuration options.
- ▶ *Maximum in-memory session count* specifies the maximum number of sessions to keep in memory. Default value is 1000 sessions:
 - For local sessions, this value specifies the number of sessions in the base session table.
 - For persistent sessions, this value specifies how many sessions are cached before manual updates or before the session manager reverts to reading a session from the persistent storage automatically.
- ▶ *Allow overflow* specifies whether to allow the number of sessions in memory to exceed the value specified in the maximum in-memory session count field.

For local sessions, use the Allow overflow option to manage session storage. Sessions can either be limited to store in the primary cache table of the session manager, or optioned to allow additional sessions to be stored in secondary extended tables.

Important: Allowing an unlimited amount of sessions can potentially exhaust system memory and even allow for system sabotage. For best performance, define a primary cache of sufficient size to hold the normal working set of sessions for a given application server.

- ▶ *Session timeout* specifies the amount of time to allow a session to remain idle before invalidation. The default value is 30 minutes. This setting is important for performance

tuning. It directly influences the amount of memory that is consumed by the JVM to cache the session information. Session timeout also impacts the session manager invalidation process time intervals. For the default timeout value, the invalidation process interval is around 300 seconds. Using default settings, it can take up to five minutes beyond the timeout threshold of thirty minutes for a particular session to become invalidated.

If you select the **No timeout** option, a session can never be removed from the memory unless explicit invalidation is performed. This persistent session can cause a memory leak when the user closes the window without logging out from the system. To use this option, make sure that enough memory or space in a persistent store is kept to accommodate all sessions.

- ▶ *Security integration* specifies that the user ID be associated with the HTTP session.

Note: Do not enable this property if the application server contains a web application that has *form-based login* configured as the authentication method and the local operating system is the authentication mechanism. Doing so causes authorization failures when users try to use the web application.

- ▶ *Serialize session access* determines if concurrent session access in a given JVM is allowed. Serialized access ensures thread-safe access when the session is accessed by multiple threads. No special code is necessary for using this option. This option is not recommended when user requests are issued frequently because it can affect performance.

You can set an optional property, the *Maximum wait time*, to specify the maximum amount of time that a servlet request waits on an HTTP session before continuing execution. The default value for this setting is five seconds.

- ▶ *Distributed environment settings* determines how to persist sessions (memory-to-memory replication or a database) and set tuning properties. For session recovery support, WebSphere Application Server provides distributed session support persist sessions replication. You can use session recovery support when the user's session data must be maintained across a server restart or when the user's session data is too valuable to lose through an unexpected server failure. Memory-to-memory persistence is available only in a Network Deployment distributed server environment.
- ▶ *Custom properties* specifies additional settings for session management. For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=rprs_custom_properties

28.2.2 Accessing session management properties

You can access all session management configuration settings using the administrative console, as shown in Figure 28-2 on page 968.

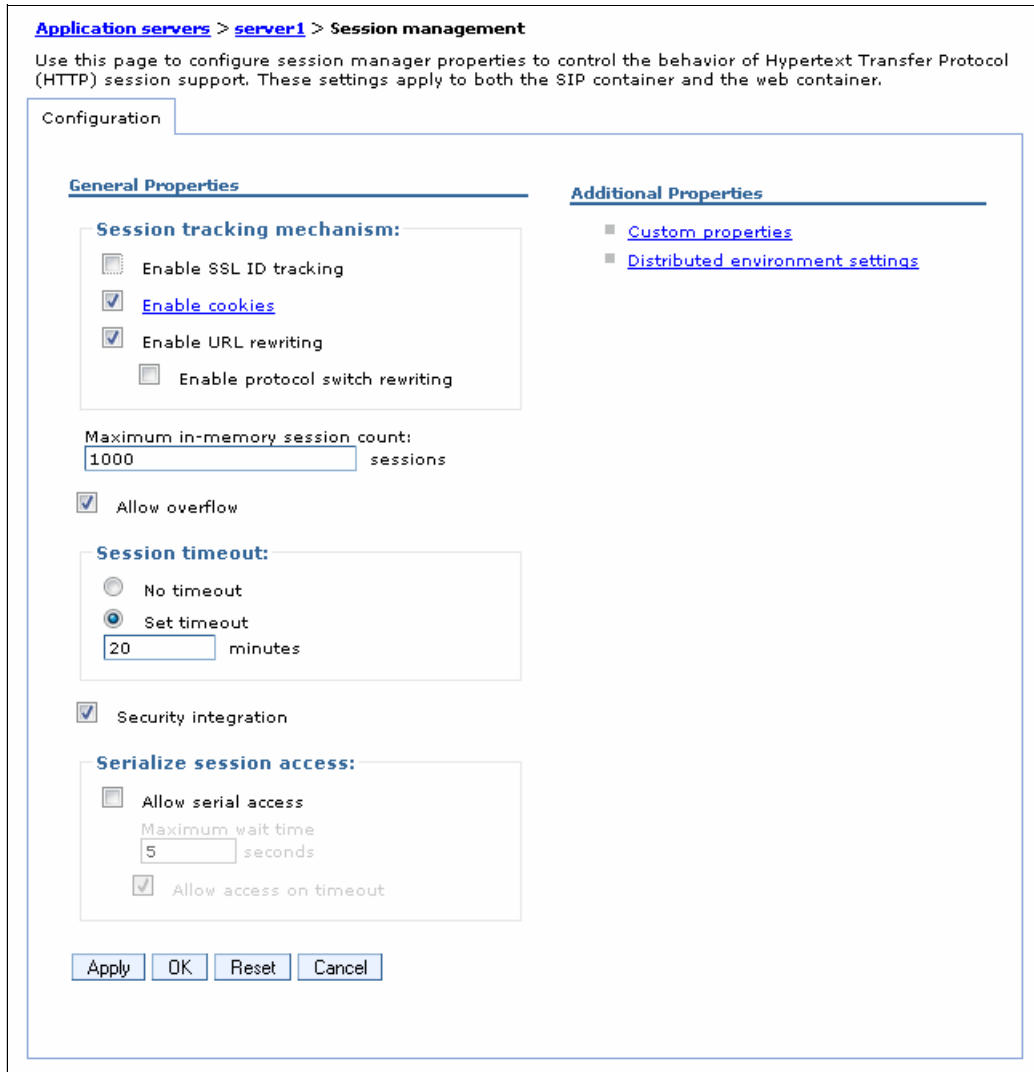


Figure 28-2 Session management configuration setting

You can change the configuration for the web container, enterprise application, or web module level. To make these changes use the following guidelines:

- ▶ Application server web container level:
 - a. Click **Servers** → **Server Types** → **WebSphere application servers** → *server_name*.
 - b. Under the Container Settings of the Configuration tab, click **Session management**.
Or Under Container Settings, expand **Web Container Settings**, and click **Web container**. Under **Additional Properties**, click **Session management**.
- ▶ Application level:
 - An enterprise application:
 - a. Click **Applications** → **Application Types** → **WebSphere enterprise applications** → *application_name*.
 - b. Under **Web Module Properties** section of the Configuration tab, click **Session management**.

- An OSGi application:
 - a. Click **Applications** → **Application Types** → **Business-level applications** → *application_name* → *eba_asset_name*.
 - b. Under Additional Properties, click **Session management**.
- ▶ Web module level:

If the application is an enterprise application, complete the following steps:

 - a. Click **Applications** → **Application Types** → **WebSphere enterprise applications** → *application_name*.
 - b. Under the Modules section of the Configuration tab, click **Manage Modules** → *module_name*.
 - c. Under **Additional Properties**, click **Session Management**.

Note: If you are working at the web module or application level and want these settings to override the inherited session management settings, under General Properties select **Override session management**.

28.2.3 Selecting session tracking options

To set or change the session mechanism type:

1. Go to the appropriate level of Session Management described in 28.2.2, “Accessing session management properties” on page 967.
2. Under General Properties, select the session tracking mechanism:
 - To track sessions with cookies, select **Enable Cookies**.
To change the cookie settings, click hot link of **Enable Cookies**.
 - To track sessions with URL rewriting, select **Enable URL Rewriting**.
If you want to enable protocol switch rewriting, select **Enable protocol switch rewriting**. This option defines whether the session ID, added to a URL as part of URL encoding, is included in the new URL if a switch from HTTP to HTTPS or from HTTPS to HTTP is required.
 - To track sessions with SSL information, select **Enable SSL ID tracking**.
3. Click **OK** and **Save**, and synchronize the configuration changes.
4. Restart the application server or the cluster.

28.2.4 Scheduled invalidation configuration

To complete the schedule sessions cleanup setting, as described in 28.1.2, “Session invalidation” on page 964, complete the following steps:

1. Go to the appropriate level of Session Management described in 28.2.2, “Accessing session management properties” on page 967.
2. Under Additional Properties, click **Distributed environment settings**.
3. Under Additional Properties, click **Custom tuning parameters**.
4. Under General Properties, click **Custom settings**.
5. Under Schedule sessions cleanup of General Properties, as shown in Figure 28-3 on page 970, specify your setting value.

[Application servers](#) > [was85Cluster01Member01](#) > [Session management](#) > [Distributed environment settings](#) > [Tuning parameters](#) > [Custom tuning parameters](#)

Use this page to specify tuning parameters for session management.

Configuration

General Properties

Write frequency

End of servlet service
 Manual update
 Time based:
 seconds

Write contents

Only updated attributes
 All session attributes

Schedule sessions cleanup:

Specifies distributed sessions cleanup schedule
 First time of day (0-23):
 Second time of day (0-23):

Figure 28-3 Schedule session cleanups

28.2.5 Cookie setting

When you select cookie as the session mechanism type, you can view or change the cookies settings by clicking the **Enable Cookies** hot link, as described in 28.2.3, “Selecting session tracking options” on page 969. Figure 28-4 on page 971 shows the available cookie settings.

[Application servers](#) > [server1](#) > [Session management](#) > [Cookies](#)

Use this page to specify cookie settings for Hypertext Transfer Protocol (HTTP) session management.

Configuration

General Properties

Cookie name

Restrict cookies to HTTPS sessions

Set session cookies to HTTPOnly to help prevent cross-site scripting attacks

Cookie domain

Cookie maximum age

Current browser session

Set maximum age
 seconds

Cookie path

Use the context root

Set cookie path

Figure 28-4 Cookie configuration page

The available cookies settings, shown in Figure 28-4, are described in the following list:

► **Cookie name**

Specifies a unique cookie name for session management. The default value is JSESSIONID.

► **Restrict cookies to HTTPS sessions**

Specifies that the session cookies include the secure field. Enabling this feature restricts the exchange of cookies to HTTPS sessions only and the session cookie's body includes the secure indicator field.

► **Set cookies as HTTP only to help prevent cross-site scripting attacks**

Specifies that session cookies include the HTTP only field. When checked, browsers that support the HTTP only attribute do not enable cookies to be accessed by client-side scripts. For security cookies, see the global security settings for web single sign-on (SSO).

► **Cookie domain**

Dictates to the browser whether to send a cookie to particular servers. For example, if you specify a particular domain, the browser sends back session cookies only to hosts in that domain. The default value is the server.

Note: The Lightweight Third Party Authentication (LTPA) token or cookie that is sent back to the browser is scoped by a single DNS domain that is specified when security is configured. Thus, all application servers in an entire WebSphere Application Server domain must share the same DNS domain for security purposes.

- ▶ **Cookie maximum age**

Specifies the amount of time that the cookie lives in the client browser. This option includes the following choices:

- Expire at the end of the *Current browser session* which is the default option.
- Expire by configuring *Set maximum age*

If you choose the maximum age option, specify the age in seconds. This value corresponds to the *Time to Live (TTL)* value described in the Cookie specification.

- ▶ **Cookie path**

Sets the paths on the server that define where the browser sends the session tracking cookie. Specify any string that represents a path on the server:

- Use the context root
- Set cookie path, which is also the default option (use the forward slash (/) to indicate the root directory).

Specifying a value restricts the paths to which the cookie is sent. By restricting paths, you can keep the cookie from being sent to certain URLs on the server. If you specify the root directory, the cookie is sent no matter which path on the given server is accessed.

28.3 Storing session information

By default, WebSphere places session objects in memory as local session cache. However, the administrator can enable persistent session management to place session objects in a persistent store.

Administrators must enable persistent session management in the following situations:

- ▶ In a distributed environment, when the user's session data must be recovered by another cluster member after a cluster member in a cluster fails or is shut down.
- ▶ The user's session data is too valuable to lose through unexpected failure at the application server.
- ▶ The administrator desires better control of the session cache memory footprint by sending cache overflow to a persistent session store.

28.3.1 Local sessions

Many web applications use the simplest form of session management, which is the in-memory, local session cache. The local session cache keeps session information in memory and local to the WebSphere Application Server where the session information was first created.

Local session management does not share user session information with other clustered servers. The local session management lacks a persistent store for the sessions it manages. A server failure eliminates the WebSphere Application Server instances and also destroys any sessions that are managed by those instances.

The administrator can define a limit on the number of sessions that are held in the in-memory cache by specifying the *Maximum in-memory session count* setting, as shown in 28.2.1, "Session management properties" on page 966. The session manager also permits an unlimited number of sessions in memory by enabling the *Allow overflow* setting. If you choose to enable session overflow, monitor the state of the session cache closely for performance purpose.

28.3.2 Persistent sessions management

WebSphere Application Server provides the following options for persistent session management:

- ▶ Database session persistence, where sessions are stored in the database specified.
- ▶ Memory-to-memory session replication using the data replication service available in distributed server environments.

In a distributed environment, you can have both these two session mechanism options, as shown in Figure 28-5. In a stand-alone environment, you can only set the database session persistence.

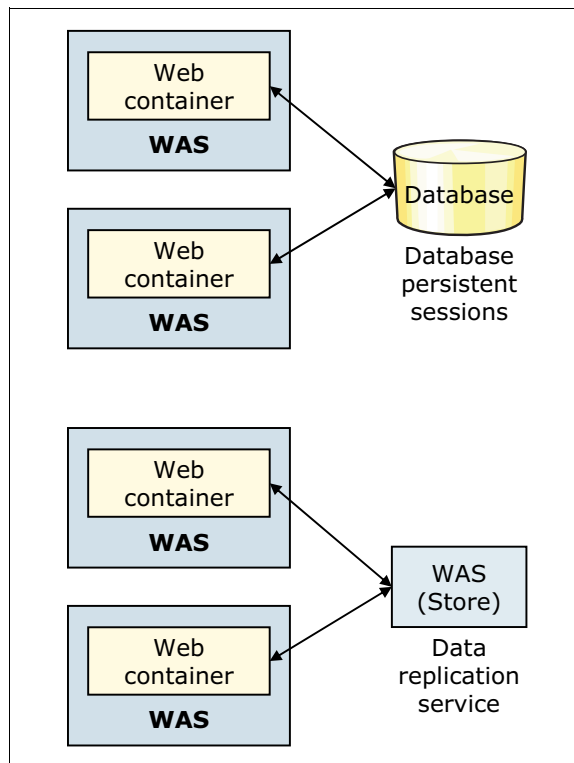


Figure 28-5 Persistent session options

All information that is stored in a persistent session store must be serialized. As a result, all of the objects that are held by a persistent session store must implement `java.io.Serializable`. In general, consider making all objects that are held by a session serialized, even if immediate plans do not call for the use of persistent session management. Enabling this feature makes the transition between local and persistent management occur transparently.

The session manager maintains a cache of the most recently used sessions in memory. If it cannot find the session information from the cache, session manager queries the persistent storage. Retrieving a user session from the cache eliminates a more expensive retrieval from the persistent store. Session data is stored to the persistent store based on your selections for write frequency and write content option.

Note: WebSphere eXtreme Scale dynamic cache provider can be used as a new session replication approach. This shared in-memory cache sits in a highly-available replicated grid. User sessions can be shared between any set of application servers, even across data centers, allowing a more reliable and fault-tolerant user session state. No application code change is required when using WebSphere eXtreme Scale to store and manage session data. For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=welc6tech_dyn_intro

Configuring the persistent session setting

To specify the persistent session:

1. Go to the appropriate level of Session Management described in 28.2.2, “Accessing session management properties” on page 967.
2. Under Additional Properties, click **Distributed environment settings**.
3. Under General Properties, as shown in Figure 28-6, select which session storage mechanism you want to use.

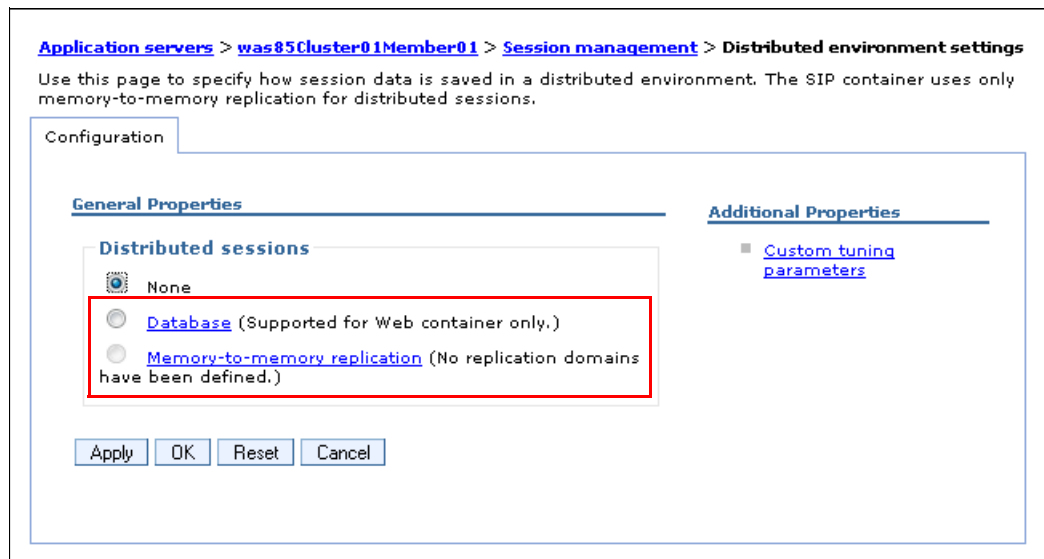


Figure 28-6 Distributed sessions setting

4. Click the **Database** or **Memory-to-memory replication** hot link and then specify the details of each persistent session configuration.

28.3.3 Enabling database persistence

In this section, we discuss enabling database persistence.

Database preparation

Before enabling database persistence, complete the following steps:

1. Create a session database.
2. Create a table for the session data:
 - In distributed environments, the session table is created automatically when you define the data source for the session management database. However, if you want to use a page (row) size greater than 4 KB, you need to create the table space manually as described at the website:
<http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=tperdb2t>
 - If you want to expand the column size limits, you can create the table externally, as described at the following website:
http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=tprs_table_creation
 - (z/OS DB2) Create a table for the session data, as described at the following website:
http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=tprs_db2tzos
3. Create a JDBC provider.
4. Create a data source point for the database.

Click **Resources** → **JDBC** → **JDBC Providers** → **JDBC_provider** → **Data Sources** → **New**.

The data source must be non-XA enabled and must be a non-JTA enabled data source.

The JNDI name of the database for persistence, for example, can be `jdbc/Sessions`.

Enabling the database persistence

To enable database persistence:

1. Go to the appropriate level of Session Management described in 28.2.2, “Accessing session management properties” on page 967.
2. Under Additional Properties, click **Distributed environment settings**.
3. Select the **Configuration** tab, previously shown in Figure 28-6 on page 974. Click the **Database** hot link.
4. Specify the database information, as shown in Figure 28-7 on page 976.
 - The Data Source JNDI name from the preparation step.
 - The database user ID and password that is used to access the database and for table creation.
 - Configure a table space and page sizes if you create them manually.
 - Switch to a multi-row schema.

Using multi-row sessions becomes important if the size of the session object exceeds the size for a row. If the multi-row session enabled, the session manager breaks the session data across multiple rows as needed. This method allows WebSphere Application Server to support large session objects. It also provides a more efficient

mechanism for storing and retrieving session contents under certain circumstances. See 28.5, “Session management tuning” on page 986 for more information.

The screenshot shows a web browser window with the following navigation path: **Application servers** > **was85Cluster01Member01** > **Session management** > **Distributed environment settings** > **Database settings**. Below the navigation path, there is a heading "Configuration" and a sub-heading "General Properties". The form contains the following fields and controls:

- Datasource JNDI name:** A text input field containing "jdbc/Sessions".
- User ID:** A text input field containing "db2admin".
- Password:** A text input field with masked characters "*****".
- DB2 row size:** A dropdown menu with "ROW_SIZE_4KB" selected.
- Table space name:** An empty text input field.
- Use multi row schema**
- Buttons: **Apply**, **OK**, **Reset**, and **Cancel**.

Figure 28-7 Database setting for session persistence

5. Optional: If you want to change the default tuning parameters, click **Custom tuning parameters**.
6. Click **OK** and Save the configuration changes. In cluster environment, repeat these steps for each server in the cluster. Save and synchronize the changes.
7. Restart the application servers or cluster.

Database session persistence can also be configured using scripting, for more information about this process, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-ba-se-dist&topic=txml_dbsessionpersist

28.3.4 Memory-to-memory replication

Memory-to-memory replication uses the data replication service to replicate data across many application servers in a cluster without using a database. Separate threads handle replication within an existing application server process. In this mode, sessions can replicate to address HTTP Session *single point of failure (SPOF)* and eliminates the effort maintaining a replication database. Session information between application servers is encrypted.

The data replication service is an internal WebSphere Application Server component. In addition to its use by the session manager, it is also used to replicate dynamic cache data and stateful session beans across many application servers in a cluster.

Note: Memory-to-memory replication requires the high availability (HA) manager to be active. For more information about the HA manager, refer to chapter 15.3, “High availability and failover” on page 538.

Data replication service modes

The memory-to-memory replication function is accomplished by creating a data replication service instance in an application server that communicates to other data replication service instances in remote application servers.

You can set up a replication service instance to run in any of the following modes:

- ▶ **Server mode:** The server is used to receive backup copies of other application server sessions. It does not send copies of sessions that are created in that particular server.
- ▶ **Client mode:** The server broadcasts or sends copies of the sessions it owns. It does not receive backup copies of sessions from other servers.
- ▶ **Both mode:** The server simultaneously sends copies of the sessions it owns and acts as a backup table for sessions that are owned by other application servers. This mode is the default setting.

Replication configuration type

The following list notes the officially supported configuration types. However, WebSphere Application Server allows additional possibilities for memory-to-memory replication configuration. Only the following configurations are officially supported:

- ▶ Peer-to-peer replication
- ▶ Client/server replication
- ▶ Single replication
- ▶ Custom replication

Single replication in a cluster is the default setting. You can also modify the number of replicas within a cluster through the replication domain.

Peer-to-peer topology

Each application server stores sessions in its own memory. It also stores sessions to and retrieves sessions from other application servers. Each application server can retrieve sessions from other application servers. Each application server can also provide sessions to other application servers.

The basic peer-to-peer topology (using client, server, or both mode for replication) is the default configuration and has a single replica (you can also add additional replicas by configuring the replication domain). The example of peer-to-peer replication is shown in Figure 28-8 on page 978.

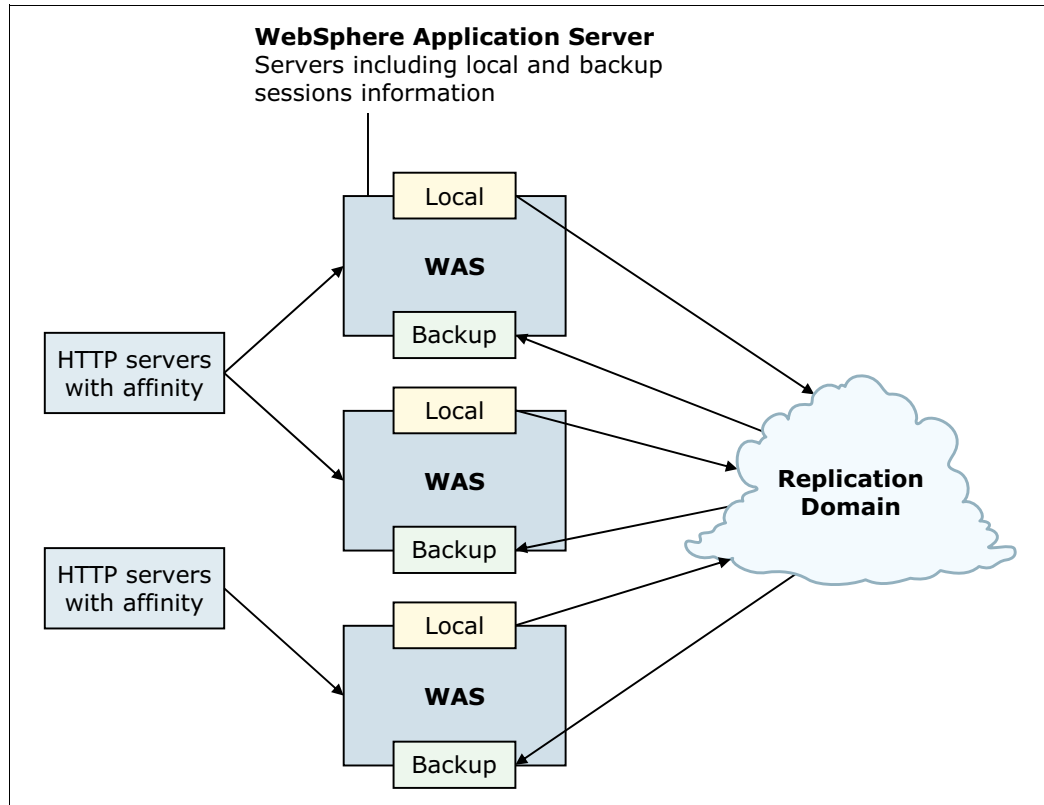


Figure 28-8 Example of peer-to-peer topology

Session hot failover

A new feature called session hot failover was added to WebSphere Application Server V8.5. This feature is only applicable to the peer-to-peer mode. In a cluster environment, session affinity in the web server plug-in for WebSphere Application Server routes the requests for a given session to the same server. If the current owner server of the session fails, the web server plug-in routes the requests to another appropriate server in the cluster. This feature causes the web server plug-in to failover to a server that already contains the backup copy of the session, therefore avoiding the overhead of session retrieval from another server containing the backup.

Client/server topology

The client/server configuration, used to attain session affinity, consists of a cluster of servers that are configured as *client only* and *server only*. The servers, configured as *server only*, are dedicated replication servers that store sessions and provide session information replication clients. These replication servers do not respond to user web requests. Client replication servers send session information to the replication servers and retrieve sessions from the replication servers. They respond to user web requests and store only the sessions that belong to themselves. Figure 28-9 on page 979 shows the example of client/server mode.

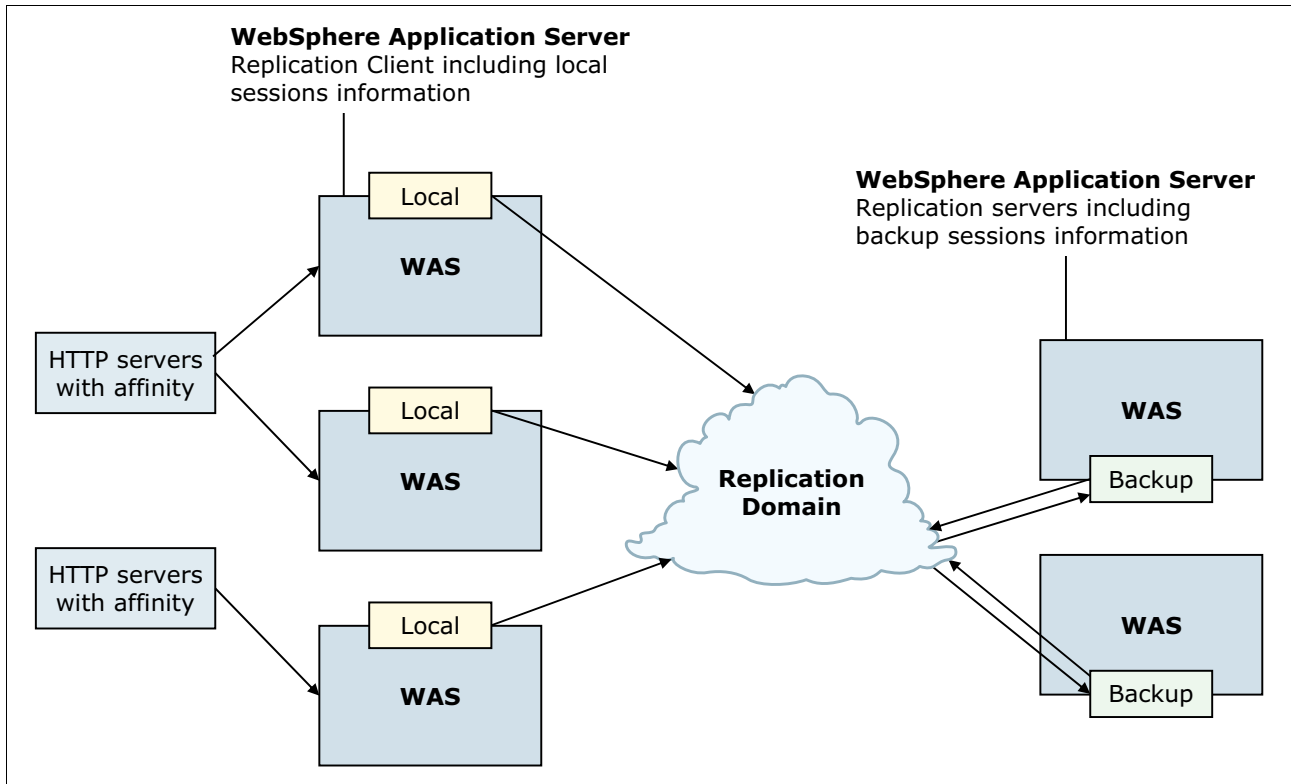


Figure 28-9 Client-Server topology

Table 28-2 shows the comparison of two configuration topologies.

Table 28-2 The difference between peer-to-peer topology and client/server topology

| Topology | Advantage | Disadvantage |
|--------------|--|---|
| Peer-to-peer | Each peer contains the replication copies, no additional processes and products are required to avoid a single point of failure. | It consumes large amounts of memory in a cluster environment that contains many servers and work with many users. |
| | Session hot failover feature. | Large communication usage for every change of a session because change must be replicated to all application servers. |

| Topology | Advantage | Disadvantage |
|---------------|---|--|
| client/server | Reduces the consumption of memory on each server because only session backup is stored on replication servers. | Need additional servers that must be configured and maintained as replication servers. |
| | Recycle a backup server without affecting the servers that are running the application. | Need multiple replication servers configured to avoid a single point of failure. |
| | No need to have a one-to-one communication for session replicas between servers within a domain. | Need to start the backup replication servers first to avoid unexpected timing windows for replication clients. |
| | Running applications on lower-end hardware while running replication servers on powerful computers to reduce hardware cost. | |

Replication domain

Replication domains are used for replication by the HTTP session manager, dynamic cache service, and stateful session bean failover components. All memory-to-memory data replication service instances that need to share information must be in the same replication domain.

Note: Create a separate replication domain for each consumer. For example, create one replication domain for the session manager and another separate replication domain for the dynamic cache. Configure only one replication domain when you configure session manager replication and stateful session bean failover. Using this pattern ensures that the backup state information of HTTP sessions and stateful session beans reside on the same application server.

To create, view, and configure the replication domains on the administrative console, click **Environment** → **Replication domains** → *replication_domain_name* (as shown in Figure 28-10 on page 981).

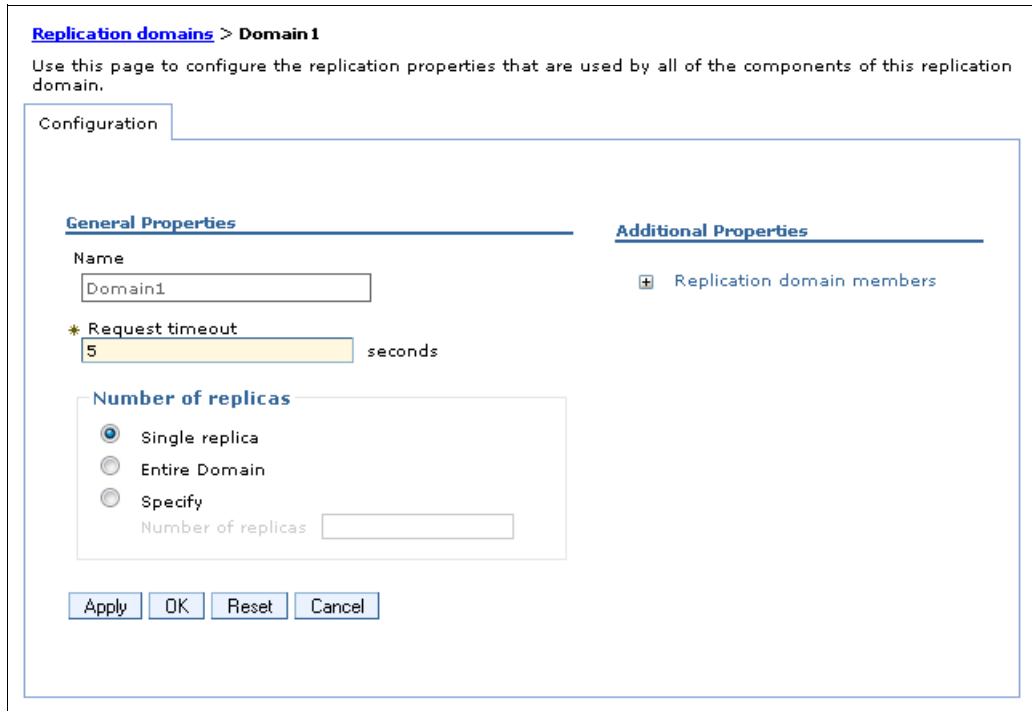


Figure 28-10 Replication Domains

In Figure 28-10:

- ▶ Name: Specifies a unique name within the cell for the replication domain.
- ▶ Request timeout: Specifies how long a replication domain consumer waits when requesting information from another replication domain consumer before it gives up and assumes that the information does not exist. The default value is five seconds.
- ▶ Number of replicas: Specifies the number of replicas created for every session entry.
 - Single replica: Every session is replicated to exactly one other application server within the domain. This is the default value.
 - Entire domain: Each object is replicated to every application server within the domain.
 - Specify: You must specify the number of replicas that you want created for each session.

Enabling memory-to-memory replication

Complete the following actions to enable the memory-to-memory replication.

Preparation

Complete these tasks before enabling data for the replication service:

1. Create a cluster consisting of at least two application servers. To learn more about how to create a cluster, refer to chapter 15.1, “Clustering” on page 520.
2. Install applications to the cluster.

Creating a replication domain for this cluster

Refer to “Replication domain” on page 980 to complete this task.

Configuring the cluster members

To configure the cluster members:

1. Click **Servers** → **Server Types** → **WebSphere application servers** → *server_name*.
2. Under Container Settings section, click **Session management**.
3. Click **Distributed environment settings** under Additional Properties.
4. The configuration tab is then opened (previously shown in Figure 28-6 on page 974). Click **Memory-to-memory replication**.
 - a. Choose a replication domain.
 - b. Select the replication mode. Clicking **Both client and server** identifies this topology as a peer-to-peer topology. In a client/server topology, click **Client only** for servers that respond to user requests. Click **Server only** for those servers that are used as replication servers. See Figure 28-11.

Application servers > was85Cluster01Member01 > Session management > Distributed environment settings > Memory-to-memory replication

Use this page to configure memory-to-memory replication for failure recovery.

Configuration

General Properties

Replication domain
Domain1

Replication mode
Both client and server

Apply OK Reset Cancel

Figure 28-11 Memory-to-memory replication setting

5. Click **OK**.
6. Optional: If you want to change the default tuning parameters, click **Custom tuning parameters**.
7. Click **OK** and Save the changes on the Memory-to-memory replication page and the Session management page.
8. Repeat these steps for the rest of the application servers in the cluster.
9. Synchronize the configuration, and restart the cluster.

Note: You must configure all session managers connected to a replication domain to have the same topology. If one session manager instance in a domain is configured to use the client/server topology, the rest of the session manager instances in that domain must be a combination of servers configured as *Client only* and *Server only*. If one session manager instance is configured to use the peer-to-peer topology, all session manager instances must be configured as *Both client and server*.

Multiple data replication service instances that exist on the same application server and are configured to be part of the same domain must have the same mode. Multiple instances exist on the same application server due to session manager memory-to-memory configuration at various levels.

HTTP session replication in the z/OS controller

WebSphere Application Servers on z/OS, that are enabled for memory-to-memory replication, can store replicated session data in the z/OS controller. They can also replicate data to other WebSphere Application Servers. For more information, refer to the following website:

<http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-zos&topic=cprsmemory>

28.4 Session affinity

Session affinity is a favored relationship between a client and application server. Affinity established between the two can override load-balancing algorithms. Though load-balancing can be overridden it is done in a way that contributes to application performance by using in-memory cache. The application server that serves the clients first request creates this affinity through session information and cookies.

28.4.1 What is the session affinity

The session management facility requires an affinity mechanism so that all requests for a particular session are directed to the same application server instance in the cluster. This routing ensures that all of the HTTP requests are processed with a consistent view of the user's HTTP session. This requirement conforms to the Servlet 2.3 specification, in that multiple requests for a session cannot coexist in multiple application servers, and provides better performance as sessions are cached in local memory.

WebSphere Application Server ensures that session affinity is maintained. Each server ID (clone ID or partition ID) is appended to the session ID. When a session is created, its ID is passed back to the browser as part of a cookie or URL encoding. When the browser makes further requests, the cookie or URL encoding is sent back to the web server. The web server plug-in for WebSphere Application Server examines the HTTP session ID in the cookie or URL encoding, extracts the unique ID of the cluster member handling the session, and forwards the request.

The JSESSIONID cookie can be divided into the following parts:

- ▶ Cache ID
- ▶ Session ID
- ▶ Separator
- ▶ Clone ID
- ▶ Partition ID

For example, if the JSESSIONID cookie is 0000SHOQmBQ8EokAQtzl_HYdxIt:vuel491u. Each parts mapping of the JSESSIONID is shown in Table 28-3.

Table 28-3 Cookie mapping

| Content | Value in the example |
|------------|-------------------------|
| Cache ID | 0000 |
| Session ID | SHOQmBQ8EokAQtzl_HYdxIt |
| separator | : |

| Content | Value in the example |
|-------------------------|----------------------|
| Clone ID / Partition ID | vuel491u |

Within a cluster, clone ID is used to identify the cluster member when routing the request to application servers within a cluster. It must be unique to maintain session affinity. When memory-to-memory replication in peer-to-peer mode is selected, a partition ID will be used instead of a clone ID.

The clone ID can be seen in the web server plug-in configuration file `plug-in-cfg.xml`. You can use the session management custom property `HttpSessionCloneId` to change the clone ID of the cluster member, as described in 28.2.1, “Session management properties” on page 966. You can also configure the clone ID for each application server by using scripting. For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=txml_httpsessionclone

Note: Session affinity can still be broken if the cluster member handling the request fails. To avoid losing session data, use persistent session management. In persistent sessions mode, the cache ID and server ID (clone ID or partition ID) changes in the cookie when there is a failover or when the session is read from the persistent store.

28.4.2 Session affinity and failover

Sessions created by cluster members in the cluster environment share a common persistent session store. The session affinity, in the web server plug-in for WebSphere Application Server, routes the requests for a given session to the same server. If the current owner server instance of the session fails, the web server plug-in routes the requests to another appropriate server in the cluster. The user can continue to use session information without impact. Figure 28-12 on page 985 shows an example. Note that only a single cluster member can control and access a given session at a time.

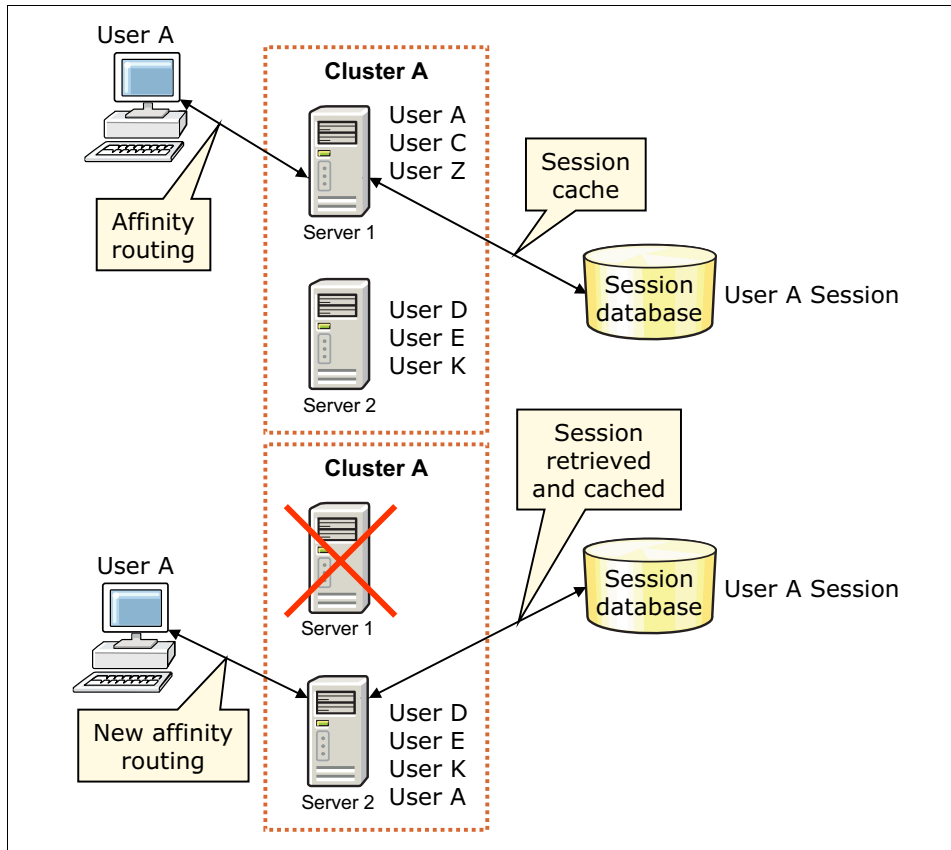


Figure 28-12 Session affinity and failover

After a server failure, web server plug-in redirects the user to another cluster member, and the user's session affinity switches to this replacement cluster member. After the initial read from the persistent store, the replacement cluster member places the user's session object in the in-memory cache. From then on, requests for that session go to the selected cluster member. The requests for the session can go back to the failed cluster member when it is recovered.

Session affinity with on demand router

When using Intelligent Management, the on demand router (ODR) uses session affinity for routing work requests. After a session is established on a server, later work requests for the same session go to the original server. Further more, the ODR can provide session affinity support for the following scenarios:

- ▶ When the ODR routes to servers that are not WebSphere Application Server products.
- ▶ When using the dynamic cluster.
- ▶ When the application using a custom session setting, such as a session ID cookie name, is something other than JSESSIONID.

For more information about this feature, refer to the following information center website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=twve_odrpassive

Session affinity consideration

WebSphere provides session affinity on a best-effort basis. There are narrow windows where session affinity fails, for example:

- ▶ When a cluster member is recovering from a crash, a window exists where concurrent requests for the same session can be received by different cluster members. To avoid or limit this situation, set the retry timeout to a smaller value, if your environment allows.
- ▶ A server overload or long processing times can cause requests that belong to the same session to go to different cluster members.

The session affinity can impact or be impacted by other system configurations, for example:

- ▶ Health policies from Intelligent Management can impact the session affinity. If the policies that define the action are putting a server into maintenance mode then session affinity to the server is broken.
- ▶ The cluster routing options and workload management options are impacted by session affinity. After a session is created at the first request, all the subsequent requests must be served by the same member of the cluster. So the predefined load balancing options will not work.
- ▶ High availability also impacts the session affinity. The high availability manager can help obtain better session peer to peer replication. When the current owner server instance of the session fails, then the requests can be routed to another appropriate server in the cluster for failover.

28.5 Session management tuning

This section includes guidance for performance tuning of the WebSphere Application Server session support.

28.5.1 Session performance considerations

The administrators have several options for improving the performance of session management:

- ▶ Reducing session object size
Large session objects consume the JVM memory and impact the performance of session persistence and replication. Consider reducing object size allowance settings.
- ▶ Reducing the session cache size
This action reduces the memory required by the cache. Enabling overflow cache and persistence session management can impact the design. You need to balance the options of memory consumption and retrieval frequency.
- ▶ Creating additional application servers
Depending on the memory and CPU capacity of the machines involved, you can add additional server instances or physical machines. With more instances or machines, you can dispatch the application for workload balance and operate without it. The session affinity can guarantee the consistent user experience.
- ▶ Invalidating unneeded sessions
Invalidate any session that the user is no longer using to release session cache and session persistence storage. You can also use the schedule-based invalidation, where scans for invalid objects can be deferred to low demand time frames.

- ▶ Increasing available memory

Increasing the JVM heap size allows it to hold a larger session cache. However, you need to balance the overall application performance keeping in consideration garbage collection and large heap size.
- ▶ Reducing the session timeout interval

By reducing this interval to match the average user, the session manager purges the sessions from the cache and the persistent store. However, avoid setting too low a value, which can frustrate users or trigger frequent and expensive scans of the persistent store for timed out sessions.
- ▶ Utilize session affinity to help achieve higher cache hits
- ▶ Reducing persistent store I/O:
 - Optimize the use of the HttpSession within a servlet. Only store the minimum amount of data required in HttpSession.
 - Specify `session=false` in the JSP directive for JSP pages that do not need to access the session object.
 - Optimize choose the write frequency mode
- ▶ Using Multi-row persistent sessions for database persistence

This method reduces both the data retrieval time and the serialization impact. Even with this feature's support, you still need to keep the session objects small.

Note: Avoid circular references within sessions if using multi-row session support. The multi-row session support does not preserve circular references in retrieved sessions.

For more information about best practices using sessions, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=cprs_best_practice

28.5.2 Session management tuning

WebSphere Application Server session support has many options for tuning session performance to match the checklist, as mentioned in 28.5.1, “Session performance considerations” on page 986. These options support administrator flexibility in determining the performance and failover characteristics for their environment. Table 28-4 shows the tuning features summary. In this section, we cover *write frequency*, *write contents* and *multi-row schema*. For more information about other tuning features, refer to section 28.2, “Session management configuration” on page 966.

Table 28-4 Summary of tuning features

| Feature or option | Goal | Applies to sessions in memory, database, or memory-to-memory |
|-------------------|---|--|
| Write frequency | Minimize database write operations. | Database |
| Session affinity | Access the session in the same application server instance. | All |
| Multi-row schema | Fully utilize database capacities. | Database |

| Feature or option | Goal | Applies to sessions in memory, database, or memory-to-memory |
|----------------------------------|--|--|
| Base in-memory session pool size | Fully utilize system capacity without overburdening system. | All |
| Write contents | Allow flexibility in determining what session data to write | Database |
| Scheduled invalidation | Minimize contention between session requests and invalidation of sessions. | Database |
| Table space and row size | Increase efficiency of write operations to database. | Database (DB2 only) |

WebSphere Application Server provides predefined performance tuning options for the following distributed session persistence tuning parameters:

- ▶ Write frequency settings: How often session data is written
- ▶ Write contents settings: How much data is written
- ▶ Session cleanup settings: When the invalid sessions are removed from the system

To view and edit the tuning parameters:

1. Go to the appropriate level of Session Management described in 28.2.2, “Accessing session management properties” on page 967.
2. Under Additional Properties, click **Distributed environment settings** → **Custom tuning parameters**. The configuration page, Figure 28-13 on page 989, is then displayed with the following options:
 - Very high (optimize for performance)
 - High
 - Medium
 - Low (optimize for failover)

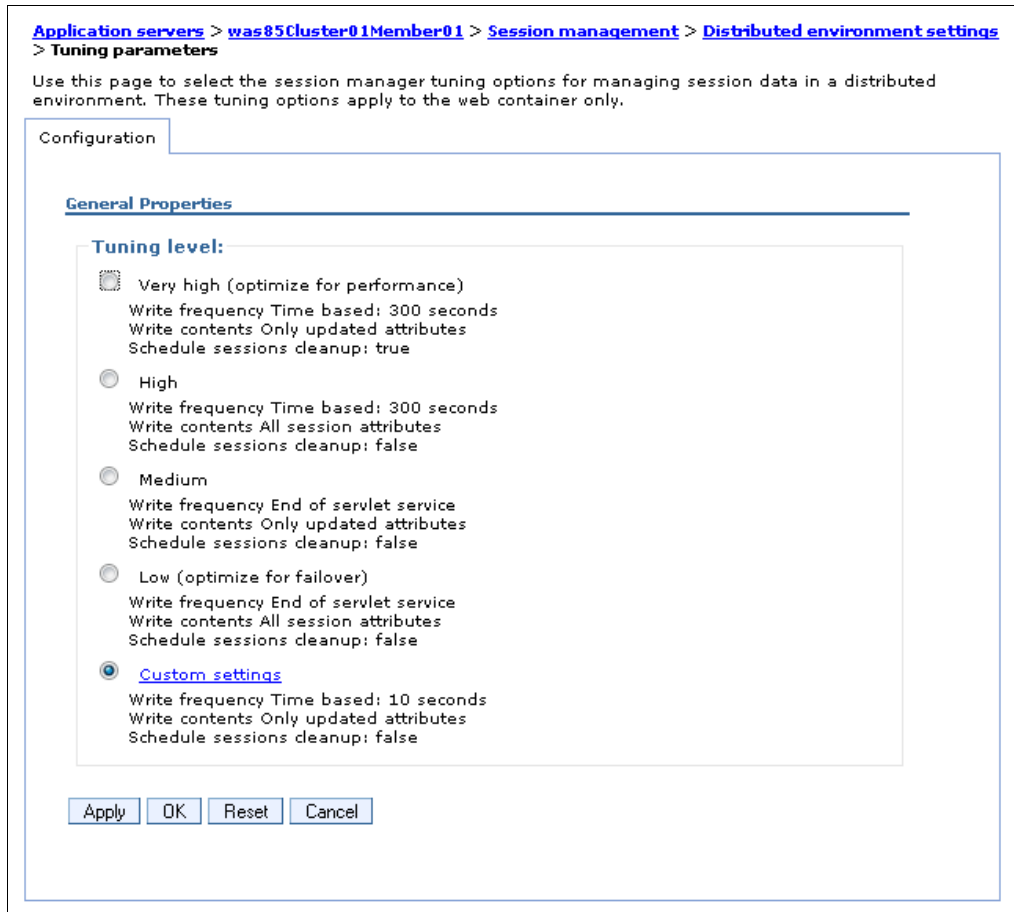


Figure 28-13 Tuning parameters for session management

For detailed information about the values of each tuning option, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=tune_para

3. You can use predefined options or customize the tuning parameters, as previously shown in Figure 28-13. Click the **Custom settings** hotlink on the configuration page. After the hotlink opens, the available tuning parameters are displayed, as shown in Figure 28-14 on page 990.

[Application servers](#) > [was85Cluster01Member01](#) > [Session management](#) > [Distributed environment settings](#) > [Tuning parameters](#) > [Custom tuning parameters](#)

Use this page to specify tuning parameters for session management.

Configuration

General Properties

Write frequency

End of servlet service
 Manual update
 Time based: seconds

Write contents

Only updated attributes
 All session attributes

Schedule sessions cleanup:

Specifies distributed sessions cleanup schedule
 First time of day (0-23):
 Second time of day (0-23):

Figure 28-14 Custom tuning parameters

The following sections go into more detail about these custom settings.

Writing frequency settings

The settings determine how often session data is written to the persistent data store:

► End of servlet service

If the session data changed, it is written to the persistent store after the servlet finishes processing an HTTP request. Therefore, the session data is written to the persistent store at the completion of the `HttpServlet.service()` method call. The content written to the persistent store is controlled by the write contents option setting.

► Manual update

The modified session data and last access time are written to the persistent store when the `IBMSession.sync()` method is called on the object:

- Manual update mode requires an developer to use the `IBMSession.sync()` for managing sessions.
- If manual update mode is specified but the servlet or JSP terminates without invoking the `sync()` or the application does not invoke the `sync()`, the session manager saves content differently. During those circumstances, the session manager saves the contents of the session object into the local session cache, not the persistent data store. The session manager updates only the last access time in the persistent store asynchronously and at a later time.
- The session data that is written to the persistent store is controlled by the write contents option selected.

► Time-based

The session data is written to the persistent store based on the specified write interval value:

- The expiration of the write interval does not necessitate a write to the persistent store unless the session has been touched.
- Only the last access time is written to the persistent store, if the session write interval has expired but the session has only been retrieved.
- The session data that is written is dependent on the write contents settings.
- Time-based write allows the servlet or JSP to issue the `IBMSession.sync()` method to force the write of session data to the database.
- If the time between session servlet requests for a particular session is greater than the write interval, the session effectively is written after each service method invocation.
- The session cache needs to be large enough to hold all of the active sessions as persistent store writes increase. Extra persistent store writes occur because the receipt of a new session request can result in writing the oldest cached session to the persistent store.
- The session invalidation time must be at least twice the write interval to ensure that a session is not inadvertently invalidated prior to being written to the persistent store.
- A newly created session is always written to the persistent store at the end of the service method.

Note: Time-based writes require session affinity for session data integrity. You can gain potential performance improvements by reducing the frequency of persistent store writes.

Consider an example where the web browser accesses the application once every five seconds. The following three modes are available for selection and manage that situation differently:

- In *End of servlet service* mode, the session is written out every five seconds.
- In *Manual update* mode, the session is written out when the servlet issues `IBMSession.sync()`, as shown in Example 28-5.

Example 28-5 Using the IBMSession class to manually update the persistent store

```
public void service (HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException {
    // Use the IBMSession to hold the session information and manual update
    method sync()
    com.ibm.websphere.servlet.session.IBMSession session =
    (com.ibm.websphere.servlet.session.IBMSession)req.getSession(true);
    Integer value = 1;
    //Update the in-memory session stored in the cache
    session.putValue("MyManualCount.COUNTER", value);
    //The servlet saves the session to the persistent store
    session.sync();
}
```

}

- ▶ In *Time-based* mode, the servlet or JSP does not need to use the `IBMSession` class or issue the `IBMSession.sync()` method. If the write interval is set to 120 seconds, the session data is written out every 120 seconds.

Last access time attribute: The last access time attribute is updated each time the session is accessed by the servlet or JSP, whether or not the session is changed. This update is done to make sure the session does not time out.

Write contents settings

WebSphere supports the following modes for writing session contents to the persistent store:

- ▶ Write changed (the default)
Writes only the session data that was updated using `setAttribute()` and `removeAttribute()`.
- ▶ Write all
Writes all the session data to the database.

When using database persistence, the behavior for subsequent servlet or JSP requests for this session varies depending on whether the single-row or multi-row database mode is in use, as follows.

Table 28-5 Write content setting for single-row or multi-row schemas

| Write Contents | Behavior with single-row schema | Behavior with multi-row schema |
|----------------|--|---|
| Write changed | If any session attribute is updated, all objects bound to the session are written. | Only the session data modified through <code>setAttribute</code> method or <code>removeAttribute</code> method calls is written. |
| Write all | All bound session attributes are written. | All session attributes that currently reside in the cache are written. If the session never left the cache, all session attributes are written. |

The combination of the Write all mode with a time-based write can greatly reduce the performance penalty and essentially give you optimum performance.

Single and multi-row schemas (database persistence)

When using the single-row schema, each user session maps to a single database row, which is WebSphere's default configuration for persistent session management. When using the multi-row schema, each user session maps to multiple database rows, with each session attribute mapping to a database row. Table 28-6 gives the design considerations for choosing single-row or multi-row.

Table 28-6 Single-row compared to multi-row

| Programming concepts on usage | Application scenario |
|-------------------------------|--|
| Benefit of single-row | You can read/write all values with just one record read/write, which takes up less space in a database because you are guaranteed that each session is only one record long. |

| Programming concepts on usage | Application scenario |
|-------------------------------|---|
| Limitation of single-row | There is a 2 MB limit of stored data per session. The sum of all session attributes is limited to 2 MB. |
| Benefit of multi-row | The application can store an unlimited amount of data. You are limited only by the size of the database and a 2 MB-per-record limit. The application can read individual fields instead of the entire record. When large amounts of data are stored in the session but only small amounts are specifically accessed during a given servlet's processing of an HTTP request, multi-row sessions can improve performance by avoiding unneeded Java object serialization. |
| Limitation of multi-row | If data is small in size, you might not want the extra impact of multiple row reads when everything can be stored in one row. |

Tip: In the case of multi-row usage, design your application data objects so they do not have references to each other. This prevents circular references.

Consider them all

Table 28-7 shows the summary of considering row-type, write contents, and write frequency for best results.

Table 28-7 Write contents, row-type, and write frequency comparison

| Row type | Write contents | Write frequency | Action for setAttribute | Action for removeAttribute |
|----------|----------------|--|---|---|
| Single | Write changed | End of servlet service / Manual update | If any of the session data changed, write all of this session's data from cache. ^a | If any of the session data changed, write all of this session's data from cache. ^a |
| | | Time-based | If any of the session data changed, write all of this session's data from cache. ^a | If any of the session data changed, write all of this session's data from cache. ^a |
| | Write All | End of servlet service / Manual update | Always write all of this session's data from cache. | Always write all of this session's data from cache. |
| | | Time-based | Always write all of this session's data from cache. | Always write all of this session's data from cache. |

| Row type | Write contents | Write frequency | Action for setAttribute | Action for removeAttribute |
|----------|----------------|--|--|---|
| Multiple | Write changed | End of servlet service / Manual update | Write only thread-specific data that changed. | Delete only thread-specific data that was removed. |
| | | Time-based | Write thread-specific data that changed for <i>all</i> threads using this session. | Delete thread-specific data that was removed for <i>all</i> threads using this session. |
| | Write all | End of servlet service / Manual update | Write all session data from cache. | Delete thread-specific data that was removed for <i>all</i> threads using this session. |
| | | Time-based | Write all session data from cache. | Delete thread-specific data that was removed for <i>all</i> threads using this session. |

a. When a session is written to the database using single-row mode, all of the session data is written. Therefore, no database deletes are necessary for properties removed with the removeAttribute() method because the write of the entire session does not include removed properties.

28.5.3 Session database tuning

To maximize performance tuning, it is required to tune the underlying session persistence database. WebSphere provides a first step by creating an index for the sessions table when the table is created. The index is composed of the session ID, the property ID for multi-row sessions, and the web application name.

Most database managers provide a great deal of capability in tuning at the table or table space level. However, creating a separate database or instance provides the most flexibility in tuning.

In general, tune and configure the database appropriately for the database that experiences a great deal of I/O. The database administrator (DBA) monitors and tunes the database buffer pools, database log size, and write frequency. Additionally, maximizing performance requires striping the database or instance across multiple disk drives and disk controllers and using any hardware or operating system buffering that is available to reduce disk contention.

Managing your session database connection pool

When using persistent session management, the session manager interacts with the session database through a WebSphere Application Server data source. Each data source controls a set of database connections known as a *connection pool*. The maximum pool size represents the number of simultaneous accesses to the persistent session database available to the session manager. For high-volume web sites, simultaneous data source queuing can impact the overall performance of the web application. However, each connection represents memory impact, so a large pool decreases the memory available for WebSphere to execute applications. To avoid memory issues, performance tuning is needed to balance the optimal setting for a given application.

For more information about the data source and connection pool, consult the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=RRAPProperty_displayName

Note: The session affinity routing, combined with session caching, reduces database read activity for session persistence. Likewise, manual update write frequency, time-based write frequency, and multi-row persistent session management reduce unnecessary writes to the persistent database. Incorporating these techniques can also reduce the size of the connection pool required to support session persistence for a given web application.

28.6 Stateful session bean failover

Each EJB container provides a method for stateful session beans to failover to others. When enabled, all stateful session beans in the container can failover to another instance of the bean and still maintain the session state. Stateful session bean uses the functions of the data replication service and workload management. In contrast to the HTTP session persistence, stateful session EJB availability is handled by using only memory-to-memory replication. Using the EJB container properties, you can specify a replication domain for the EJB container and enable the stateful session bean failover using memory-to-memory replication.

You can also override the parent object's stateful session bean replication settings from the module level. This action enables you to specify whether failover occurs for the stateful session beans at the EJB module level or container level. The following two examples relate how you can enable failover for specific results:

- ▶ You want to enable failover for all applications except for a single application. Enable failover at the EJB container level and override the setting at the application level to disable failover on the single application.
- ▶ You want to enable failover for all applications except for a single module of an application. Enable failover at the EJB container level and then override the setting at the module application level to disable failover on the single module.

28.6.1 Enabling stateful session bean failover

In this section, we discuss how to enable stateful session bean failover.

Configuring stateful session bean failover at the EJB container level

To view and edit stateful session bean failover properties at the EJB container level, complete the following steps:

1. Click **Servers** → **Server Types** → **WebSphere application servers** → *server_name*.
2. Under the Container Settings section of the Configuration tab, click **EJB Container Settings** → **EJB container**.
3. Under the General Properties section, select the **Enable stateful session bean failover using memory-to-memory replication** option, as shown in Figure 28-15 on page 996

This option is disabled until you define a replication domain. There is a *memory-to-memory replication* hyperlink to help you configure the replication settings. If no replication domains are configured, the link takes you to a window where you can create one. If at least one domain is configured, the link takes you to a window where you can select the replication settings to be used by the EJB container

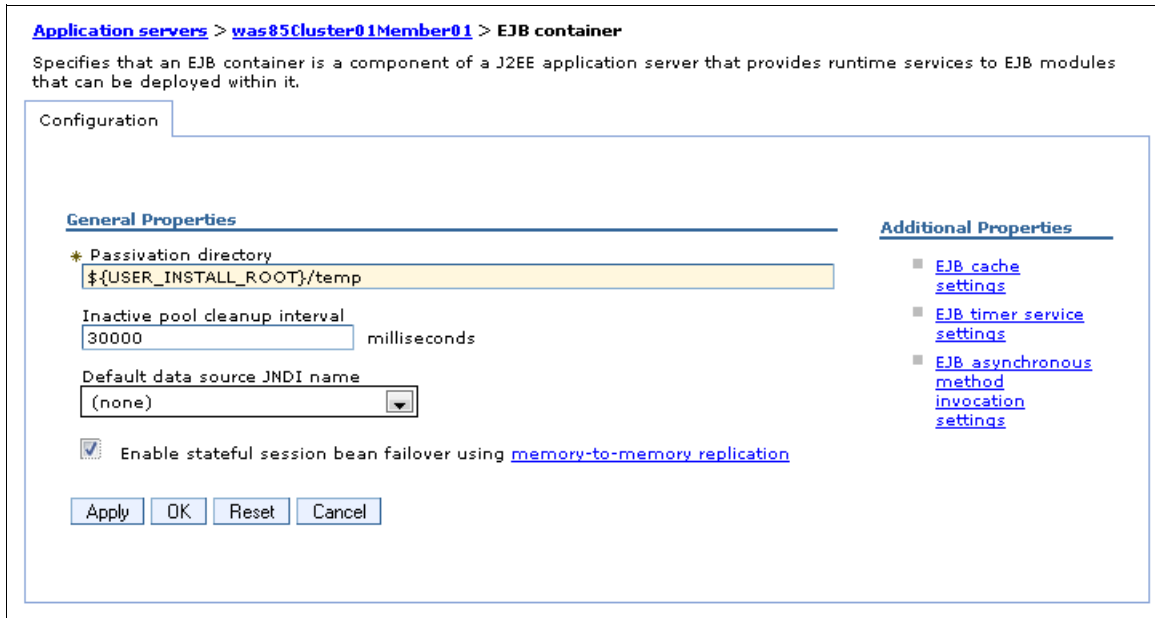


Figure 28-15 Stateful session bean failover settings at the container level

4. Click **OK** and **Save** your changes.

Configuring stateful session bean failover at the applications level

To access and edit stateful session bean failover properties at the EJB application level, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications** → **application_name**.
2. Select your choice of replication settings. Under the Enterprise Java Bean Properties section of the Configuration tab, select **Stateful Session Bean Failover Settings**. The Stateful Session Bean Failover Settings panel appears, as shown in Figure 28-16.

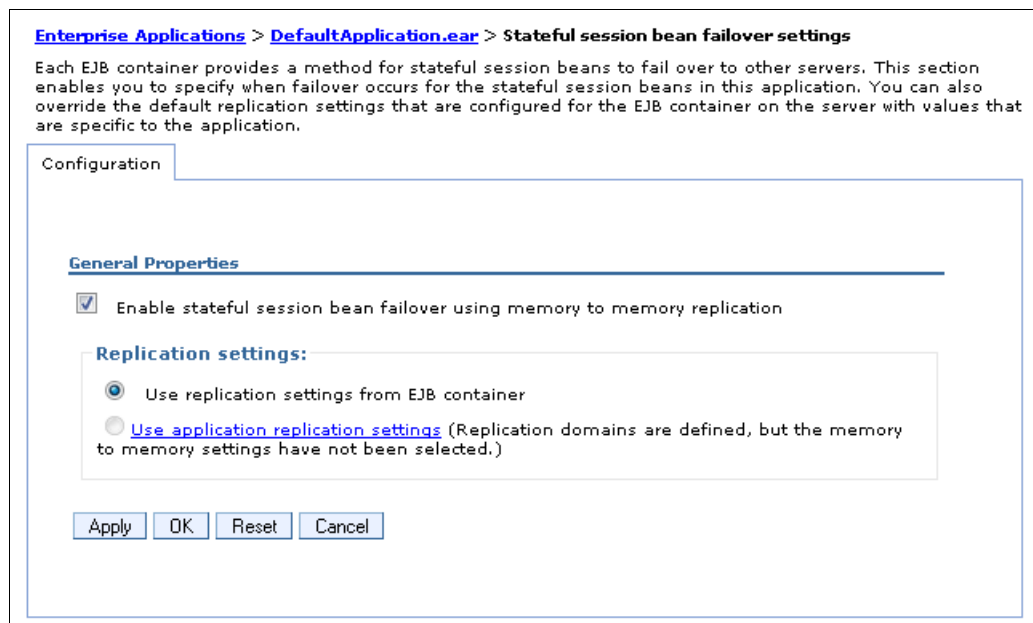


Figure 28-16 Stateful session bean failover settings at the application level

In Figure 28-16 on page 996:

- Enable stateful session bean failover using memory-to-memory replication
This option enables stateful session bean failover. If you want to disable the failover, clear this option. Click the check box.
- Use replication settings from EJB container
If you select this option, any replication settings that are defined for this application are ignored.

Important: When engaging the *use replication settings from EJB container* option, you must configure memory-to-memory replication at the EJB container level. Otherwise, the settings on this window are ignored by the EJB container during server start, and the EJB container logs a message that indicates that stateful session bean failover is not enabled for this application.

- Use application replication settings
If you select this option, you override the EJB container settings. This option is disabled until you define a replication domain. This selection has a *memory-to-memory replication* hyperlink to help you configure the replication settings. If no replication domains are configured, the link takes you to a window to create one. If at least one domain is configured, the link takes you to a window where you can select the replication settings to be used by the application.

3. Click **OK** and Save your changes.

You can also configure stateful session bean failover at the application level using the `wsadmin` script. For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=txml_ejbsfsbapp

Configuring stateful session bean failover at the EJB modules level

To access and edit stateful session bean failover properties at the EJB module level, complete the following steps:

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications** → ***application_name***.
2. Under Modules section, select **Manage Modules**.
3. Select the JAR file that you want to configure.
4. Under Additional Properties, select **Stateful session bean failover settings**.
5. Select **Enable stateful session bean failover using memory to memory replication**.
6. Select your choice of Replication settings. You have a choice of two radio buttons:
 - Use application or EJB container replication settings: If you select this button, any replication settings that are defined for this EJB module are ignored.
 - Use module replication settings: If you select this button, you override the replication settings for the EJB container and application. This button is disabled until you define a replication domain. This selection has a hyperlink to help you configure the replication settings. If no replication domains are configured, when you click the link, then you visit a page where you can create one. If at least one domain is configured, when you click the link, then you visit a page where you can select the replication settings to be used by the EJB container.
7. Select **OK** and Save your changes.

Enabling failover of servants in an unmanaged z/OS server

With WebSphere Application Server V8.5 for z/OS, you can enable the stateful session bean failover among servants. Failover only occurs between the servants of a given unmanaged server. If an unmanaged z/OS server has only one servant, enabling failover has no effect. An unmanaged z/OS server that has failover enabled does not fail over to another unmanaged z/OS server.

For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=tejb_sfsbfzos

28.6.2 Stateful session bean failover consideration

When you enable the stateful session bean failover, consider which of the following impacts and stateful session bean configurations:

- ▶ Stateful session bean activation policy with failover enabled.
- ▶ Stateful session bean use of *container managed units of work* or *bean managed units of work* with failover enabled.
- ▶ If you create either an HTTP session or a stateful session bean that stores a reference to another stateful session bean, make sure the HTTP session and stateful session bean are configured to use the same replication domain.
- ▶ Do not use a local and a remote reference to the same stateful session bean.
- ▶ Avoid referencing non-persistent EJB timers in a stateful session bean instance when failover is enabled.
- ▶ Avoid the use of remote asynchronous methods on stateful session beans.

For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=cejb_sfsbf



Part 6

Maintenance



Managing an environment with the centralized installation manager

WebSphere Application Server V7 introduced the centralized installation manager. Starting from WebSphere Application Server V8, the centralized installation manager evolved with new capabilities and is integrated with both the Installation Manager and the job manager.

The centralized installation manager (CIM) can reduce the number of steps that are required to install and maintain the WebSphere Application Server environment. Its features allow the centralized installation manager to work outside the WebSphere Application Server cell to schedule the installation and maintenance tasks.

The process for managing Version 7 and previous versions differs from the process for managing V8 releases. Starting with CIM Version 8, the Installation Manager is used to install the product on remote machines, where Versions 6.1 and 7 use ISMP and Update Installer.

This chapter includes the following topics:

- ▶ The centralized installation manager prerequisites
- ▶ Planning considerations
- ▶ Using centralized installation manager with V8 releases
- ▶ Using centralized installation manager with prior releases
- ▶ Managing V8 release environments with the centralized installation manager
- ▶ Managing V6.1 and V7 with the centralized installation manager

For additional information about working with the centralized installation manager, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tins_cim_overview.html

29.1 The centralized installation manager prerequisites

To avoid issues when using the centralized installation manager, ensure that your platform satisfies the requirements listed in this section.

29.1.1 Linux and UNIX target requirements

The centralized installation manager, through RXA, uses SSH Version 2 to access UNIX and Linux target workstations. This usage requires using either OpenSSH 3.6.1 (or, if accessing AIX targets, OpenSSH 4.7) or Sun SSH 1.1 on the target hosts.

Note that OpenSSH 3.7.1, or higher, contains security enhancements not available in earlier releases, and is recommended.

Using OpenSSH: OpenSSH Version 4.7.0.5302 for IBM AIX Version 5.3 is not compatible with Remote Execution and Access Version 2.3. If your target systems are running AIX Version 5.3 with OpenSSH Version 4.7.0.5302 installed, the file transfer might stop in the middle of the transfer. To avoid this problem, revert the OpenSSH version from Version 4.7.0.5302 to Version 4.7.0.5301

Using the SSH protocol

RXA does not supply SSH code for UNIX operating systems. You must ensure that SSH is installed and enabled on any target that you want to access using centralized installation manager.

In all UNIX environments, except Solaris, the Bourne shell (`sh`) is used as the target shell. On Solaris targets, the Korn shell (`ksh`) is used instead because of problems encountered with `sh`.

To communicate with Linux and other SSH targets using password authentication, you must edit the `/etc/ssh/sshd_config` file on the targets and set the following property:

```
PasswordAuthentication yes
```

The default value for the `PasswordAuthentication` property is `no`.

After changing this setting, stop and restart the SSH daemon using either of the following commands sequences:

- ▶ `stopsrc -s sshd`
 `startsrc -s sshd`
- ▶ `/etc/init.d/sshd stop`
 `/etc/init.d/sshd start`

Installing a secure shell public key to access remote targets

UNIX platforms generally support the use of SSH protocol. To use the SSH public/private key as an authentication method for accessing remote workstations, SSH must be installed and enabled on the installation target system. On AIX and Linux systems, issue the following command to ensure that SSH is enabled on the installation target:

```
ps -e | grep sshd
```

You can generate an RSA private key and its corresponding public key using the `ssh-keygen` command, as shown in the following example:

```
ssh-keygen t rsa
```

Take the default location for storing the private key and make note of it. If you specify a non-empty string for the passphrase prompt, make sure that you remember the string because you need it when you want to use the generated private key.

Additionally, you must know the location of the SSH public key file on the deployment manager and the administrative ID and password for the installation target. This information is the same administrative ID and password that you use later to install or uninstall software packages on the same installation target.

29.1.2 Windows target requirements

Many RXA operations require access to resources that are not generally accessible by standard user accounts. Therefore, the account names that you use to log in to remote Windows system targets must have administrative privileges.

To ensure that your Windows system targets cooperate with the centralized installation manager, configure the following components:

- ▶ Simple file sharing
- ▶ Firewall
- ▶ Administrative sharing, if the target is Windows Vista 7 or 2008

For more information about Windows system targets configuration, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/cins_cim_rxa_requirements.html

29.1.3 IBM i targets

Use of SSH public/private key authentication is not supported on IBM i platforms.

For information about the IBM i targets pre-configuration, refer to the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tins_cim_gettingstarted.html

29.1.4 Additional requirements

Before using the centralized installation manager to install or uninstall maintenance on IBM AIX operating systems as a non-root user, you must install and configure *sudo*, an open-source tool, on the target AIX operating systems.

To install and configure *sudo*, download *sudo* from the IBM AIX Toolbox for Linux Applications official download website:

<http://www-03.ibm.com/systems/power/software/aix/linux/toolbox/date.html>

After *sudo* is downloaded, log on to the target system as root and issue the following command to install *sudo*:

```
rpm -i sudo-*.rpm
```

If your AIX system does not already have *rpm* installed, you can download an AIX **installp** image for the **rpm** package manager for IBM POWER® from the following website:

<http://www-03.ibm.com/systems/power/software/aix/linux/toolbox/download.html>

Authorize a non-root user ID that you specify to run the `slibclean` command as a root user without providing a password. Next, issue the `visudo` command to add the following entry to the `/etc/sudoers` configuration file (replace `userid` with the real user ID that you will be using):

```
userid ALL = NOPASSWD: /usr/sbin/slibclean
```

Log in with the specified user ID and then issue the `sudo -l` command. If successful, a message that is similar to the following example displays:

```
User userid might run the following commands on this host:
(root) NOPASSWD: /usr/sbin/slibclean
```

If you do not have `sudo` installed or if `sudo` is installed but not configured correctly for the specified user ID, error messages are displayed.

29.2 Planning considerations

This section lists planning considerations for WebSphere Application Server.

Note: Ensure that your environment meets the centralized installation manager prerequisites for your platform before you attempt to work with the centralized installation manager. For details, refer to 29.1, “The centralized installation manager prerequisites” on page 1002.

29.2.1 WebSphere Application Server V8 releases

For WebSphere Application Server V8 releases, you access the centralized installation manager functions through the job manager or deployment manager. Using the job manager or deployment manager, you can perform the following functions:

- ▶ Install, update, and uninstall Installation Manager on remote machines.
- ▶ Install, update, and uninstall WebSphere Application Server V8 release offerings on remote machines.
- ▶ Collect, distribute, and delete files on remote hosts.
- ▶ Manage WebSphere Application Server profiles on remote hosts.
- ▶ Run scripts on remote hosts.
- ▶ Support for z/OS operating system targets.

Important: With the centralized installation manager for Version 8 releases, you cannot install or update Installation Manager on a z/OS target. Prior to working with z/OS targets with the centralized installation manager, ensure that the Installation Manager is already installed.

- ▶ Support to add targets outside the cell
- ▶ Job scheduling

When working with WebSphere Application Server V8 releases and the centralized installation manager, you need an additional 120 MB of disk space for the Installation Manager installation kit. If your environment includes different operating systems, additional disk space is required for other Installation Manager versions.

29.2.2 WebSphere Application Server V6.1 and V7

For WebSphere Application Server V6.1 and V7, you access the centralized installation manager functions using only the deployment manager. The centralized installation manager for Version 6.1 and Version 7 does not support z/OS operating system targets. Using the centralized installation manager for Version 7, you can perform the following functions:

- ▶ Install, update, and uninstall WebSphere Application Server Network Deployment V7 on remote machines
- ▶ Install and uninstall WebSphere Application Server V6.1 and V7 refresh packs, fix packs, and interim fixes on remote machines
- ▶ Install and uninstall customized installation packages (CIPs)

If you plan to use the centralized installation manager for Version 6.1 or Version 7, the disk space that is required increases significantly and varies depending on how many binaries there are in the repository.

The centralized installation manager relies on current information regarding the versions of WebSphere Application Server that are installed on each node. This information is kept current on the deployment manager configuration by the node agent that is running on each node. The deployment manager is aware of the correct versions of WebSphere Application Servers that are installed on each node, if the node agent of each node is started at least once after an update is applied. To ensure that the deployment manager receives this information, the centralized installation manager starts the node agent automatically after each installation or uninstallation of maintenance.

The centralized installation manager relies on the node agent to effectively stop the server processes on the target node. If the node agent is not running, it is up to the administrator to ensure that all the server processes are stopped on the target node before initiating any maintenance update operations on the node.

29.3 Using centralized installation manager with V8 releases

In WebSphere Application Server V8 releases, the centralized installation manager uses the Installation Manager to manage targets. You can access the centralized installation manager using several methods, including the job manager console or the command line. This section provides more information about using aspects of the centralized installation manager with WebSphere Application Server V8 releases.

29.3.1 Installation Manager

Important: WebSphere Application Server V8 releases use Installation Manager. Be sure to use an up-to-date version of Installation Manager when working with the centralized installation manager. The minimum required version of the Installation Manager is Version 1.5.2.

Installation Manager is integrated with the centralized installation manager, deployment manager, and job manager to provide a single, central place of administration for the following tasks:

- ▶ Installing and uninstalling products
- ▶ Updating and rolling back fix packs and interim fixes

► Installing and uninstalling feature packs

Installation Manager provides a common installation technology, which makes it easier and faster to manage your software. Installation Manager can install a desired level of maintenance in single pass, based on GUI commands or response files. One instance of Installation Manager can manage WebSphere and other IBM products, such as Rational.

Installation Manager also allows you to record response files when invoked with the **-record** and **-skipInstall** options. This method makes it easier to prepare a valid response file for your environment. To learn more about Installation Manager options, refer to 2.1, “IBM Installation Manager overview” on page 32 and the information center at the following website:

http://pic.dhe.ibm.com/infocenter/install/v1r5/topic/com.ibm.silentinstall12.doc/topics/t_silent_create_response_files_IM.html

The job manager or deployment manager stores the Installation Manager installation kit in their local directory. By default, the kit resides at the job manager or deployment manager profile_home/IMKit directory. To change the location of the kit, in the job manager or deployment manager console, click **Jobs** → **Installation Manager installation kits**.

You can add multiple Installation Manager installation kits for many platforms. Figure 29-1 illustrates an example of Installation Manager installation kits V1.5.2 for different platforms.

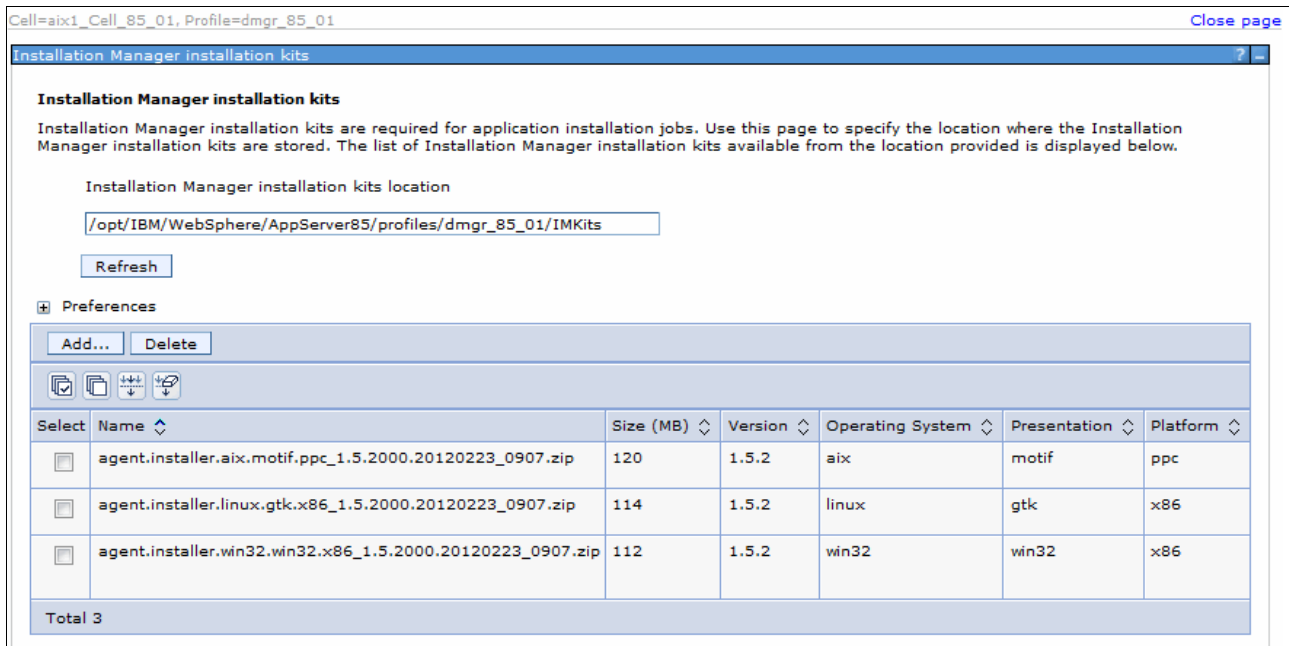


Figure 29-1 Installation Manager installation kit V1.5.2 in the job manager

Note: To install software on your targets using the Installation Manager, you must add the proper version of the Installation Manager installation kit that is valid for your target operating systems.

29.3.2 Accessing the centralized installation manager

You can access the centralized installation manager in WebSphere Application Server V8 releases from the following interfaces:

- ▶ The job manager console
- ▶ The deployment manager console
- ▶ The **AdminTask** object from **wsadmin**

The console for the centralized installation manager tools is available under the Jobs menu and is accessible from the job manager or deployment manager for WebSphere Application Server V8 releases. Figure 29-2 shows the following menu options in the job manager or deployment manager console:

- ▶ **Submit**: Prepares and submits jobs to one or more targets.
- ▶ **Status**: Displays a current status of submitted jobs.
- ▶ **Targets**: Lists defined targets and allows you to create new targets.
- ▶ **Target resources**: Lists target resources, which is useful when working with target groups.
- ▶ **Target groups**: Lists and defines target groups.
- ▶ **Installation Manager installation kits**: Configures the Installation Manager installation kit repository.

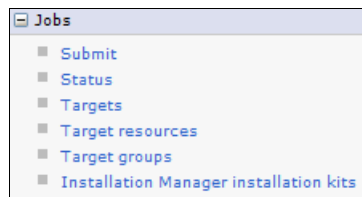


Figure 29-2 Centralized installation manager Jobs menu

You can also work with the centralized installation manager using the command line. You can use the **AdminTask** object by invoking it from the **wsadmin** script from the job manager or the deployment manager host. Example 29-1 presents a Jython example of the **AdminTask** object usage. In this example, a **testConnection** job type invokes the target to verify that the connection between job manager or deployment manager and *yourhost.com* is active.

Example 29-1 AdminTask command used for connection test on WebSphere Application Server

```
AdminTask.submitJob('-jobType testConnection -targetList [yourhost.com]' -username  
admin -password admin)
```

29.4 Using centralized installation manager with prior releases

WebSphere Application Server V8 releases do not require IBM Update Installer or IBM Installation Factory. However, it supports these products in case you plan to use Version 6.1 or Version 7 together with WebSphere Application Server Version 8 or above.

The following sections explain additional components that are used when managing Version 6.1 or Version 7 installations using WebSphere Application Server V8.5.

29.4.1 IBM Update Installer

The centralized installation manager for Version 6.1 and Version 7 installs Update Installer on the target systems. Update Installer installs fix packs and other maintenance on Version 6.1 and Version 7 targets for the central installation manager.

Version note: To avoid issues when using the centralized installation manager, be sure you use Update Installer V7.0.0.15 or later and Installation Factory V7.0.0.15 or later.

If you previously installed Update Installer on any of the target hosts in a directory location other than the *install_root/UpdateInstaller* directory, you might want to consider uninstalling Update Installer with its uninstall process because it is not used by the centralized installation manager. You do not need to uninstall the previous copy of Update Installer for centralized installation manager to work properly.

When you install fix packs or other maintenance on the target systems, the centralized installation manager installs Update Installer if the option is selected. If the version of Update Installer located in the *install_root/UpdateInstaller* directory does not meet the minimum version that is required by the interim fix or fix pack, the centralized installation manager can automatically install a newer version on the target. The automatic install happens only if the newer version is downloaded to the centralized installation manager repository.

You can use the centralized installation manager to install Update Installer on nodes that are federated to the deployment manager cell.

29.4.2 The centralized installation manager repository structure

The centralized installation manager for WebSphere Application Server V6.1 or V7 works with the additional repository that is illustrated in Figure 29-1 on page 1006. This repository consists of directories that contain the installation images for product files, the maintenance files, and the Update Installer files. These directories are located in *app_server_install_root/cimrepos* and use the following directory names and files types:

► UPDI70

This directory holds the 7.0.0.*-WS-UPDI-*.zip files that contains the Update Installer installation images for the operating systems that you want in your repository. For example, the following file might be copied into this directory:

7.0.0.17-WS-UPDI-AixPPC64.zip

► WAS70Updates

This directory holds the *.pak files that contain interim fixes for WebSphere Server Network Deployment V7. You can remove these files when they are no longer required. The following file might be in this directory:

7.0.0.1-WS-WAS-IFPK75887.pak

► WAS70FPn

This directory contains the .pak files that make up a specific fix pack for WebSphere Server Network Deployment V7. The following lists shows examples of the files that might be copied into the WAS70FP1 directory. Refer to the WebSphere Application Server V7 support website for the list of files that are required for each fix pack.

For example, for WebSphere Application Server Network Deployment V7 Fix Pack 17, the following .pak files are copied to the WAS70FP17 directory:

7.0.0-WS-WAS-AixPPC64-FP0000017.pak

7.0.0-WS-WASSDK-AixPPC64-FP0000017.pak

▶ ND61Updates

This directory holds the .pak files that contain interim fixes for WebSphere Server network Deployment V6.1.

▶ ND61FPn

This directory contains the .pak files that make up a specific fix pack for WebSphere Application Server V6.1. Refer to the WebSphere Application Server V6.1 support website for the list of files that are required for each fix pack. For example, for WebSphere Application Server Network Deployment V6.1 Fix Pack 37, the following .pak files are copied to the ND61FP37 directory:

6.1.0-WS-WAS-*platform_architecture*-FP0000037.pak

6.1.0-WS-WASSDK-*platform_architecture*-FP0000037.pak

6.1.0-WS-WASWebSvc-*platform_architecture*-FP0000037.pak

6.1.0-WS-WASEJB3-*platform_architecture*-FP0000037.pak

▶ WAS70

This directory is created during the installation of the centralized installation manager repository. It contains the product installation files for the operating systems in your environment:

jdk.7000.aix.ppc64.zip

was.nd.7000.aix.ppc64.zip

▶ Descriptors

This directory is created when the centralized installation manager is installed and contains the following descriptor files:

InstallPackageND61FP37.xml

InstallPackageND70FP17.xml

29.4.3 Package types

In this section, we describe the package types that the centralized installation manager supports.

Product installation

The descriptor and binary files are included in product packages and are loaded when the product is loaded into the repository. During the product installation, the following descriptors are included:

- ▶ Maintenance for WebSphere Application Server Network Deployment V6.1 descriptor files that are provided by the product installation
- ▶ Maintenance for WebSphere Application Server Network Deployment V7
- ▶ Update Installer for WebSphere Application Server V7
- ▶ WebSphere Application Server Network Deployment V7

Maintenance tool

This package contains the Update Installer, which is the tool used to apply maintenance to a WebSphere Application Server V6.1 or V7 environment. Before using the centralized installation manager to apply maintenance on remote systems, you must download the latest level of Update Installer. You must first install fix packs locally on the deployment manager system using Update Installer.

Refresh and fix packs

With this package type, you can download binary files based on a specific platform. When a refresh or fix pack for IBM WebSphere Application Server is released, it usually comes with a fix pack for Java SDK. The centralized installation manager requires both fix packs in the repository and installs both fix packs to the selected targets.

Interim fixes

This package type is used to apply a small update to the WebSphere Application Server runtime, usually provided by WebSphere support.

Refer to 29.6, “Managing V6.1 and V7 with the centralized installation manager” on page 1036 for more information about adding additional packages or registering a repository.

29.4.4 Accessing the central installation manager

You can manage WebSphere Application Server V6.1 or V7 from the centralized installation manager in WebSphere Application Server V8.5 using the following interfaces:

- ▶ The deployment manager console
- ▶ The `AdminTask` object from `wsadmin`

In the deployment manager console, the centralized installation manager jobs are also available from the Jobs menu, but they are reserved only for WebSphere Application Server V8 releases. Additional tools for WebSphere Application Server V6.1 and V7 only are available by clicking **System administration** → **Centralized Installation Manager**, as shown in Figure 29-3 on page 1011. Shown are the two centralized installation manager sections available from the deployment manager V8.5 console.

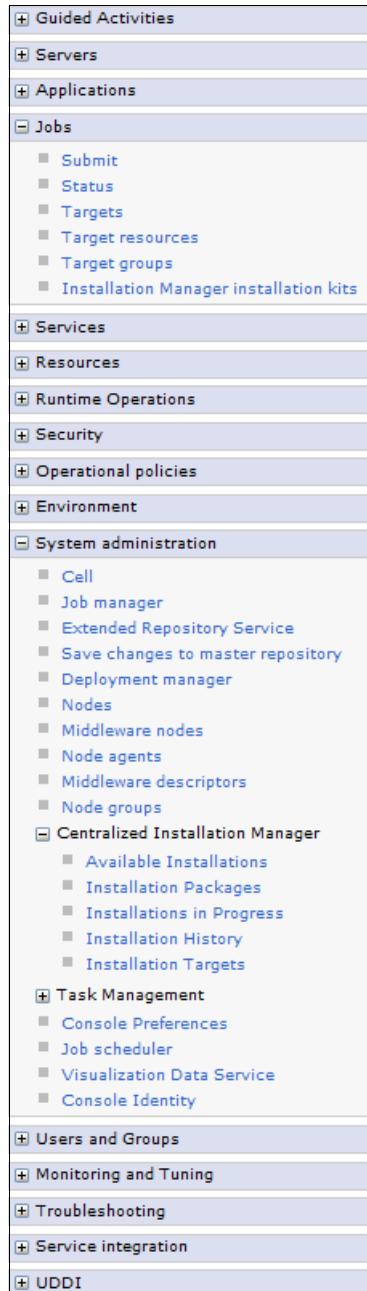


Figure 29-3 Centralized installation manager sections from the deployment manager V8.5 console

The tools for WebSphere Application Server V6.1 and V7 (**System administration** → **Centralized Installation Manager**) are described in the following list:

- Available Installations** Installs installation packages to the targets.
- Installation Packages** Displays all available descriptors and packages in the centralized installation manager repository.
- Installations in Progress** Displays the status of installations that are in progress.
- Installation History** Displays the history log for installations done with the centralized installation manager.
- Installation Targets** Lists and defines installation targets for the centralized installation manager.

To work with the centralized installation manager through the command line, you can use the **AdminTask** object by invoking it from the **wsadmin** script from the deployment manager host. Example 29-2 presents a Jython example of **AdminTask** object usage. In this example, a connection test is made with target host `yourhost.com` from the deployment manager to check if the connection is active.

Example 29-2 AdminTask command used for connection test on WebSphere Application Server V7

```
AdminTask.testConnectionToHost ('[-hostName yourhost.com  
-platformType linux -adminName root -adminPassword password]')
```

Note: The **AdminTask** object uses different methods and parameters when working with the centralized installation manager for WebSphere Application Server V8 releases than for V6.1 or V7. For more information about the available methods, refer to 29.6.11, “The centralized installation manager AdminTask commands” on page 1053.

29.5 Managing V8 release environments with the centralized installation manager

Managing the full lifecycle of WebSphere Application Server V8 release environment using the centralized installation manager consists of the following steps:

1. Define your targets.
2. Install Installation Managers on targets.
3. Install the product.
4. Create profiles.
5. Register with job manager.
6. Work with environment.

Note: Not every step is required when working with the centralized installation manager. It depends on your WebSphere Application Server environment and if you are working with existing installations or creating a new environment.

29.5.1 Adding new targets

To register a new target:

1. Start the job manager or deployment manager and the targets. In the web console, click **Jobs** → **Targets** → **New Host**.
2. Supply the form presented in Figure 29-4 on page 1013 with the host name of your target and its operation type. Specify the user that will be used for product installation and management, or select to use a public/private key pair to authenticate to the target. If you select a non-root user, be sure that this account allows you to install the product. Refer to 29.1.4, “Additional requirements” on page 1003 for more information about this topic.
3. You can also select the **Save security information** option to save the credentials together with the host. This action eliminates supplying the credentials each time you submit a job to that host.
4. In the *Installation Manager data location path(s)* field, you can also supply paths on the target host to the Installation Manager installation directory. If there is no Installation Manager on that host, leave this field blank. If the Installation Manager was not installed in the standard location (during installation, a custom path was given), use the custom path in this field. This action helps the centralized installation manager to obtain which

Installation Manager it must use on the target. Path information helps because the inventory job that the centralized installation manager uses to detect the installed WebSphere software, might not be able to locate the Installation Manager product. Figure 29-4 shows the window used to create a new target.

5. Click **OK** to save the target.

Targets > New...

Use this panel to register a new host with the job manager.

General Properties

* Host name
saw211-sys1

Operating system
AIX

* Administrative user with installation authority
root

Target authentication

Password authentication

* Password
.....

* Confirm password
.....

Use sudo

Sudo user name
.....

Sudo password
.....

Confirm sudo password
.....

Public-private key authentication

* Full path to keystore
.....

Passphrase
.....

Confirm passphrase
.....

Save security information

Figure 29-4 New target definition

Complete the following steps to list the available resources discovered by the centralized installation manager on the target:

1. Click **Jobs** → **Targets**.
2. Select the **check boxes** next to your targets, and click the **Display Resources** button.

- In the pop-up menu, select **All** to see all available resources, as shown in Figure 29-5. Note that not all targets contain the version information. Targets that have versions are managed nodes registered in the job manager, such as deployment manager or administrative agent servers, and the version is their product version level. Targets that are just added as new targets do not contain this information.

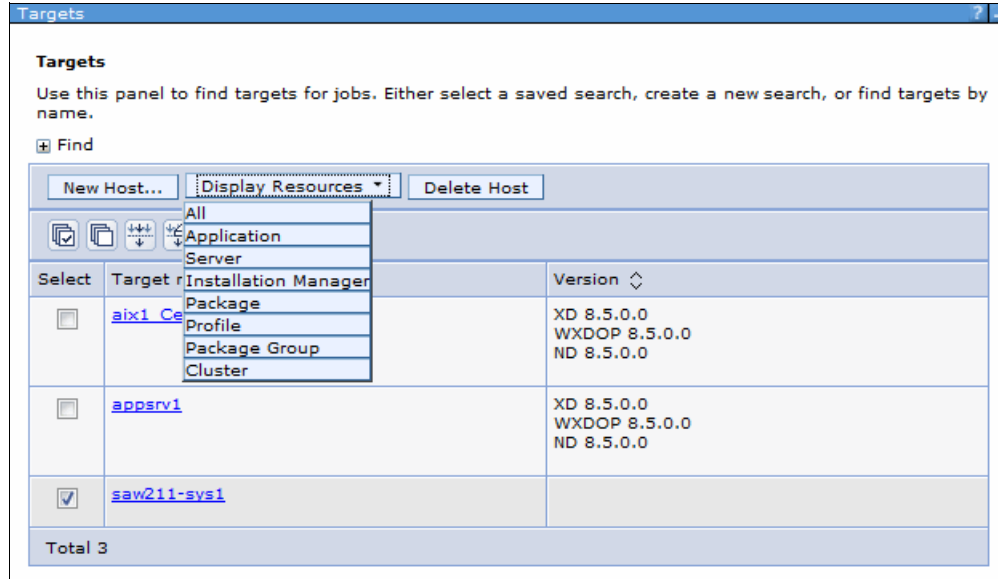


Figure 29-5 Listing resources on targets

Discovered resources are displayed (Figure 29-6 on page 1015). If your target does not contain any WebSphere Application Server or Installation Manager products, the list is empty. You can also filter the resources you want to see by selecting a different **Display Resources** option, for example, limited only to Applications.

| Resources | Quantity | Target name |
|--|----------|-------------|
| InstallationManager/InstallationManager | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment/Profile/AppSrv8_01 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment/Profile/AppSrvX | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment/Profile/Dmqr8_01 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server Network Deployment V8.0/Package/IBM WebSphere Application Server Network Deployment/Profile/V8JobMqr01 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server V8.5 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server V8.5/Package/IBM WebSphere Application Server Network Deployment | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server V8.5/Package/IBM WebSphere Application Server Network Deployment/Profile/AdminAgent01 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server V8.5/Package/IBM WebSphere Application Server Network Deployment/Profile/AppSrv_85_01 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server V8.5/Package/IBM WebSphere Application Server Network Deployment/Profile/JobMqr01 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server V8.5/Package/IBM WebSphere Application Server Network Deployment/Profile/dmqr_85_01 | 1 | saw211-sys1 |
| InstallationManager/InstallationManager/PackageGroup/IBM WebSphere Application Server V8.5/Package/IBM WebSphere SDK Java Technology Edition (Optional) | 1 | saw211-sys1 |
| Total 14 | | |

Figure 29-6 Resources discovered by the centralized installation manager

Note that the target resources include information, such as the package or package group that was used to install a given product with the Installation Manager. In Figure 29-6, there are package groups for both *IBM WebSphere Application Server Network Deployment V8.5* and *IBM WebSphere Application Server Network Deployment V8.0* which were used as the base for WebSphere Application Server resources. You can also see that there are profiles resources derived from the *IBM WebSphere Application Server Network Deployment* packages.

29.5.2 Installing Installation Manager on remote targets

With defined targets, you can install the Installation Manager on those targets. Complete the following steps to prepare your Installation Manager installation kit for use on your targets:

Tip: WebSphere Application Server V8.5 comes with Installation Manager V1.5.2. You can use this version to work with the centralized installation manager, but consider using the newest Installation Manager version available. To download a current version of IBM Installation Manager, go to the following website:

http://www-947.ibm.com/support/entry/portal/All_download_links/Software/Rational/IBM_Installation_Manager

1. Download the compressed Installation Manager copy valid for your target operating system to your local machine.
2. Click **Jobs** → **Installation Manager installation kits**.
3. Optionally, if you want to change the default local directory where the job manager or deployment manager keeps binaries, enter the path under **Installation Manager installation kits location**. By default, it is the `profile_home/IMKits` directory, where `profile_home` is your job manager or deployment manager profile directory.
4. Click **Add**, and point to the Installation Manager compressed file on your local machine. The Installation Manager installation kit is transferred to the job manager or deployment manager and registered in its repository.

Tip: If you do not want to transfer the Installation Manager installation kit using your local computer and HTTP transfer, you can copy this file directly to the Installation Manager installation kit location directory on the deployment manager or job manager.

After you complete these steps, a view similar to Figure 29-7 is displayed.

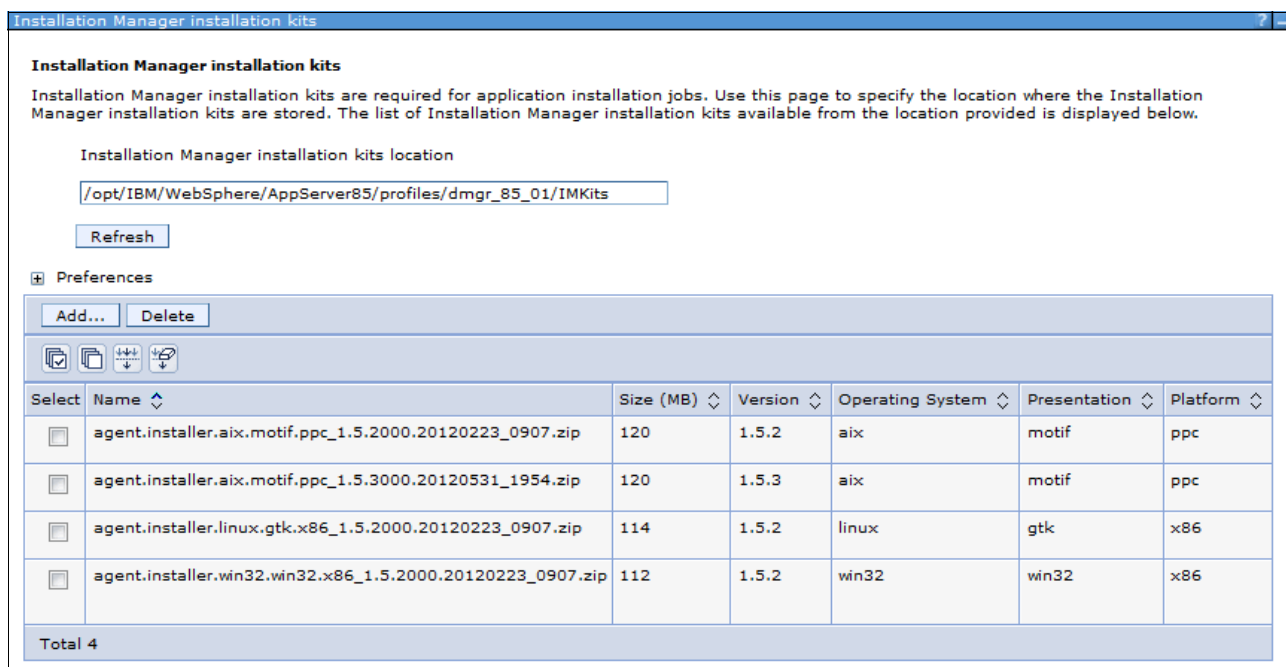


Figure 29-7 Installation Manager installation kit definitions

To delete the Installation Manager installation kit, select it from the list shown in Figure 29-7 on page 1016 and click the **Delete** button. The compressed file with the Installation Manager will be deleted from the server.

You can install additional versions of Installation Manager for other platforms of your choice that are supported by the Installation Manager. Each of them is kept in the Installation Manager installation kit directory.

Note: The centralized installation manager repository location is configured in the CIMJMMetadata.xml file. This file is located in the job manager or deployment manager profile_home/properties/cimjm directory.

Complete the following steps to proceed with the installation of Installation Manager on remote targets:

1. Click **Jobs** → **Submit** from the Jobs navigation section.
2. From the drop-down menu, choose the **Install IBM Installation Manager** job, and click **Next**.
3. Choose the job targets:
 - a. Select a group of targets from the list, or select **Target names**.
 - b. If you selected **Target names**, specify a target name, and click **Add**, or click **Find**, and specify the chosen targets in the Find targets window.
 - c. If user authentication is required, specify a user name, password, or any other authentication values as needed (Figure 29-8).

The screenshot shows a web-based configuration interface for installing IBM Installation Manager. On the left, a vertical sidebar lists five steps: Step 1: Choose a job type, Step 2: Choose job targets (highlighted with a yellow arrow), Step 3: Specify job parameters, Step 4: Schedule the job, and Step 5: Review the summary and submit the job. The main content area is titled 'Choose job targets' and has a sub-header 'Job type: Install IBM Installation Manager'. There are two radio buttons: 'Target groups' (unselected) and 'Target names' (selected). Below 'Target groups' is a dropdown menu with the text '-- No groups --'. Below 'Target names' is a text input field containing 'saw211-sys1', followed by 'Add' and 'Find...' buttons. A 'Remove' button is located at the bottom right of the target list. Below this section is the 'Target authentication' section, which includes a 'User name' field with 'root' entered, a selected 'Password authentication' radio button, and two password fields labeled '* Password' and '* Confirm password', both containing masked characters (dots).

Figure 29-8 Configuring job authentication method for target

- d. Click **Next**.

- Because you specified the Installation Manager installation kit in the previous step, you can leave the The path and file name of Installation Manager kit field empty, as shown in Figure 29-9. Configuring the Installation Manager installation kit before submitting the job is also useful because when you install the Installation Manager on a group of targets (consisting of different platforms), the Installation Manager automatically uses an appropriate Installation Manager installation kit for each target. The automatic function does not require that the Installation Manager has a valid copy for the given platform, so providing a static directory only works for single targets.

Figure 29-9 Installing the Installation Manager on the target

You can also leave the Installation Manager agent data location and Installation Manager installation directory empty if you want to use the default Installation Manager install directory.

To proceed to the next step, accept the terms in the license agreements by selecting **I accept the terms in the license agreements**. To review the license, extract the Installation Manager installation kit and from the extraction point location run the `install.exe` command for Windows systems or `install` command for AIX, Linux or Solaris operating systems. You can also run `installc -c` to review the license in text mode.

Click **Next** to continue.

- If you want to schedule this job to run at a specified time, you can use this step to define specific dates and times. You can also specify the job to run on a given basis, such as daily or weekly. To just install the product, leave the form with its defaults with Availability interval selected to **Run once**, and click **Next**.

Figure 29-10 on page 1019 shows the available configuration options of this step.

Submit a job to the job manager

Schedule the job by specifying when the job available, when the job expires, if the job is to rerun after a period of time, and what email address is to receive notification when the job is done. Jobs are scheduled for availability and expiration relative to the time of the machine on which the job manager resides.

Step 1: Choose a job type

Step 2: Choose job targets

Step 3: Specify job parameters

→ Step 4: Schedule the job

Step 5: Review the summary and submit the job

Schedule the job

Job type: Install IBM Installation Manager

Notification

Email addresses

Initial Availability

Specify when this job is first available.

Make the job available now.

Schedule availability

Date (MM/dd/yyyy) Time (HH:mm:ss)
 / / : :

Expiration

Specify when this job is no longer available.

Use default expiration - 1 days.

Expire the job based on a date

Date (MM/dd/yyyy) Time (HH:mm:ss)
 / / : :

Expire the job based on a duration

Expire after

Job Availability Interval

Jobs can run repeatedly based on an interval. Specify the interval that the job is available.

Availability interval

Figure 29-10 Scheduling a job

6. Review the summary and then click **Finish** to submit the job.

After you submit the job, a unique identifier is allocated to it to track the job status. The administrator can always return to the **Jobs** → **Status** view to track the job progress (Figure 29-11).

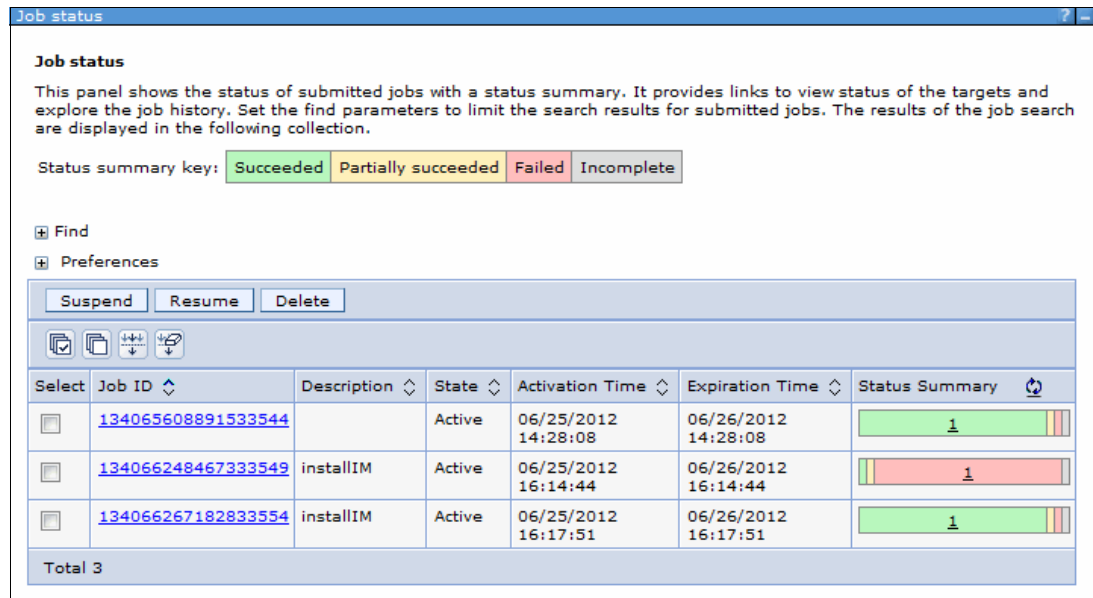


Figure 29-11 Monitoring jobs status

If the job is successful or it fails, additional messages and files might be shown (Figure 29-12). If files are present, you can click them to see the full path to the file on the target. This path can be used to locate the file and obtain it using a Collect file job. In case of an error, investigate the error message or file, correct the condition that caused the error, and submit the job again.

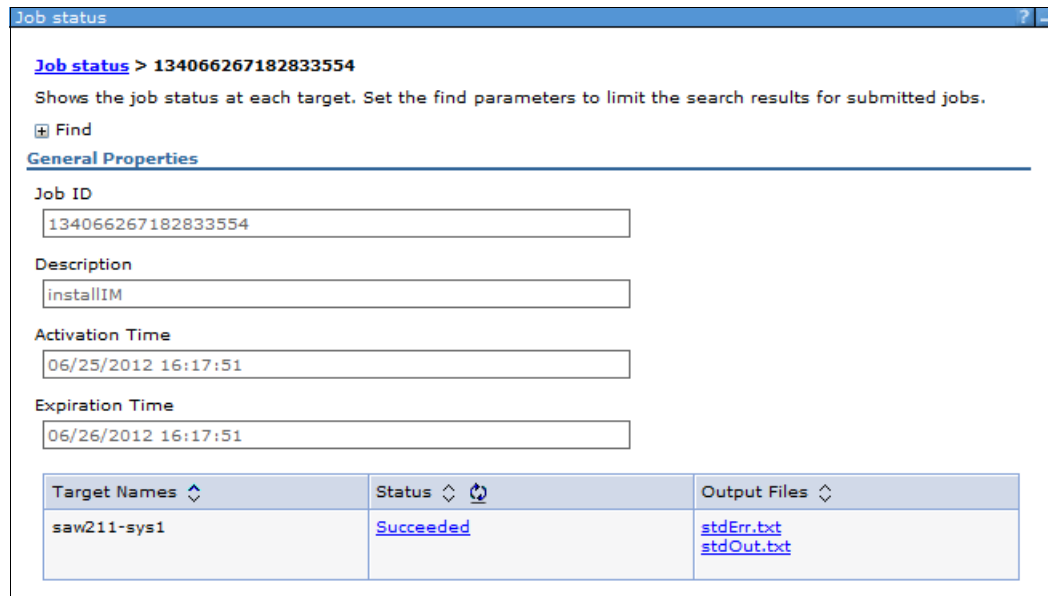


Figure 29-12 Additional jobs messages located in files accessible from the console

Updating the Installation Manager on remote targets

To update the Installation Manager, select the **Update IBM Installation Manager** job.

For additional information about this procedure, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tagt_jobmgr_update_im.html

Uninstalling the Installation Manager on remote targets

Select **Uninstall IBM Installation Manager** to proceed with this procedure. Note that to uninstall the Installation Manager, all products installed using it must be uninstalled first. For additional information about this topic, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tagt_jobmgr_uninstall_im.html

29.5.3 Installing a Secure Shell (SSH) public key

Installing a secure shell public key is not a required step, but can be done using the job manager or the deployment manager console.

Complete the following steps to install a secure shell public key:

1. Click **Jobs** → **Submit** from the navigation tree of the administrative console.
2. Select the **Install SSH Public Key** job and click **Next**.
3. Select the job targets, and supply the administrative user name and password. Click **Next**.
4. On the Specify the job parameters window, specify the location of the public key file that you want to install on the selected target, and click **Next** (Figure 29-13).

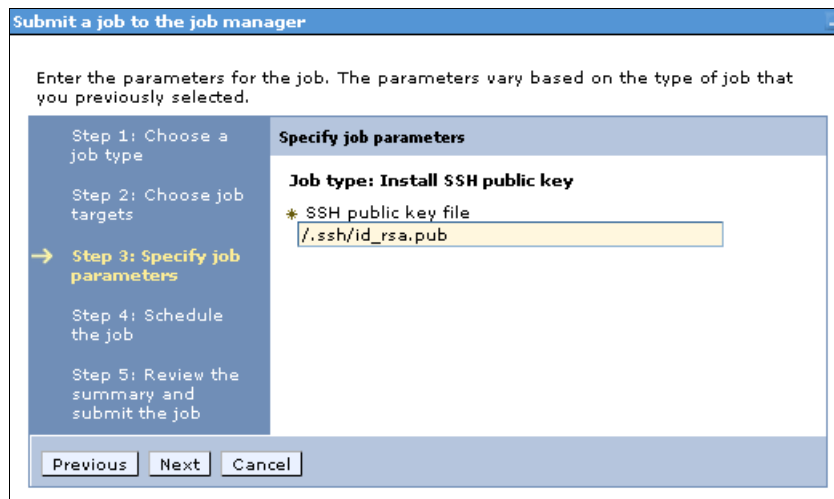


Figure 29-13 Installing a public key on target

To learn more about how to obtain the keys, refer to 29.1, “The centralized installation manager prerequisites” on page 1002.

5. Schedule the job, and click **Next**.
6. Review the summary, and click **Finish** to submit the job.

If the job is successful, you can use the public/private key pair method to authenticate from the job manager or network manager host to the target host.

29.5.4 Installing WebSphere Application Server binaries on remote hosts

With the targets defined and the Installation Manager installed on the targets, complete the following steps to install the WebSphere Application Server binaries on remote hosts:

1. Consider running an inventory job to refresh the job manager or deployment manager targets repository:
 - a. In the administrative console, click **Job** → **Submit**.
 - b. In the Job type menu list, select the **Inventory** job, and click **Next**.
 - c. Specify the target names and target authentication, and click **Next**.
 - d. Schedule and submit the job.
2. Use the Manage offerings job to install the product:
 - a. In the administrative console, click **Job** → **Submit**.
 - b. In the Job type menu list, select the **Manage offerings** job, and click **Next**.
 - c. Specify the target names and target authentication, and click **Next**.
 - d. Specify the required parameter:

Response file path name The full path name to the Installation Manager response file. The path must point to a file located on job manager or deployment manager machine.

Tip: You can obtain the Installation Manager response file using the **IBMIM.exe** command for Windows or **IBMIM** command for UNIX, executed from installation_manager_home/eclipse directory:

```
IBMIM -record <path_to_your_file>.xml -skipInstall  
<path_to_empty_directory>
```

Example 29-3 shows a sample response file that can be used for WebSphere Application Server V8.5 product installation. Note that you must edit the highlighted properties to tell Installation Manager which repository to use and where to install the product on the target. There are many more ways to edit this file to instruct Installation Manager. For more information about this topic, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/tagt_job_install_was.html

Example 29-3 Response file used to install WebSphere Application Server V8.5 product

```
<?xml version="1.0" encoding="UTF-8"?>  
<!--The "acceptLicense" attribute has been deprecated. Use "-acceptLicense"  
command line option to accept license agreements.-->  
<agent-input acceptLicense='true'>  
<server>  
<repository location='<MY REPOSITORY LOCATION>' />  
<repository  
location='https://www.ibm.com/software/repositorymanager/service/com.ibm.web  
sphere.ND.v85/8.5.0.0' />  
</server>  
<profile id='IBM WebSphere Application Server V8.5'  
installLocation='<LOCATION TO INSTALL PRODUCT ON TARGET MACHINE>'  
<data key='eclipseLocation' value='<LOCATION TO INSTALL PRODUCT ON TARGET  
MACHINE>' />  
<data key='user.import.profile' value='false' />
```

```

<data key='cic.selector.os' value='win32' />
<data key='cic.selector.ws' value='win32' />
<data key='cic.selector.arch' value='x86' />
<data key='cic.selector.nl' value='en' />
</profile>
<install modify='false'>
<offering id='8.5.0.0-WS-WAS-IFPM62795' version='8.5.0.20120503_1150'
profile='IBM WebSphere Application Server V8.5' features='-'/>
<offering id='8.5.0.0-WS-WAS-IFPM63690' version='8.5.0.20120611_1318'
profile='IBM WebSphere Application Server V8.5' features='-'/>
<offering id='8.5.0.0-WS-WAS-IFPM63827' version='8.5.0.20120503_1507'
profile='IBM WebSphere Application Server V8.5' features='-'/>
<offering id='8.5.0.0-WS-WASProd-IFPM64186' version='8.5.0.20120611_1543'
profile='IBM WebSphere Application Server V8.5' features='-'/>
<offering id='8.5.0.0-WS-WASProd-MultiOS-IFPM63479'
version='8.5.0.20120613_2056' profile='IBM WebSphere Application Server
V8.5' features='-'/>
<offering id='com.ibm.websphere.ND.v85' version='8.5.0.20120501_1108'
profile='IBM WebSphere Application Server V8.5'
features='core.feature,ejbdeploy,thinclient,embeddablecontainer,com.ibm.sdk.
6_64bit,samples,liberty' installFixes='none' />
</install>
<preference name='com.ibm.cic.common.core.preferences.eclipseCache'
value='<LOCATION TO IBM IMShared FOLDER ON TARGET MACHINE>' />
<preference name='com.ibm.cic.common.core.preferences.connectTimeout'
value='30' />
<preference name='com.ibm.cic.common.core.preferences.readTimeout'
value='45' />
<preference
name='com.ibm.cic.common.core.preferences.downloadAutoRetryCount'
value='0' />
<preference name='offering.service.repositories.areUsed' value='true' />
<preference name='com.ibm.cic.common.core.preferences.ssl.nonsecureMode'
value='false' />
<preference
name='com.ibm.cic.common.core.preferences.http.disablePreemptiveAuthenticati
on' value='false' />
<preference name='http.ntlm.auth.kind' value='NTLM' />
<preference name='http.ntlm.auth.enableIntegrated.win32' value='true' />
<preference
name='com.ibm.cic.common.core.preferences.preserveDownloadedArtifacts'
value='true' />
<preference name='com.ibm.cic.common.core.preferences.keepFetchedFiles'
value='false' />
<preference name='PassportAdvantageIsEnabled' value='false' />
<preference name='com.ibm.cic.common.core.preferences.searchForUpdates'
value='true' />
<preference name='com.ibm.cic.agent.ui.displayInternalVersion'
value='true' />
<preference name='com.ibm.cic.common.sharedUI.showErrorLog' value='true' />
<preference name='com.ibm.cic.common.sharedUI.showWarningLog' value='true' />
<preference name='com.ibm.cic.common.sharedUI.showNoteLog' value='true' />
</agent-input>

```

- e. Specify optional parameters:

IBM Installation Manager Path

Specify the path to install Installation Manager on the remote machine. If this parameter is blank, then Installation Manager is considered to be installed in the default location.

IBM Installation Manager agent data location

Specify an IBM Installation Manager data location that is not the default location for the manageOfferings job.

IBM Installation Manager key ring file

If the package repository requires a key ring file for authentication, specify the full path name of the key ring file on the job manager machine.

Key ring file password

If the key ring file is password protected, specify the key ring password.

Tip: To avoid trouble, if you installed the Installation Manager using defaults, use defaults during this task. Do not use a non-default data location unless you are familiar with IBM Installation Manager.

- f. Select **I accept the terms in the license agreements**, and click **Next**.
- g. Schedule the job, or click **Next** to proceed with the next step.
- h. Review the summary of the job, and click **Finish** to submit it.
3. At this point, your job is submitted and has a unique identifier. You can track its status by clicking **Jobs** → **Status**.

To learn more about installing the WebSphere Application Server product using jobs, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tagt_job_install_was_gui.html

29.5.5 Creating a WebSphere Application Server profile on a remote target

In this example, two profiles are created: a deployment manager profile and a default profile, which is federated into the deployment manager cell.

To create profiles with centralized installation manager, you need response files. Consider preparing your own response files on your local, experimental environment. Two response files that we use in this section are shown in Example 29-4 and in Example 29-5 on page 1025.

Example 29-4 Response file for a Deployment Manager profile

```
create
profileName=Dmgr85_03
profilePath=/opt/IBM/WebSphere/AppServer85/profiles/Dmgr85_03
templatePath=/opt/IBM/WebSphere/AppServer85/profileTemplates/cell/dmgr
nodeName=aix-target2CellManager01
```



```
cellName=aix-target2Cell01
hostName=saw211-sys1
adminUserName=admin85
adminPassword=admin85
enableAdminSecurity=true
samplesPassword=admin
appServerNodeName=aix-target2Node01
nodeProfilePath=/opt/IBM/WebSphere/AppServer85/profiles
```

Example 29-5 Response file for a default server profile federated to the deployment manager

```
create
profileName=AppSrv_v85_01
profilePath=/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_v85_01
templatePath=/opt/IBM/WebSphere/AppServer85/profileTemplates/cell/default
nodeName=aix-target2CellManager01
cellName=aix-target2Cell01
hostName=saw211-sys1
enableAdminSecurity=true
adminUserName=admin85
adminPassword=admin85
samplesPassword=admin
appServerNodeName=aix-target2Node01
dmgrProfilePath=/opt/IBM/WebSphere/AppServer85/profiles/Dmgr85_03
nodePortsFile=/opt/IBM/WebSphere/AppServer85/profiles/Dmgr85_03/properties/nodeportdef.props
portsFile=/opt/IBM/WebSphere/AppServer85/profiles/Dmgr85_03/properties/portdef.props
```

Complete the following steps to use the centralized installation manager to create the deployment manager profile and a default profile:

1. Click **Jobs** → **Submit**.
2. From the drop-down menu, select **Manage profiles**.
3. Select your targets and authentication method, and click **Next**.
4. Specify the installation home directory of WebSphere Application Server product. Specify the path from Example 29-3 on page 1022, where it was defined during the base product installation.

Specify the path to the response file that will be used to create the profile. In this case, copy the response file from Example 29-4 on page 1024 to the job manager or deployment manager machine, and type the path to that file (Figure 29-14 on page 1026). Click **Next**.

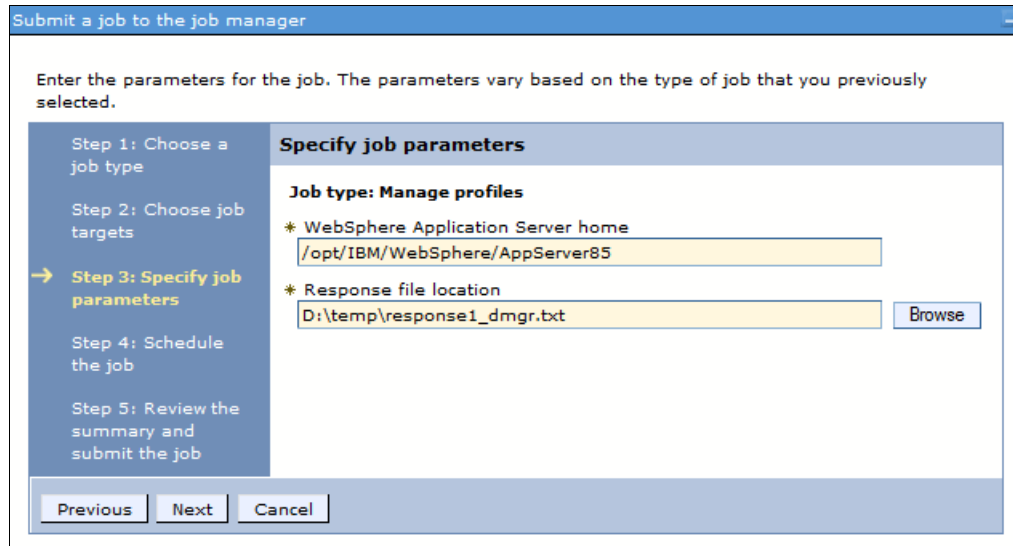


Figure 29-14 Creation of profile using the Manage profiles job

5. Schedule the job, or submit it to run once, and click **Next**.
6. Review the job details, and submit it by clicking **Finish**, or change its parameters by clicking **Previous**.

If the job completes successfully, proceed with creating the next profile. If any errors occur during installation, inspect them, correct them, and run the manage profiles job again.

To create a second profile, complete steps 1 on page 1025 through 6 again, but in step 4, enter the path to the response file shown in Example 29-5 on page 1025.

When the second job completes successfully, the profiles are ready, but not started. To use them, the administrator has to start the deployment manager and the node agent of the server profile. You can either log into the targets and invoke the `./startManager` and `./startNode` commands or use the job manager or deployment manager administrative console.

Complete the following steps to use the job manager or deployment manager administrative console to start the deployment manager and node agent:

1. Click **Jobs** → **Submit**.
2. Select **Run command job on remote host** from the drop-down menu.
3. Add your target on which you installed the profiles.
4. Specify the user name that you used to install the product, supply the password or use the public/private key authentication method, and click **Next**.
5. Specify the script name you want to run in the Command or script field, and run `startManager.sh` to start the deployment manager or `startNode.sh` to start the node agent.

In the field under Working directory, specify the path to your command. If you want to start the deployment manager server, use:

```
/opt/IBM/WebSphere/AppServer85/profiles/Dmgr85_03/bin
```

You can use your own directory where you installed the deployment manager profiles. If you want to start the server profile, use:

```
/opt/IBM/WebSphere/AppServer85/profiles/AppSrv_v85_01/bin
```

Alternately, you can use your own directory where you installed the federated server profile (Figure 29-15).

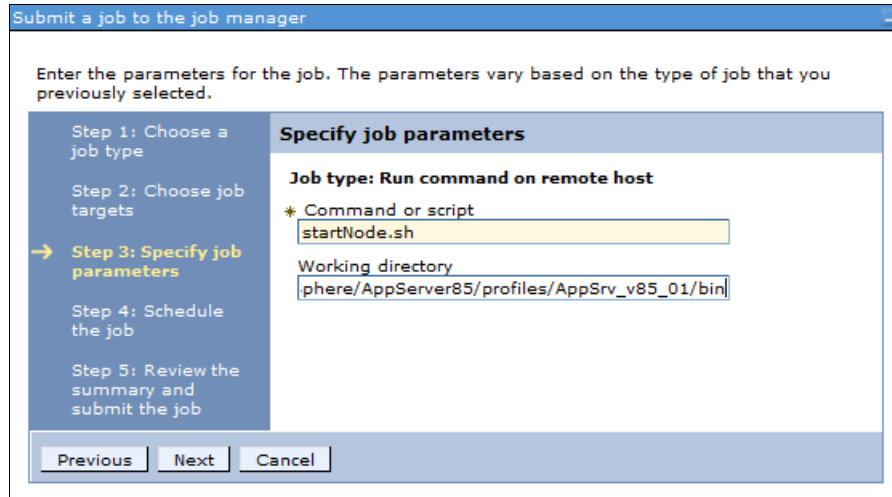


Figure 29-15 Running a remote script using job manager

6. Schedule your job, and click **Next**.
7. Review the job details, and submit the job by clicking **OK**.

29.5.6 Registering and unregistering the profile with the Job Manager

To fully control target WebSphere Application Server profiles, you have to register the newly created deployment manager.

Note: Apart from deployment manager servers, you can also register administrative agent type servers to manage even more complex WebSphere Application Server environments.

Complete the following steps to register deployment manager:

1. In the target deployment manager administrative console, click **System administration** → **Deployment manager**.
2. Click the **Job managers** link under the Additional Properties section.
3. Click **Register with Job Manager**.

4. Supply the managed node name for your job manager profile (Figure 29-16).

Specify the job manager host name, secure http port, and the alias name to be used in the job manager. Specify the user name and password of the job manager user.

The screenshot shows a dialog box titled "Deployment manager" with a breadcrumb path: "Deployment manager > Job managers > Register with Job Manager". Below the breadcrumb is a paragraph of instructions: "Register a managed node with job manager. If you are using the administrative agent administrative console, you register a node that is already registered to the administrative agent. If you are using the deployment manager administrative console, you register the deployment manager. Specify an alias if the managed node name is in use by another node. Use the administrative host port of the job manager, which defaults to 9943 when security is enabled." Below this is a section titled "General Properties" with a horizontal line. The form contains the following fields: "Managed node name" (value: aix1_CellManager_85_01), "Alias" (value: WAS_v85_aix1), "Host name" (value: saw211-sys1), "Port" (value: 9944), "User name" (value: admin85j), "Password" (masked with 7 dots), and "Confirm password" (masked with 7 dots). At the bottom are three buttons: "OK", "Reset", and "Cancel".

Figure 29-16 Registering the deployment manager in the job manager

Complete the following steps to view the job managers from the deployment manager console:

1. Click **System administration** → **Deployment manager**.
2. Click **Job managers** under the Additional Properties section.

A list similar to Figure 29-17 is displayed that lists every job manager instance in which your deployment manager server is registered.

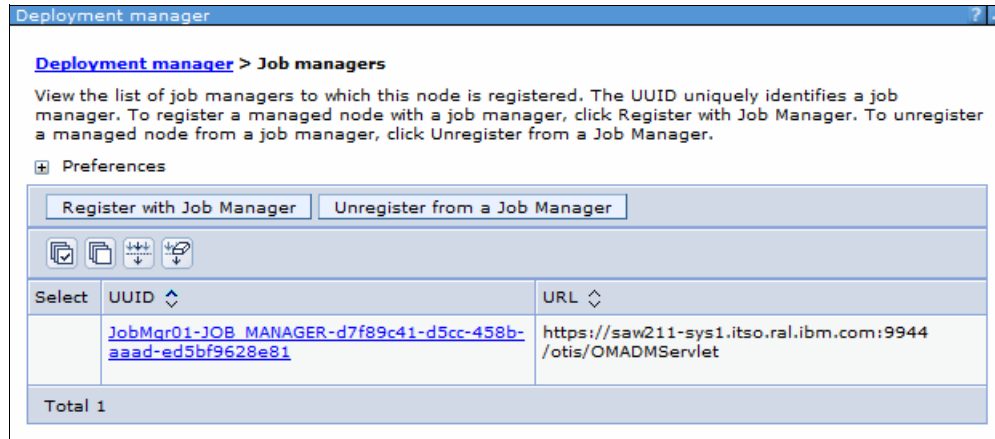


Figure 29-17 Registered deployment manager ID

Complete the following steps to unregister your deployment manager from the Job Manager:

1. Click **System administration** → **Deployment manager**.
2. Click the **Job managers** under the Additional Properties section.
3. Click **Unregister from a Job Manager**.
4. Supply the form from Figure 29-16 on page 1028 with all of the information about your job manager profile. WebSphere Application Server uses this information to unregister your deployment manager.

Note that although only the Managed node name is the required field, you have to specify all information about your job manager to unregister it.

If some information is missing or is not correct, an error message similar to the one shown in Figure 29-18 is displayed.

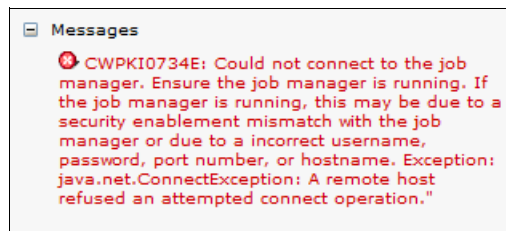


Figure 29-18 Error message when trying to unregister a host providing wrong data

29.5.7 Working with remote targets

In this section, the job manager is used to deploy applications to the target. We assume that you followed the previous steps and your environment is prepared to deploy applications.

You can download the sample applications from the WebSphere Application Server samples website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.samples.doc/ae/welcome_samples.html

A JavaServer Faces 2.0 JSFSample example is used in this chapter.

Complete the following steps to install a new application on a remote target:

1. Copy the application ear file (JSFSample.ear) into the job manager directory `profile_home/config/temp/JobManager`.
2. Log on to the job manager console.
3. Click **Jobs** → **Submit**.
4. From the Job type box, select the **Distribute file** job, and click **Next**.
5. Select your target and the authentication method you want to use against the operating system, and click **Next**.
6. Specify the source as JSFSample.ear. It is a relative path to the job manager directory from step 1.

Specify the destination where the file will be uploaded. In this case, use the `/opt/IBM/WebSphere/AppServer/profiles/Dmgr8_01/downloadedContent` path. Click **Next**.

Note: Your user must have access to the directory where you upload the application.

Figure 29-19 illustrates step 6.

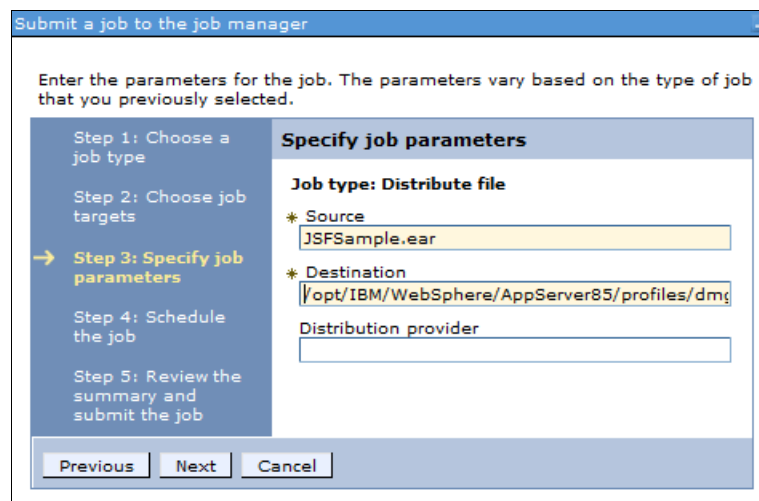


Figure 29-19 Copying the application to the target using a job

7. Schedule the job, and click **Next**.
8. Review the job details, and click **Finish** to submit it.

When the job finishes successfully, the file is transferred to the directory of your choice on the target. Complete the following steps to install it remotely on the WebSphere Application Server target:

1. Click **Jobs** → **Submit**.
2. From the Job type box, select the **Install application** job, and click **Next**.
3. Select your target and the authentication method you want to use against the deployment manager.
4. Specify the job parameters. Enter the application file name. For example, if you copied the application to the `dmgr_profile_home/downloadedContent` directory on a remote deployment manager, you can simply enter the application name (JSFSample.ear) in the Application location.

To specify which server the application is deployed to, click the **Find** button next to the Server name. Select the desired server instance, and click **OK**. See Figure 29-20 for details.

Find targets > Find Target Resources

Set the find parameters to limit the search results for target resources. The results of the search are displayed in the following collection. Select the resource that you want to use in the job.

Find

Type: Server

Resource name: =

Status: =

Server type: =

Target name: = aix1_CellManager_85_01

Group name: =

Context: =

Maximum results: 50

Available resources common to all selected targets

- node/aix1_CellManager_85_01/server/dmgr
- node/saw211-sys1Node01/server/appsrv85_01**
- node/saw211-sys1Node01/server/appsrv85_02
- node/saw211-sys1Node01/server/nodeagent

Figure 29-20 Specifying a target server for an application

After you specify the server instance, the Node name field is automatically set with the correct server node name. Your window at this point looks similar to Figure 29-21.

Submit a job to the job manager

Enter the parameters for the job. The parameters vary based on the type of job that you previously selected.

Step 1: Choose a job type

Step 2: Choose job targets

→ Step 3: Specify job parameters

Step 4: Schedule the job

Step 5: Review the summary and submit the job

Specify job parameters

Job type: Install application

* Application name
JSFSample

Application location
JSFSample.ear

Server name
appsrv85_01 Find...

Node name
saw211-sys1Node01

Cluster name
Find...

Previous Next Cancel

Figure 29-21 Deploying an application using the job manager

If your topology uses application clusters, specify the cluster name on which the application will be deployed in the Cluster name field. In this case, we run it only on a single server, so you can omit this field, and click **Next**.

5. Schedule the job, and click **Next**.
6. Review the job information, and click **Finish** to submit it.

When the job finishes successfully, the application is deployed on the remote target, but it is stopped.

If the target server is stopped, you need to start it. Complete the following steps to start the server using job manager:

1. Click **Jobs** → **Submit**.
2. From the Job type box, select the **Start server** job, and click **Next**.
3. Select the target and authentication method you want to use against the deployment manager, and click **Next**.
4. Specify the Server name you want to start, and use the **Find** button to select it from the list shown in Figure 29-20 on page 1031. After you select the target server, the Node name field is automatically set with the correct value. Click **Next**.
5. Schedule the job, and click **Next**.
6. Review the job information, and click **Finish** to submit it.

If the target server instance is available, you can start the JSFSample application using the Start application job by completing the following steps:

1. Click **Jobs** → **Submit**.
2. From the Job type box, select the **Start application** job, and click **Next**.
3. Select the target and authentication method you want to use against the deployment manager, and click **Next**.

- Specify the application name to start (Figure 29-22). Note that to use this job the application must already be deployed on the server. Use the **Find** button to select it, and click **Next**.

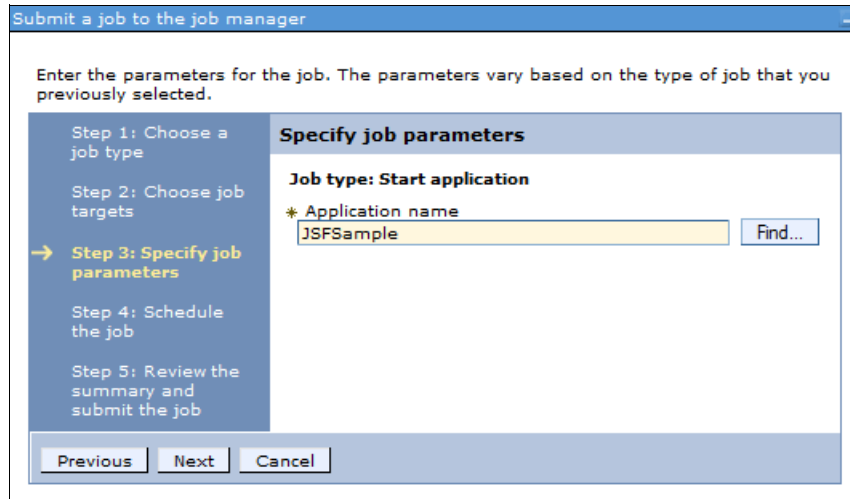


Figure 29-22 Starting the application on the target

- Schedule the job, and click **Next**.
- Review the job information, and click **Finish** to submit it.

When the job finishes successfully, you can invoke the application on the remote target. If you used the JSFSSample application, use the following URL to access the application:

`http://<your target host>:<your port>/SampleTemplating/home.faces`

You will see a window similar to the one shown in Figure 29-23 on page 1034.

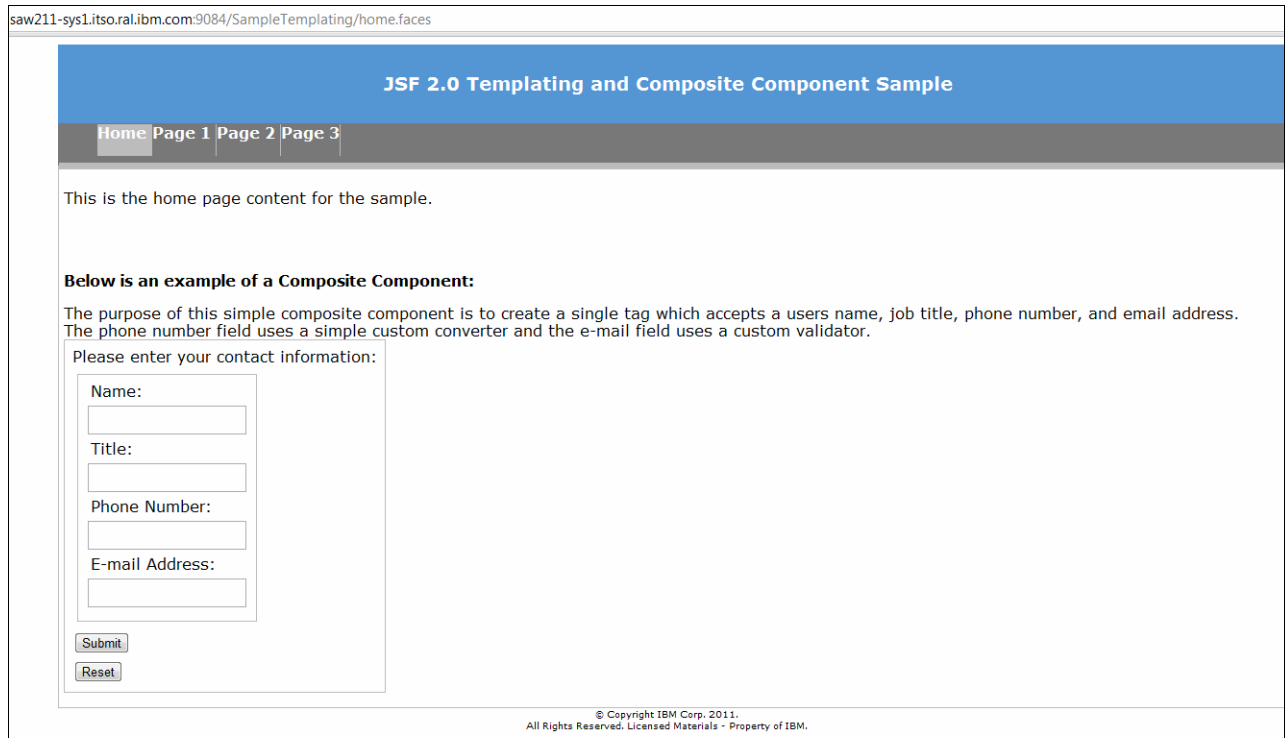


Figure 29-23 JSFSample application output

29.5.8 Installing maintenance to remote targets

To install maintenance packages on remote targets, use the Manage offerings job with the appropriate response file. This procedure is the same for WebSphere Application Server V8 releases. Before installation, make sure that all Java processes on the target runtime are stopped.

1. From the list, select the **manageOfferings** job to install maintenance:
 - a. In the administrative console, click **Job** → **Submit**.
 - b. In the Job type menu list, select the **Manage offerings** job, and click **Next**.
 - c. Specify the target names and target authentication, and click **Next**.
 - d. Specify this required parameter:
 - Response file path name: The full path name to the Installation Manager response file in which you specify the maintenance to install, repository, and target WebSphere Application Server runtime. The path must point to a file located on the job manager or the deployment manager machine. You can prepare your response file using Installation Manager parameters shown in example in 29.5.4, “Installing WebSphere Application Server binaries on remote hosts” on page 1022.
 - e. Specify these optional parameters:
 - IBM Installation Manager Path: Specify the path to install Installation Manager on the remote machine. If this parameter is blank, Installation Manager is considered to be installed in the default location.
 - IBM Installation Manager agent data location: Specify an IBM Installation Manager data location that is not the default location for the manageOfferings job.

- IBM Installation Manager key ring file: If the package repository requires a key ring file for authentication, specify the full path name of the key ring file on the job manager machine.
 - Key ring file password: If the key ring file is password protected, specify the key ring password, and confirm it.
- f. Select **I accept the terms in the license agreements**, and click **Next**.
 - g. Schedule the job, or click **Next** to proceed with the next step.
 - h. Review the summary of the job, and click **Finish** to submit it.

At this point, the job is submitted and has a unique identifier. You can track its status under the **Jobs** → **Status**.

After a successful installation, you will see a message in the stdOut.txt log similar to the following example:

```
Updated to com.ibm.websphere.ND.v80_8.0.1.20110620_2048 in the
/opt/IBM/WebSphere/AppServer directory
```

The change is also reflected in the console of the target. Log on to the updated server to see the new maintenance level. The current version is displayed on the welcome window, as shown in Figure 29-24.

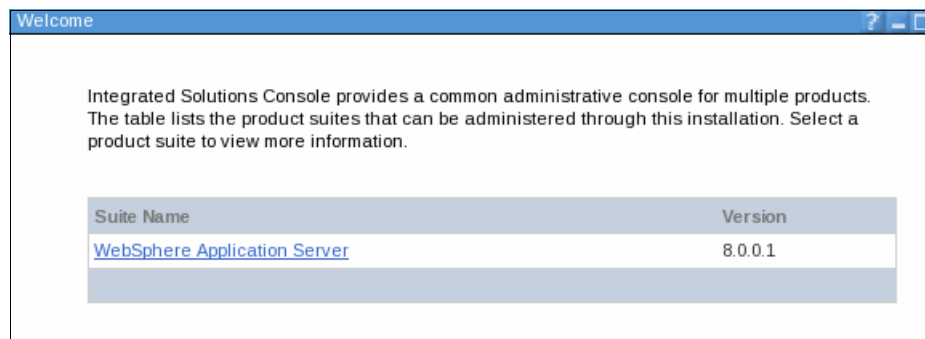


Figure 29-24 WebSphere Application Server binaries updated with Fix Pack 8.0.0.1

For more information about installing maintenance for WebSphere Application Server V8, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tagt_job_install_was_gui.html

For more information about installing maintenance for WebSphere Application Server V8.5, refer to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tagt_job_install_was_gui.html

29.5.9 Using the centralized installation manager with a command line

Every centralized installation manager operation that can be run from the job manager or deployment manager can also be run from the command-line interface. This interface allows you to create an even more automated WebSphere Application Server environment.

The following list of jobs can be used to work with your environment:

distribute
FilecollectFile
removeFile
findDataLocation
installIM
uninstallIM
updateIM
installSSHPublicKey
inventory
manageOfferings
manageprofiles
runCommand
testConnection
installApplication
startApplication
stopApplication
updateApplication
uninstallApplication
createApplicationServer
deleteApplicationServer
createProxyServer
deleteProxyServer
createCluster
deleteCluster
createClusterMember
deleteClusterMember
configureProperties
startServer
stopServer
startCluster
stopCluster
installLibertyProfileResources
uninstallLibertyProfileResources
startLibertyProfileServer
stopLibertyProfileServer
generateMergedPluginConfigForLibertyProfileServers

To discover more about each job type, see the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multipletform.doc/ae/rxml_7jobtypes.html#rxml_7jobtypes__scr5

29.6 Managing V6.1 and V7 with the centralized installation manager

To use the centralized installation manager in WebSphere Application Server V8.5 with previous Version 6.1 or Version 7 products, the following additional steps must be completed:

1. Install the IBM Installation Factory.
2. Configure the centralized installation manager repository for WebSphere Application Server V6.1 and V7 products.
3. Install additional product packages into the repository.

4. Create additional targets.
5. Work with the environment.

Note: The centralized installation manager for WebSphere Application Server V6.1 and V7 has limited functionality compared to V8 releases. Refer to 29.2, “Planning considerations” on page 1004 for more details.

29.6.1 Installing the IBM Installation Factory

Use Installation Factory V7.0.0.15 or newer versions. You can download the current version from the IBM Installation Factory for WebSphere Application Server at the following website:

<http://www-01.ibm.com/support/docview.wss?rs=180&uid=swg24020213>

After it is downloaded, extract the binaries to any directory for which you have the appropriate permissions. You can also install Installation Factory on your system using the code from the product media. Copy the Installation Factory on to your operating system. Use the **setupif** command provided on the Installation Factory disc, or as part of the package download:

- ▶ UNIX: Run the **setupif.sh** command or **setupif.sh target_location**.
- ▶ Windows: Run the **setupif.bat** command or **setupif.bat target_location**.

This command copies the Installation Factory to *user_home/InstallationFactory* by default. You can specify the target location using the target location parameter.

29.6.2 Creating the centralized installation manager repository

Before you can populate the centralized installation repository, ensure that you have write access to the directories you plan to use by completing the following steps:

1. Download the product images, and expand the file (.tar or .zip) to a temporary directory, or ensure access to the product CD.
2. Use the **ifcli.bat** command for Windows or **ifcli.sh** for UNIX to populate the repository. The default folder for the centralized installation manager repository is *<was_install_root>/cimrepos*. Example 29-6 shows the command to set up the centralized installation manager repository in the */opt/IBM/CIMRepo* directory.

Example 29-6 Command for creating the central installation manager repository for WebSphere Application Server V6.1 and V7

```
installation_factory_install_root/bin/ifcli.sh -wasPath
/opt/IBM/WebSphere/AppServer85 -repositoryPath /opt/IBM/CIMRepo
-installationPackagePath /tmp/installs/WAS70ND
```

The centralized installation manager is configured for the WebSphere Application Server product installed under */opt/IBM/WebSphere/AppServer* location. It is also populated with the WebSphere Application Server V7 binaries that are downloaded to the */tmp/installs/WAS70ND* directory.

You can also use GUI-based tools to create repositories. To learn more about using Installation Factory, go to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tins_cim_package_add_if.html

Successful creation of the repository results in the message shown in Example 29-7.

Example 29-7 Listing a successful repository creation

```
*****
The ifcli command started at Jun 26, 2012 5:44:57 PM with options: '-wasPath
/opt/IBM/WebSphere/AppServer85 -repositoryPath /opt/IBM/CIMRepo
-installationPackagePath /tmp/installs/WAS70ND '.
*****
The specified installation was successfully configured to associate with the
repository. Centralized Installation Manager must be restarted in order to use the
repository.
Adding installation package to the repository...
The installation package was successfully added to the repository.
*****
The ifcli command ended at Jun 26, 2012 5:47:31 PM with return code:
INSTCONFSUCCESS.
*****
```

You can review the structure of the repository in the /opt/IBM/CIMRepo directory. Refer to 29.4.2, “The centralized installation manager repository structure” on page 1008 for more information about the repository structure.

29.6.3 Adding packages when deployment manager is connected to the Internet

Complete the following steps to download the latest version of the Update Installer to your centralized installation manager repository. Update Installer is required to apply maintenance on the target systems.

1. Click **System Administration** → **Centralized Installation Manager** → **Installation Packages** → **Update Installer for WebSphere Application Server**.
2. Select one or more operating systems that you will use and then click **Download**. See Figure 29-25 on page 1039.



Figure 29-25 Downloading packages for centralized installation manager from the Internet

3. Review the summary, and select **Download** to start downloading the packages.
4. On the Installation packages window, after the download starts, you can monitor the status by clicking **Refresh**. If you receive errors, refer to the following file for more detailed information:

```
<dmgr_profile_home>/logs/dmgr/systemOut.log
```

Downloading descriptors

The centralized installation manager also supports the installation of Network Deployment V6.1 and V7 Fix Packs on remote nodes that are within the network deployment cell. This configuration is known as a mixed-version cell, where the deployment manager node is at Version 7 or higher and the other nodes within the cell are either at the same level as the deployment manager node or at the Version 6.1 level. The centralized installation manager does not support maintenance levels prior to Version 6.1.

Download additional installation packages and maintenance files to your centralized installation manager repository by clicking **System Administration** → **Centralized Installation Manager** → **Installation Packages** → **add package**. Select one or more descriptor files (Figure 29-26 on page 1040), and click **Download** to proceed.

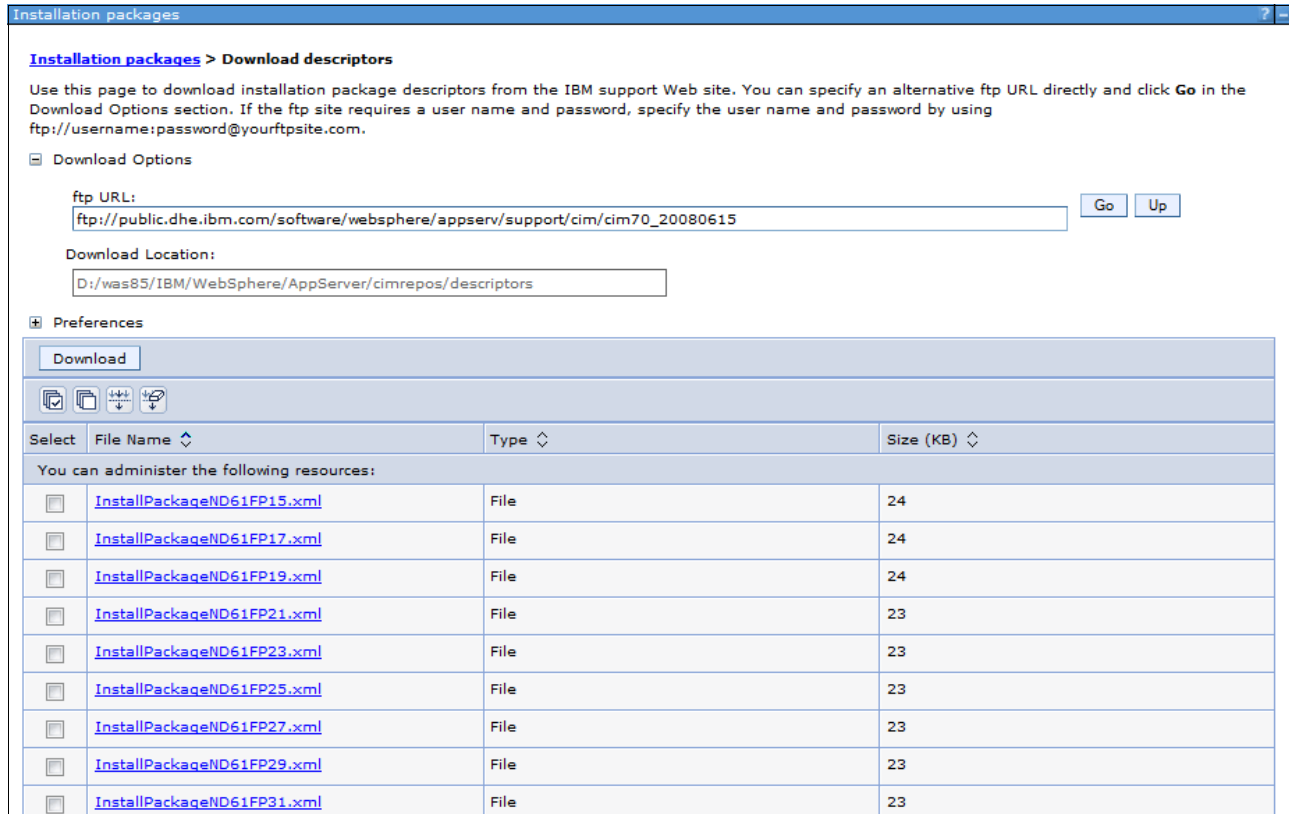


Figure 29-26 Downloading maintenance descriptors to the centralized installation manager repository

You can monitor the progress of the download by selecting **Download Status**.

Downloading fix pack binaries

After the descriptor for the required Network Deployment V6.1 Fix Pack has been downloaded using the method described here, download the *.pak files for that fix pack to the centralized installation manager repository.

The centralized installation manager uses the Update Installer for WebSphere Application Server V7 to install and uninstall WebSphere Application Server Network Deployment V6.1 and V7 Fix Packs.

To download the binary files for a refresh pack, fix pack, or maintenance tool package type, which includes the Update Installer, complete the following steps:

1. Click **System Administration** → **Centralized Installation Manager** → **Installation Packages**, as shown in Figure 29-27 and then click the package name.

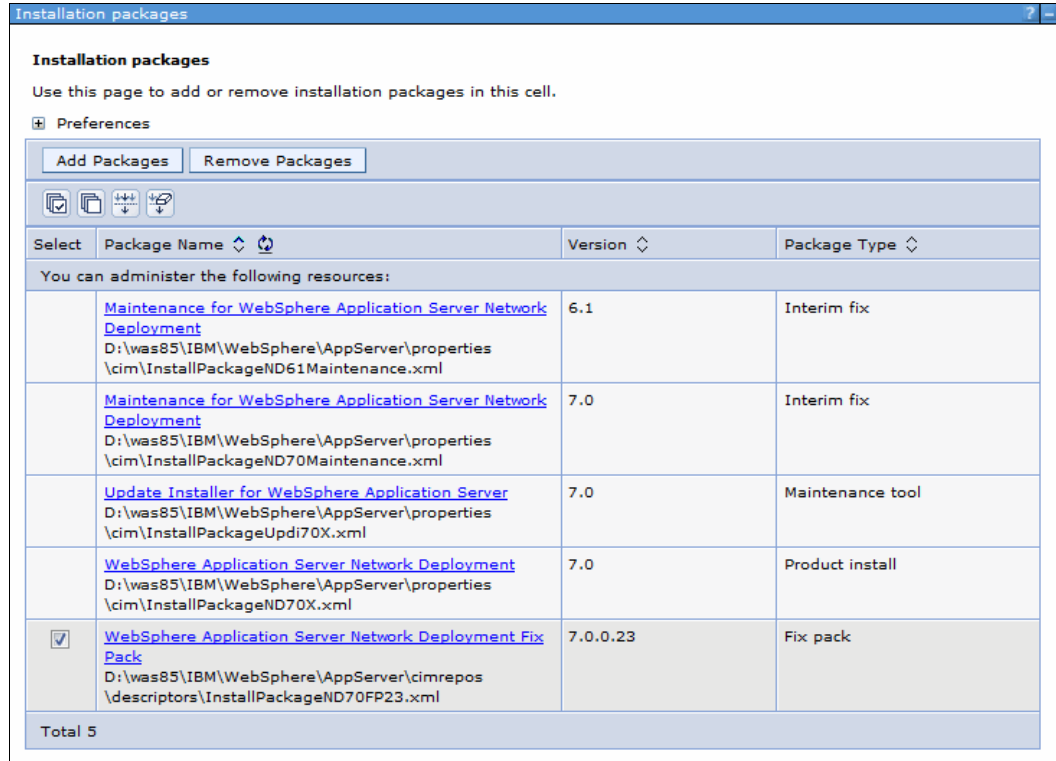


Figure 29-27 Available packages in the centralized installation manager

2. Figure 29-28 shows the next window. Select one or more platforms and then select **Download** to proceed.

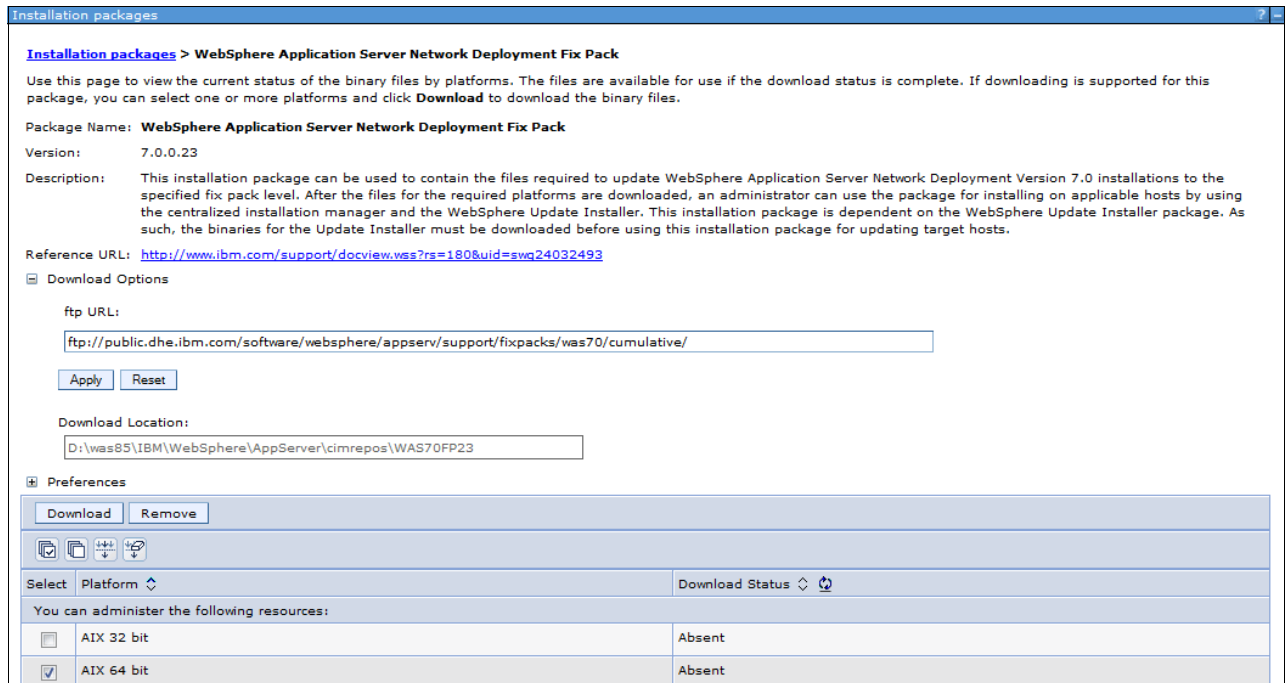


Figure 29-28 Downloading fix packs to the centralized installation manager repository

3. When all the required files are downloaded, the Download Status column displays Complete. If one or more files are missing, the Download Status column displays an Incomplete status. In this case, try to download again. If your status is Incomplete, check for error messages in the *profile_root*/logs/dmgr/SystemOut.1.log file, where *profile_root* is the profile location of the deployment manager.

Downloading interim fixes

To download interim fixes:

1. To download a specific APAR, click **System Administration** → **Centralized Installation Manager** → **Installation Packages**. Click the name of the package file, and a new window opens.
2. Select **Add files** to go to the Download Files window.
3. Enter the APAR number and then select **Search**, as shown in Figure 29-29 on page 1043.

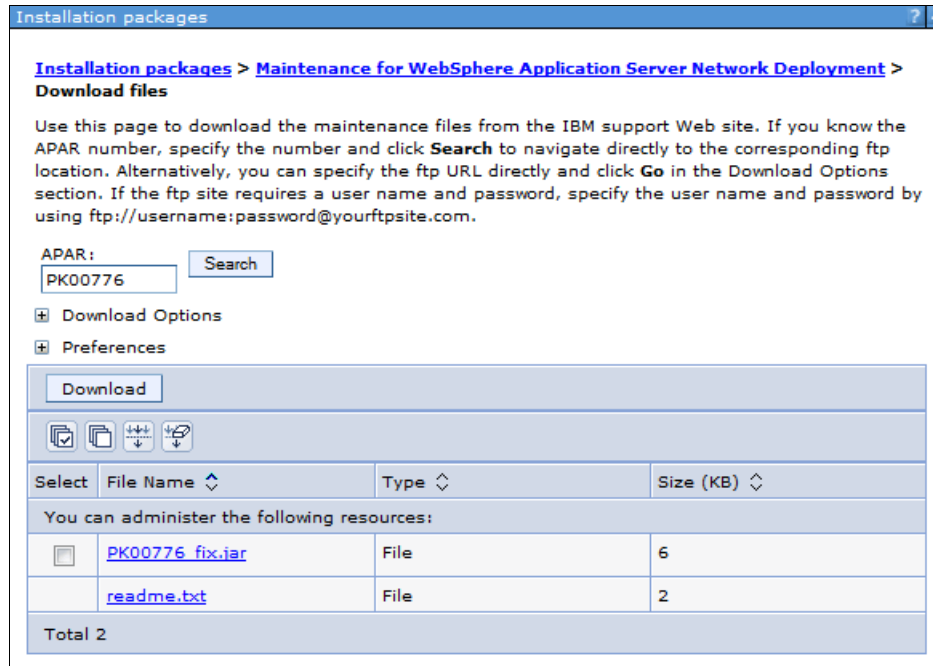


Figure 29-29 Searching for an interim fix using an Internet connection

4. Select the APAR from the list provided and then select **Download**, verify the information, and click **Download** to proceed.

29.6.4 Adding packages when the deployment manager does not have access to the Internet

If your deployment manager system does not have Internet access, you must manually download and transfer files to the centralized installation manager repository.

Before you can copy the downloaded files into the repository, ensure that you set up the directory structure described in 29.4.2, "The centralized installation manager repository structure" on page 1008.

To obtain the address of the FTP site to manually download the required file, click **Administration console** → **System Administration** → **Centralized Installation manager** → **Installation Packages** and then click **Add Packages**. The FTP URL format is:
ftp://public.dhe.ibm.com/software/websphere/appserv/support/cim/cim70_yyyymmdd

If the deployment manager does not have Internet access, an error message is displayed that the FTP URL is not known (Figure 29-30). Write down the FTP URLs because they are used on the system that has Internet access.

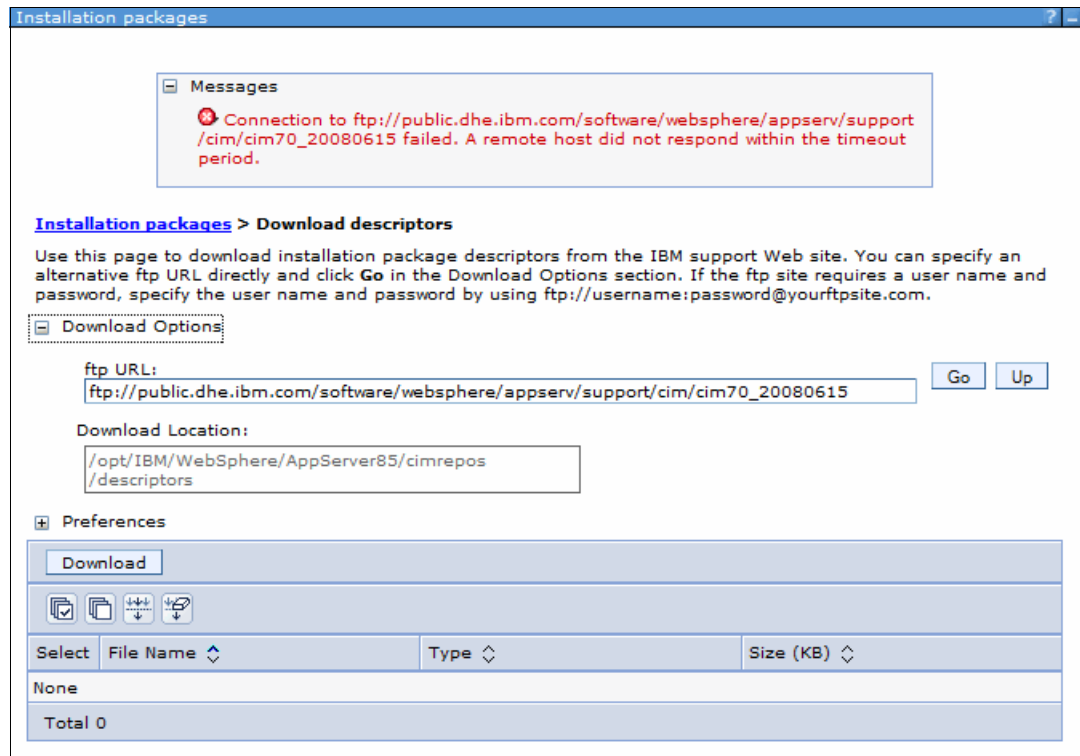


Figure 29-30 Obtaining the location of the fix packs download URL

You can find the address of the FTP site for the following items:

- ▶ Descriptor files: Click **System Administrator** → **Centralized Installation manager** → **Installation Packages**, and click the **Add packages** button. To determine the location of the FTP server, expand the **Download Options**, which gives you the FTP URL location used by the centralized installation manager for downloading the Descriptor files.
- ▶ Update Installer files: Click **System Administrator** → **Centralized Installation manager** → **Installation Packages**, and click **Update Installer for WebSphere Application Server**. Expand the **Download Options**. This gives you the FTP URL location used by the centralized installation manager for downloading the Update Installer for the various operating systems.
- ▶ Fix packs: Copy descriptor files to the descriptor directory of your centralized installation manager, click **System Administrator** → **Centralized Installation manager** → **Installation Packages**, click the name representing the particular fix pack on the table (such as WebSphere Application Server Network Deployment Fix Pack 7.0.0.17), and expand **Download Options**. This action gives you the FTP URL location used by the centralized installation manager for downloading cumulative fix packs and individual fixes. Choose the FTP URL for the type of fix you are downloading.
- ▶ Interim Fixes: Click **System Administrator** → **Centralized Installation manager** → **Installation Packages**, click **Maintenance for WebSphere Application Server Network Deployment 6.1** or **Maintenance for WebSphere Application Server Network Deployment 7.0**, and click **Add Files**. The FTP URL is available under Download Options section.

With the FTP URLs, you can go to any system with Internet access and download the required files. After the files are downloaded, copy them to the correct directory structure in the centralized installation manager repository. For more information about this topic, refer to 29.4.2, “The centralized installation manager repository structure” on page 1008 or the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tins_cim_files_manual_add.html

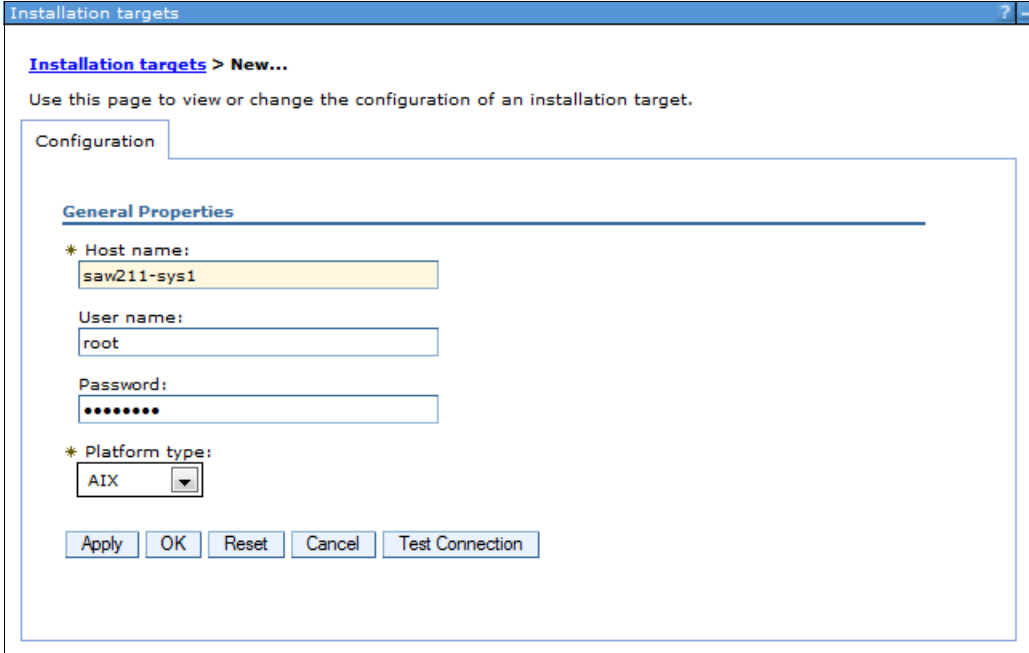
Another option for obtaining the latest Update Installer, fix packs, and interim fixes is to set up an FTP gateway on a system that has Internet access. Refer to the information center at the following website for the steps to set up an FTP Gateway:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tins_cim_files_download_no_internet.html

29.6.5 Adding and removing additional installation targets

To install products or maintenance on remote hosts, you need to specify the targets:

1. Click **System Administration** → **Centralized Installation Manager** → **Installation Targets** → **Add Installation Target**.
2. Enter the host name, user name, password and Platform type of the system that you want to add (Figure 29-31). Consider using a host name rather than an IP address because this name is used in the configuration of the node.



The screenshot shows a web-based configuration window titled "Installation targets" with a sub-tab "New...". The main heading is "Installation targets > New...". Below this is a sub-heading "Configuration" and a note: "Use this page to view or change the configuration of an installation target." The form contains a section titled "General Properties" with the following fields: "Host name:" with the value "saw211-sys1", "User name:" with the value "root", "Password:" with masked characters ".....", and "Platform type:" with a dropdown menu set to "AIX". At the bottom of the form are five buttons: "Apply", "OK", "Reset", "Cancel", and "Test Connection".

Figure 29-31 Defining targets for the centralized installation manager Version 6.1 and Version 7 targets

3. Click **OK**. The new target system is added to the list of target systems, shown in Figure 29-32 on page 1046.

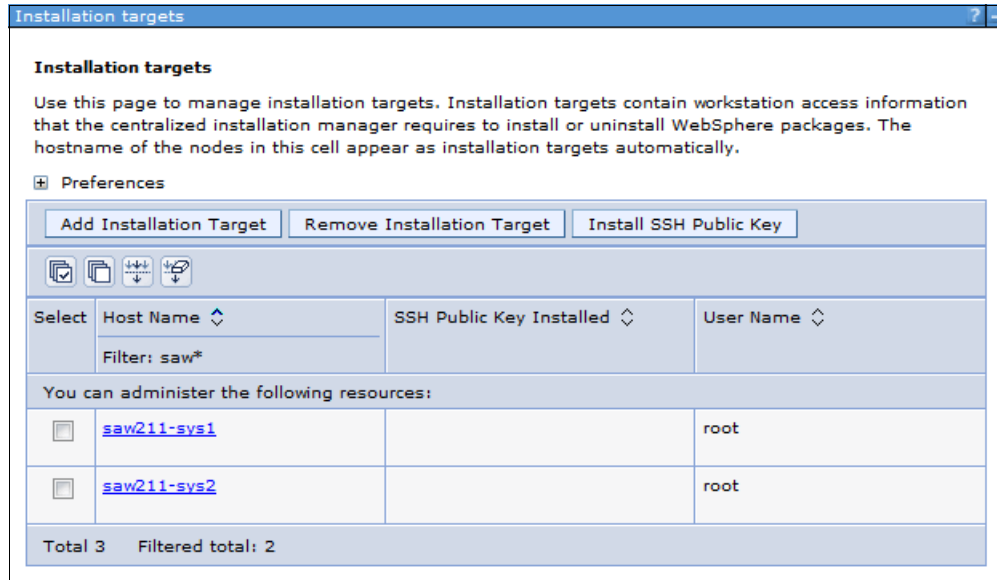


Figure 29-32 Listing the centralized installation manager Version 6.1 and Version 7 targets

- To remove a target, click **System Administration** → **Centralized Installation Manager** → **Installation Targets** (Figure 29-32), select a target, and click **Remove Installation Target**.

29.6.6 Installing a Secure Shell public key

Complete the following steps to install a Secure Shell (SSH) public key on specific installation targets:

- Click **System administration** → **Installation Targets**, select one or more targets from the table, and click **Install SSH Public Key**.
- On the next window, enter the operating system, user name and password and then click **Next**.
- You are prompted for the specific SSH public key location (Figure 29-33 on page 1047). Enter the file location, and click **Next** to continue.

To learn more about how to obtain the keys, refer to 29.1, “The centralized installation manager prerequisites” on page 1002.

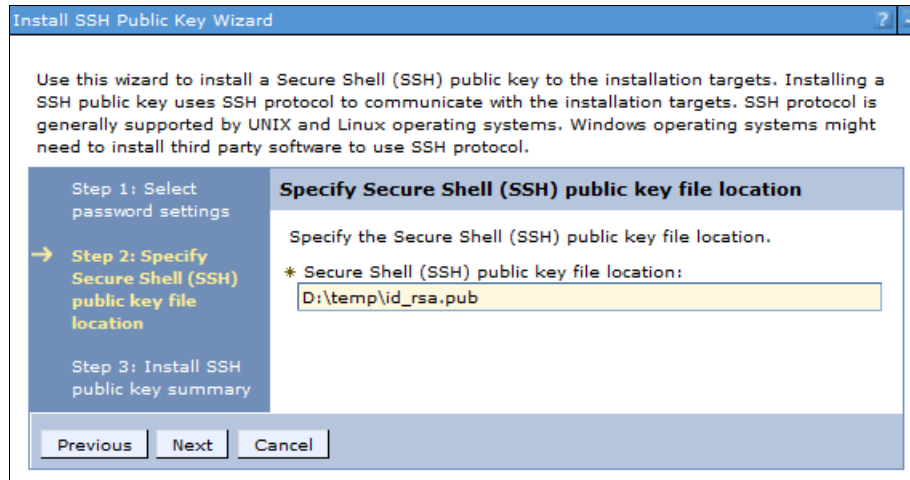


Figure 29-33 Specifying the public key location

4. Verify the summary and then click **Finish**.

After the task finishes, a window similar to Figure 29-34 is displayed.

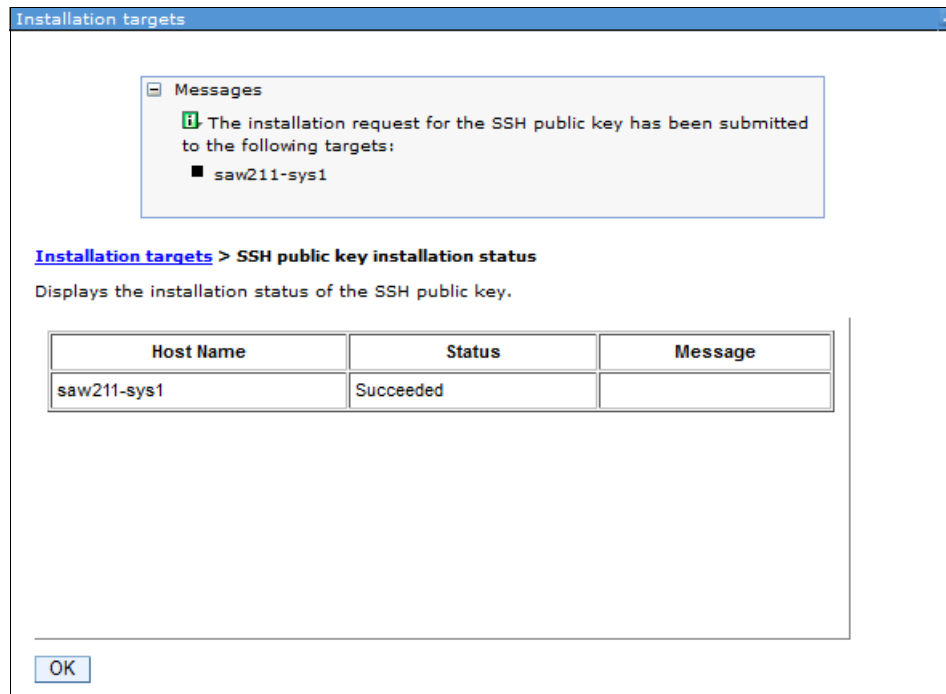


Figure 29-34 Successful installation of the public key on target message

You also see a key icon in the target list, as shown in Figure 29-35 on page 1048. This icon means that your deployment manager host can authenticate with that target using the public/private key pair method.

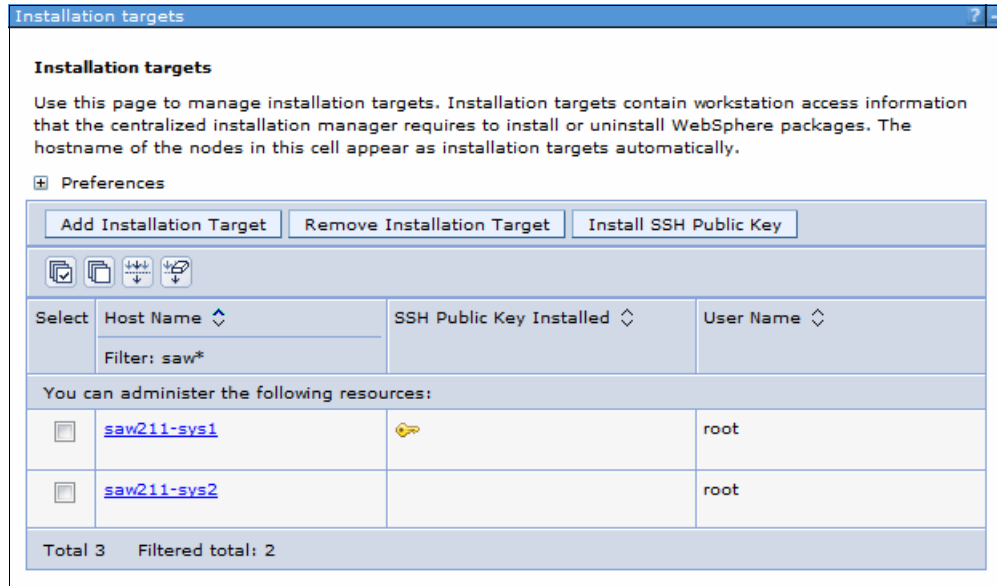


Figure 29-35 Listing the centralized installation manager Version 6.1 and Version 7 targets

For more information about accessing your remote workstations using the SSH public/private key pair authentication method, refer to “Installing a secure shell public key to access remote targets” on page 1002 or go to the information center at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tins_cim_targets_ssh.html

29.6.7 Installing packages to the target systems

The centralized installation manager relies heavily on remote node information maintained locally on the deployment manager node. This remote node information (namely the node-metadata.properties file) for each node is refreshed every time the node agent on the remote node starts and provides the centralized installation manager with up-to-date information regarding the WebSphere products and versions that are installed on the target nodes.

One example of how the node-metadata.properties file is used by the centralized installation manager is in the filtering of nodes that might be selected for the installation of an interim fix.

The node-metadata.properties file is also used by the centralized installation manager to determine only the applicable nodes during a maintenance installation. This process allows the cell administrator to see which nodes are potential candidates for this update and then initiate the installation of the interim fix on one or all the candidate nodes. Because of the availability of the node-metadata.properties file on the deployment manager node, you can use the centralized installation manager to perform this filtering without accessing the target nodes. The node agent process that runs on each node ensures that the node-metadata.properties files of the nodes on the deployment manager are kept up to date.

For this reason, if you apply maintenance to the node or install new WebSphere products (such as the Feature Pack for Web Services) outside of the centralized installation manager on the remote node, you must restart the node agent process after the installation to get the deployment manager copy of the node-metadata.properties file of the node up to date.

29.6.8 Installing a software package

Complete the following steps to install a software package:

1. Click **System Administration** → **Centralized Installation Manager** → **Available Installations**.
2. Select the package type **Product Install**, and select the installation package **WebSphere Application Server Network Deployment - 7.0**. When selecting a product install, you are required to select **Optional features**. You can choose from the following features:
 - Install the sample applications for learning and demonstration environments.
 - Install the non-English language files for using the administrative console from machines with non-English locales. If you do not select this option, only the English language pack is installed.
 - Install the non-English language files that support the application server runtime environment, such as the `wsadmin` tool and logging. If you do not select this option, only the English language pack is installed.

3. Click **Show Installation Targets** to list defined targets (Figure 29-36), and select the targets on which you want to install the product.

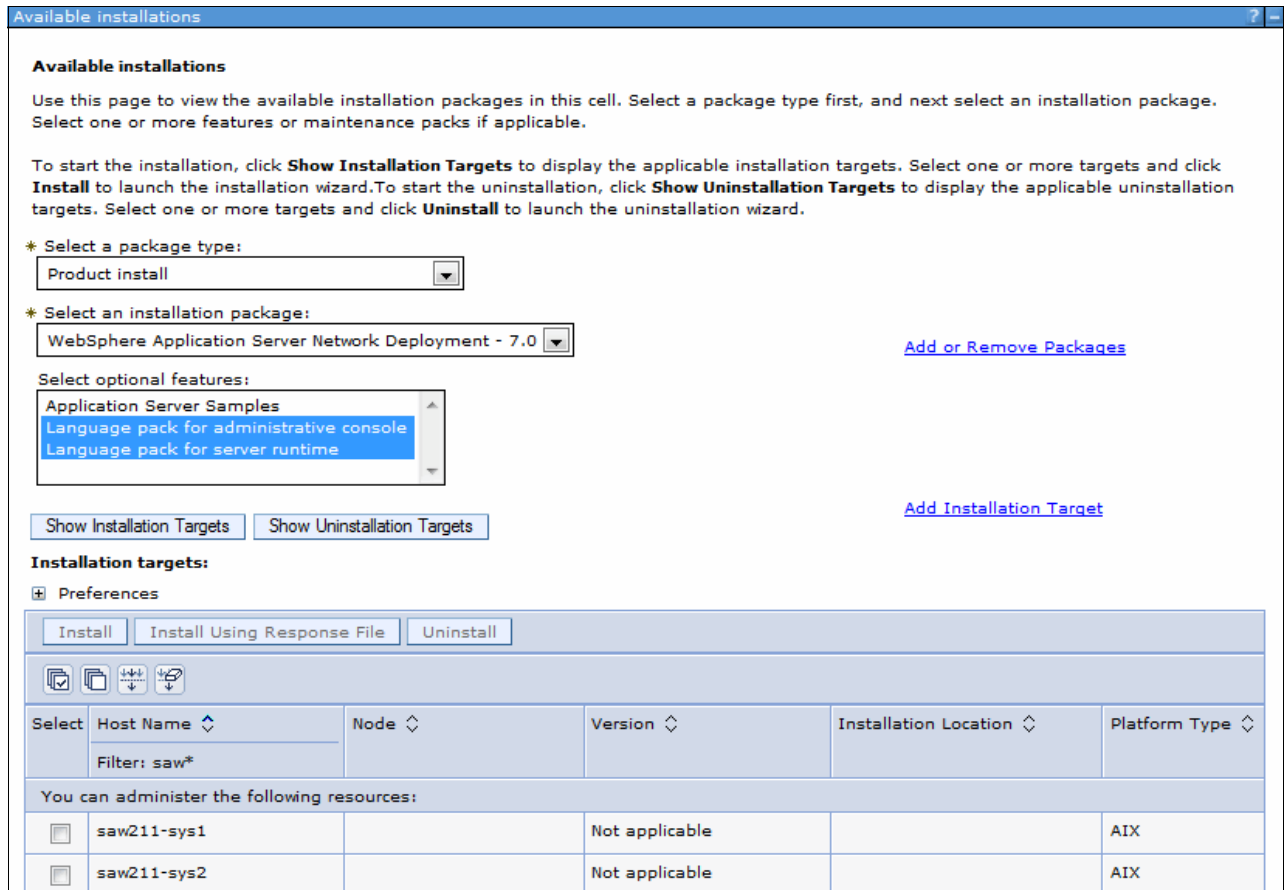


Figure 29-36 Installation of WebSphere Application Server V7 using the centralized installation manager

4. Click **Install** → **Accept License Agreement** → **Next**.
5. Select **Authentication method** to access the target. You can choose a user name and password or the use of Secure Shell (SSH) Public/Private key authentication.
6. In the next window, provide the installation directory and a working (temporary) directory and then click **Next**.
 The next two windows give you the option to disable prerequisite checking and to accept limitations to allow installing as a non-root user.
7. By default, the centralized installation manager uses 64-bit installation binaries on 64-bit operating systems. The next window allows you to override this setting. Click **Next**.
8. Review the Summary window, and select **Finish** to start the installation.
9. To watch the progress of the installation, click **System Administrator** → **Centralized Installation Manager** → **Installation Progress**.

10. When the installation is complete, click **System Administrator** → **Centralized Installation Manager** → **Installation History** for details about the installation.

29.6.9 Installing maintenance to a target system

The centralized installation manager cannot be used to install maintenance to the deployment manager.

All node agents in the cell must be stopped on the target systems. If the node agents or any other server processes are running, it is up to the administrator to make sure that they all are stopped.

Note: There is no need to explicitly install the Update Installer on the targets first before initiating the installation of the fix packs or interim fixes because the centralized installation manager automatically installs the latest version of the Update Installer on the target if needed.

After the centralized installation manager successfully completes the installation process on a remote node, it then deletes the installation image files that are located in the temporary location that you specify during the installation process. If the installation is unsuccessful, the files remain in the temporary location for you to use to determine what caused the installation error. However, you can safely delete these files.

Important: Fix packs that include updates to the Software Development Kit (SDK) might overwrite unrestricted policy files. Back up unrestricted policy files before you apply a fix pack and reapply these files after the fix pack is applied.

Using the centralized installation manager to install refresh or fix packs

To use the centralized installation manager to install refresh or fix packs:

1. Click **System Administrator** → **Centralized Installation Manager** → **Available Installations**. For the package type, select **Refresh pack, fix pack, or maintenance tool**.
2. From the drop-down list of available installation packages, choose the installation package that contains the refresh pack or fix pack that you want to install on your remote systems. These are the packages that you previously downloaded to your centralized installation manager repository.
3. Click **Show installation targets** to get a list of target systems that are available for install. Select your target systems. To continue, click **Install**.
4. The next window shows the license agreement. Review the agreement, select **Agree** and then click **Next**.
5. On the next window, specify the authentication method you want to use, your user name, and password or Secure Shell (SSH). Choose your method and then click **Next** to proceed.
6. The next window that opens depends on the type of authentication you choose. You see a window prompting you to enter a user name and password or a window prompting you for the location of the SSH private key file and keystore password.

7. Verify the installation and the working location of the installation targets. The installation location is the remote location of each installation target in which the package is to be installed. The working location specifies the directory on the remote target where the files are sent before the package is installed in the specified location. Make sure that you have enough disk space in both the installation location and the working location. Click **Next** to continue.
8. A summary window opens. Review the information entered, then to start the installation, select **Finish**.

Note: Any interim fixes that you previously installed on the remote targets are uninstalled by the Update Installer prior to installing the refresh pack or fix pack. If the refresh pack or fix pack does not include the official fixes that were included in the removed interim fixes, you must reinstall the interim fixes after you install the refresh pack or fix pack.

9. You can click **System Administrator** → **Centralized Installation Manager** → **Installation Progress** to check the progress of the installation. When complete, click **System Administrator** → **Centralized Installation Manager** → **Installation History** for details about the installation.

Using the centralized installation manager to install interim fixes

Complete the following steps to install an interim fix on remote target:

1. Click **System Administrator** → **Centralized Installation Manager** → **Available Installations**. From the Select a package type drop-down menu, select **Interim fix**.
From the Select an Installation package drop-down menu, select either **Maintenance for WebSphere Application Server Network Deployment 7.0** or **6.1**. If the interim fixes were previously downloaded to the centralized installation manager repository, they are displayed under Select one or more maintenance packs. Select the maintenance pack you want to install on your target systems.
2. Click **Show Installation Targets**. A list of your target systems are displayed. From this list, select your targets, and click **Install**. Note that you can select only targets that are applicable for that interim fix.
3. Click **View License Agreement** to read the agreement and accept the terms. Click **Next** to continue.
4. The next window is where you choose the authentication method you want to use, user name, and password or Secure Shell (SSH). Choose your method, and click **Next** to proceed.
5. The next window that opens depends on the type of authentication you choose. A window that prompts you to enter a user name and password opens or a window that prompts you for the location of the SSH private key file and keystore password opens.
6. Verify the installation and the working location of the installation targets. The installation location is the remote location of each installation target in which the package is to be installed. The working location specifies the directory on the remote target where the files are sent before the package is installed in the specified location. Make sure you have enough disk space in both the installation location and the working location. Click **Next** to continue.
7. Verify the Summary window, and select **Finish** to start the installation request. To check the status of your request, select **Installations in Progress** on the administrative console. After the installation is completed, click **Installation history** in the administrative console to review the log files for each of the installation requests that you submit. From the

Installation History window, the administrator can click **View Details** to open a window with additional details about the results. Links to logs on the remote targets are included. However, those logs can be moved, replaced, or deleted by other users or administrator if they are not viewed immediately after an installation operation.

If you attempt to install an interim fix without having a copy of the IBM Update Installer for WebSphere Software in your centralized installation manager repository, you receive the following error message:

The installation binary files required for the `install_package_name` or its dependent package Update Installer for WebSphere Application Server for `workstation_platform` do not exist.

29.6.10 Uninstalling packages

Use the following steps to use the centralized installation manager to uninstall previously installed packages on your installation targets. The tasks available for uninstall depend on your environment:

1. Click **System Administrator** → **Centralized Installation Manager** → **Available Installations**. On this window, select the package type you want to uninstall. Select the installation package to be uninstalled. Click **Show Uninstallation Targets** → **Target system** and then click **Uninstall**.
2. You are prompted for an authentication method. Enter either a user name and password or the location of the SSH key file. Next, verify the Installation target directory. Review the summary page, then click **Finish** to proceed with the uninstallation.

Refer to the installation history for the results of the uninstallation.

29.6.11 The centralized installation manager AdminTask commands

You can use the centralized installation manager **AdminTask** commands with Jacl or Jython scripting language. These commands and parameters can be used to install, uninstall, and manage various software packages and maintenance files in the centralized installation manager environment. Here is a list of **AdminTask** commands that are available for WebSphere Application Server V6.1 and V7 products:

| | |
|--|---|
| installWASExtension | Installs a server extension package. |
| installSoftware | Installs a specified software package. |
| installWithResponseFile | Installs a specified software package using parameters from a response file. |
| installMaintenance | Installs maintenance. |
| listPackagesForInstall | Lists packages from the centralized installation repository. |
| listFeaturesForInstall | Lists the features of software packages from the centralized installation repository. |
| showPackageInfo | Displays information about a specific software package. |
| showLicenseAgreement | Displays the license agreement for a specified installation package. |
| getManagedNodesOnHostByInstallLoc | Lists the names of managed nodes defined in the deployment manager cell. |

| | |
|--|--|
| listManagedNodesOnHost | Lists the names of the managed nodes located on targets in the deployment manager cell. |
| testConnectionToHost | Verifies if a connection from the deployment manager to a remote host can be established using a user name and password. |
| testConnectionToHostUsingSSHKey | Verifies if a connection from the deployment manager to a remote host can be established using an SSH private key. |
| installSSHPublicKeyOnHost | Installs an SSH public key on the remote target. |
| listKeyInstallationRecords | Lists the centralized installation manager SSH public key records of target host names. |
| updateKeyInstallationRecords | Updates the centralized installation manager SSH public key records. |
| listPendingRequests | Lists submitted installation or uninstallation requests that are not started yet. |
| listInProgressRequests | Lists submitted installation or uninstallation requests that are in progress. |
| listRequestsForTarget | Lists submitted installation or uninstallation requests for given target. |
| showLatestInstallStatus | Displays the status of the most recently submitted installation request. |
| showLatestUninstallStatus | Displays the status of the most recently submitted uninstallation request. |
| uninstallSoftware | Uninstalls the software package from the target. |
| uninstallMaintenance | Uninstalls the maintenance package from the target. |

Refer to the information center at the following website for detailed information about these commands:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/rxml_cim_commands.html



System recovery

Setting up a system backup and recovery process can help an administrator work efficiently when a system breakdown occurs. WebSphere Application Server V8.5 recovery provides enhanced features for environment recovery. This chapter covers the following topics:

- ▶ Overview
- ▶ Configuring for backup and restore
- ▶ Configuring checkpoints service
- ▶ Restoring transactions
- ▶ Recovery node with `addNode -asExistingNode` command

Important: To keep your WebSphere Application Server environment healthy, monitor the state of the system and deployed applications to discover possible hardware and software malfunctions. For more information about WebSphere Application Server monitoring, refer to Chapter 16.2, “Enabling monitoring infrastructures” on page 555.

30.1 Overview

To avoid the loss of important assets, the administrator must back up the application-serving environment and be able to recover that environment from a failure to a new environment. An overview of this procedure includes the following actions:

- ▶ Backing up the product environment configuration:
 - Administrative configuration files
 - HTTP configuration (the documentation for the web server)
- ▶ Backing up the applications:
 - Application deployment configuration files
- ▶ Backing up the application data:
 - Business data
 - Servlet session data
 - Transaction data

30.2 Configuring for backup and restore

Most administrative configuration documents, with their saved settings, are represented in WebSphere Application Server as XML files. These configuration files and documents can be located in *profile_root/config*. They must be backed up on a regular basis.

The procedure is as follows:

1. If the system is on a distributed environment, fully synchronize the administrative configuration files. If the system is a stand-alone environment, proceed to the next step.
2. For backing up the system: Run the **backupConfig** command to back up the configuration files.
3. For restoring the system: Run the **restoreConfig** command to restore the configuration files.
4. For manual back up and restoration of the system: You can use the functions for exporting and importing servers and the configuration archives for the profiles. More information, refer the 30.2.3, “Exporting and importing a configuration archive” on page 1059

Note: Restore the configuration only if the backup files are the same level of release, including fixes, as the release to which you are restoring.

For more information about the WebSphere Application Server configuration documents, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=rrun_rconfdoc_descriptions

30.2.1 Backing up a profile configuration

Use the **backupConfig** command to back up a profile configuration. Consider using this command before making any significant configuration changes to your WebSphere Application Server environment.

You can run this command from the *was_home/bin* directory using the **-profileName** option to specify the profile to back up. You can alternatively execute the command from the *profile_root/bin* directory to back up only that specific profile.

The **backupConfig** command compresses the configuration files and stores the compressed file in the current directory or a specified path. The compressed file can be restored using the **restoreConfig** command. By default, **backupConfig** stops all servers in the configuration before performing the backup. The syntax is shown in Example 30-1.

Example 30-1 backupConfig command

```
Usage: backupConfig [backup_file] [-nostop] [-quiet] [-logfile <filename>]
[-replacelog] [-trace] [-username <username>] [-password <password>] [-profileName
<profile>] [-help]
```

The **backup_file** parameter specifies the file where the backup is to be written. If you do not specify a backup file name, a unique name is generated, and the file is stored in the working directory. If you specify a backup file name with a directory name other than the current directory, the specified directory must exist.

For more information about these command options and usage, refer to the information center at the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-mp&topic=rxml_backupconfig

Example 30-2 shows the use of the **backupConfig** command to back up the profile AppSrv01 configuration.

Example 30-2 backupConfig example

```
[root@saw211-RHEL2 bin]# pwd
/opt/IBM/WebSphere/AppServer/bin
[root@saw211-RHEL2 bin]# ./backupConfig.sh -profileName AppSrv01
ADMU0116I: Tool information is being logged in file

/home/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/logs/backupConfig.log
ADMU0128I: Starting tool with the AppSrv01 profile
ADMU5001I: Backing up config directory
           /home/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/config to file
           /home/opt/IBM/WebSphere/AppServer/bin/WebSphereConfig_2012-06-21_1.zip
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node was85AppSrv01
Realm/Cell Name: <default>
Username: admin
Password:
ADMU0510I: Server server1 is now STOPPED
.....
ADMU5002I: 812 files successfully backed up
```

30.2.2 Restoring a profile configuration

Use the **restoreConfig** command to restore a profile configuration from an archive that was previously generated using **backupConfig** for that profile. The **restoreConfig** command

restores the entire contents of the *profile_root/config* directory. If the directory already exists, that configuration directory is renamed to *config.old* (then *config.old_1*, and so on) before the restore begins. By default, all servers on the node stop before the configuration restores so that node synchronization does not occur during the restoration.

Directly making changes to the application files in the *app_server_root/installedApps* directory is a process known as *hot deployment*. Be aware that if you do not make the same changes to the application files in the *app_server_root/config* directory, the changes might be overwritten if you use the **restoreConfig** command.

The **restoreConfig** command syntax is shown in Example 30-3.

Example 30-3 restoreConfig command

```
Usage: restoreConfig backup_file [-location restore_location] [-quiet]
      [-nostop] [-nowait] [-logfile <filename>] [-replaceLog] [-trace]
      [-username <username>] [-password <password>] [-profileName
      <profile>] [-help]
```

For more information about the command, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=rxml_restoreconfig

Example 30-4 shows the results of a restore procedure for an application server profile.

Example 30-4 restoreConfig command usage example

```
[root@saw211-RHEL2 bin]# pwd
/opt/IBM/WebSphere/AppServer/bin
[root@saw211-RHEL2 bin]# ./restoreConfig.sh
/home/opt/IBM/WebSphere/AppServer/bin/WebSphereConfig_2012-06-21_1.zip
-profileName AppSrv01
ADMU0116I: Tool information is being logged in file

/home/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/logs/restoreConfig.log
ADMU0128I: Starting tool with the AppSrv01 profile
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node was85AppSrv01
ADMU0512I: Server server1 cannot be reached. It appears to be stopped.
ADMU5502I: The directory
          /home/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/config already
          exists; renaming to
          /home/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/config.old
ADMU5504I: Restore location successfully renamed
ADMU5505I: Restoring file
          /home/opt/IBM/WebSphere/AppServer/bin/WebSphereConfig_2012-06-21_1.zip
          to location
          /home/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/config
.....
.....
ADMU5506I: 812 files successfully restored
ADMU6001I: Begin App Preparation -
ADMU6009I: Processing complete.
ADMU6002I: Begin Asset Preparation -
ADMU6009I: Processing complete.
```

30.2.3 Exporting and importing a configuration archive

You can use the `ConfigArchiveOperations` command group to export and import server configurations or entire cell configurations as a compressed archive (CAR) file. You can use this capability to replicate server or profile configuration.

The general procedure to use an archived file is:

1. Export a WebSphere Application Server configuration into a compressed archived file containing the server configuration.
2. Optionally, extract the files for browsing or updating for use on other systems, for example, you might need to update resource references.
3. Upload the archive file to the target system.
4. Import the archive file. The import process requires that you identify the object in the configuration you want to import and the target object in the existing configuration. The target can be the same object type as the archive or its parent. Consider the following information:
 - If you import a server archive to a server configuration, the configurations are merged.
 - If you import a server archive to a node, the server is added to the node.

Note: You can use this capability between two application server profiles under the same or different product installations, on the same or different host environments. You can also use this capability to replicate a profile configuration across different platforms, as long as you do not add operating system-specific system properties to your configuration. An exception to this is that configurations of z/OS and distributed platform profiles are not compatible, so you cannot replicate configurations between these platforms.

Profile archive

You can use the `exportWasprofile` command to export the entire cell configuration to a configuration archive. This archive file can be used to restore the configuration or clone the original profile on another machine or system.

Note: Only a base server configuration with a single node is supported by the `exportWasprofile` command.

The `exportWasprofile` command in the `AdminTask` object can be used to export profile configuration. The command executed through `wsadmin` from the `profile_root/bin` directory creates an archive of a profile. Example 30-5 shows an example of using *Jacl* to run the command.

Example 30-5 exportWasprofile using Jacl

```
[root@saw211-RHEL2 bin]# pwd
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/bin
[root@saw211-RHEL2 bin]# ./wsadmin.sh
Realm/Cell Name: <default>
Username: admin
Password:
WASX7209I: Connected to process "server1" on node was85AppSrv01 using SOAP
connector; The type of process is: UnManagedProcess
WASX7029I: For help, enter: "$Help help"
wsadmin>$AdminTask exportWasprofile {-archive
/tmp/was85_archive/Node01_archive_0622.car}
```

```
wsadmin>
```

The **importWasprofile** command in the **AdminTask** object imports and overwrites the profile with the archive file configuration, as shown in Example 30-6.

Example 30-6 importWasprofile example

```
wsadmin>$AdminTask importWasprofile {-archive  
/tmp/was85_archive/Node01_archive_0622.car -deleteExistingServers true}
```

```
wsadmin>$AdminConfig save
```

The **-deleteExistingServers** parameter is optional. It deletes existing servers from the target profile. Consider that when importing a profile with multiple servers, the node has to contain exactly the same number of servers. If the number of servers is not the same, the following error message occurs:

```
ADMB0016E: The number of servers in the configuration archive does not match the  
number of servers in the system configuration. The product only supports  
importWasprofile for profiles with the same number of servers.
```

You can also export and import the proxy profile using the **exportProxyProfile** and **importProxyProfile** commands. For more information, refer to the website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=rxml_atconfigarchive

Server archive

The **exportServer** command in the **AdminTask** object is used to export the server configuration. The command is executed through **wsadmin** from a *profile_root/bin* directory on the application server. Example 30-7 shows using Jacl to run the command to export **server1** (**-serverName** specified) of node **was85AppSrv01** (**-nodeName** specified).

Example 30-7 exportServer example

```
[root@saw211-RHEL2 bin]# pwd  
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/bin  
[root@saw211-RHEL2 bin]# ./wsadmin.sh  
Realm/Cell Name: <default>  
Username: admin  
Password:  
WASX7209I: Connected to process "server1" on node was85AppSrv01 using SOAP  
connector; The type of process is: UnManagedProcess  
WASX7029I: For help, enter: "$Help help"  
wsadmin>$AdminTask exportServer {-archive  
/tmp/was85_archive/server1_archive_0622.car -nodeName was85AppSrv01 -serverName  
server1}  
wsadmin>
```

This process removes applications from the server that you specify and breaks the relationship between the specified server and the core group of the server, cluster, or bus membership. If you export a single server of a cluster, the relation to the cluster is eliminated in the archive file.

The **importServer** command of the **AdminTask** object imports an archived server template from the previous **exportServer** result. Example 30-8 on page 1061 shows using Jacl to run

the command for importing an archive (generated in Example 30-7 on page 1060 as server1) into node was85AppSrv01 as server2.

Example 30-8 importServer example

```
wsadmin>$AdminTask importServer {-archive
/tmp/was85_archive/server1_archive_0622.car -nodeInArchive was85AppSrv01
-serverInArchive server1 -nodeName was85AppSrv01 -serverName server2}
server2(cells/saw211-RHEL2Node01Cell/nodes/was85AppSrv01/servers/server2|server.xml
1#Server_1340371919134)

wsadmin>$AdminConfig save
```

When you use the **importServer** command, you can specify a source and a target. First run the command and then you can select a configuration object in the archive (**-nodeInArchive** and **-serverInArchive** specified) as the source and select a configuration object on the system (**-nodeName** and **-serverName** specified) as the target. Figure 30-1 shows the Application Server view in the administrative console after importing server2. As you can see, server2 is added to the existing configuration.

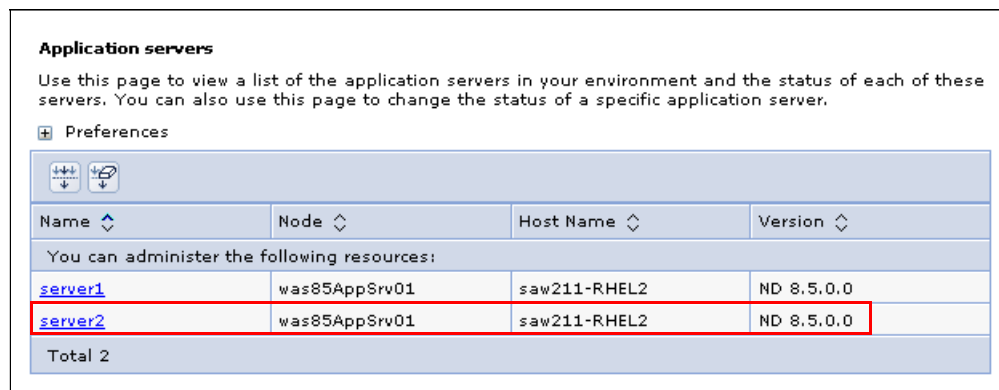


Figure 30-1 Import server to existed configuration

You can also export and import the proxy server using the **exportProxyServer** and **importProxyServer** commands. For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=rxml_atconfigarchive

30.3 Configuring checkpoints service

WebSphere Application Server V8.5 introduces the repository checkpoints service to improve administration configuration changes. Repository checkpoints represent saved images of the repository before configuration changes are made. Checkpoints are one of the following two types:

- ▶ Full checkpoint: The full checkpoint is created manually by the administrator and is a copy of the entire configuration repository. This includes applications and connectors.
- ▶ Delta checkpoints: A delta checkpoint can be created automatically when configuration changes are made and saved to the configuration repository. The delta checkpoint is formed by making a copy of the configuration documents affected by the configuration change before changes are actually applied.

There are multiple operations used to manage the repository checkpoint service, which we cover in the following section. For more information about the checkpoints service configuration, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=twve_xdappedcfg

30.3.1 Creating repository checkpoints

In this section, we discuss the multiple operations used to manage the repository checkpoint service.

Creating a full checkpoint

To create a full checkpoint:

1. Click **System administration** → **Extended repository service** → **Repository checkpoints**. Use the Repository checkpoints page, where you can create, delete, and restore checkpoints, as shown in Figure 30-2.

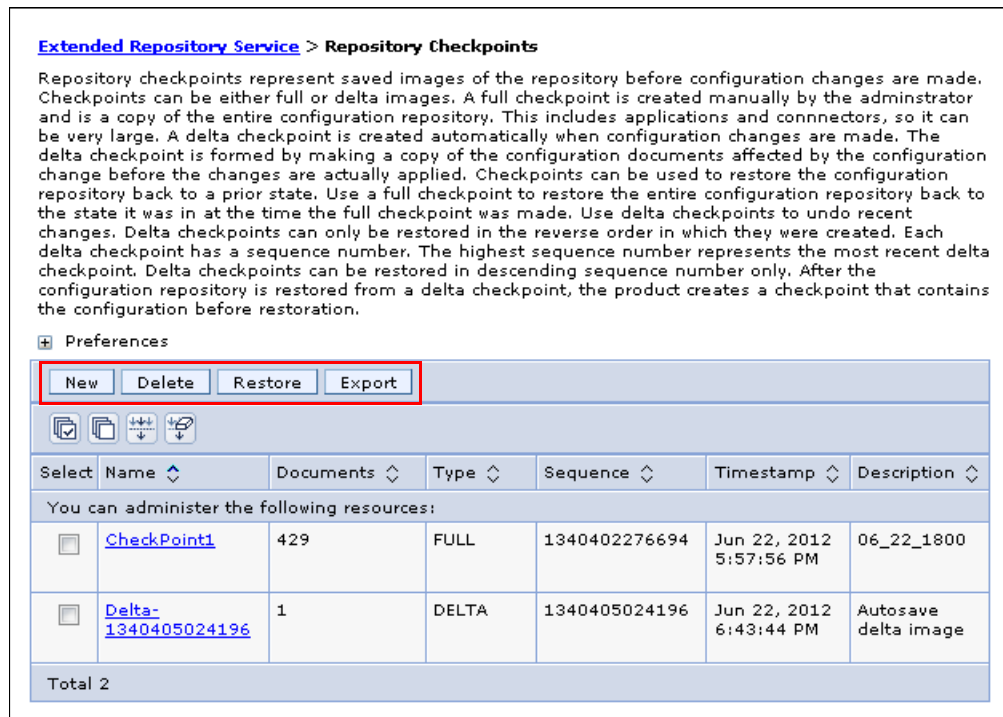


Figure 30-2 Repository checkpoints page

2. Select **New**. You are prompted for confirmation before proceeding. While the checkpoint is being created, the repository is locked as read-only mode. Any attempt to make a configuration change during this period fails.
3. Type the Name and description of the checkpoint.
4. Click **Apply** or **OK**.

Enabling and disabling automatic checkpoints

To enable or disable an automatic checkpoint:

1. Click **System administration** → **Extended repository service**, as shown in Figure 30-3.

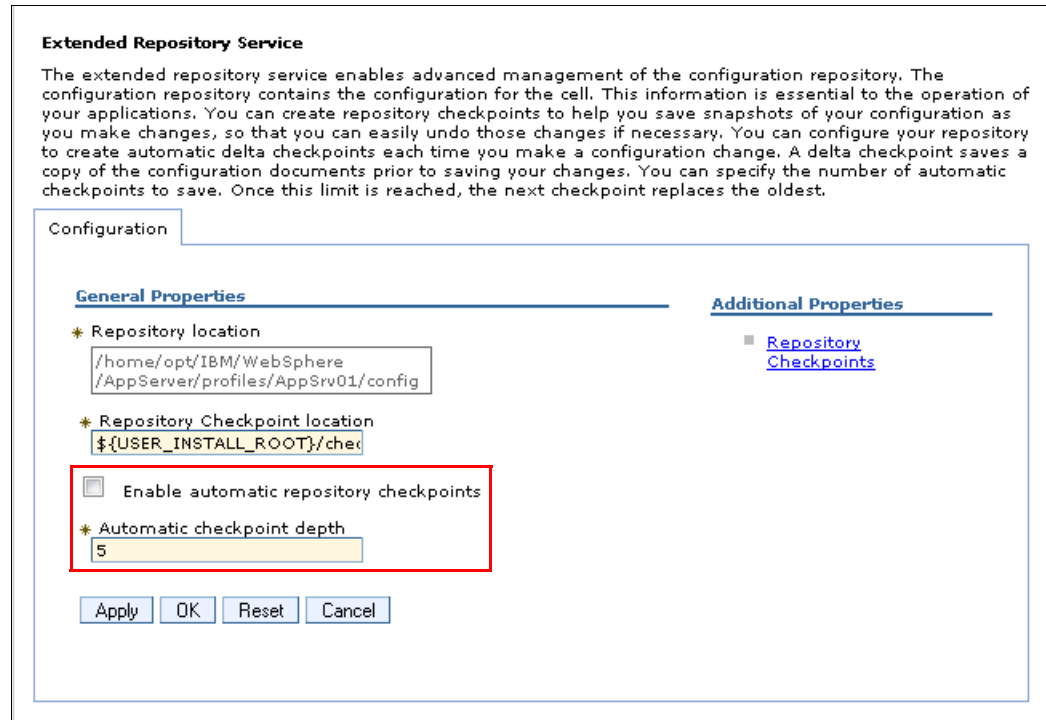


Figure 30-3 Extended Repository Service page

2. Select **Enable automatic repository checkpoints** to enable automatic checkpoints. Clear the check box that disables automatic checkpoints.
3. For Automatic checkpoint depth, specify the maximum number of checkpoints to keep. After the number of checkpoints reaches this checkpoint depth, the product deletes the oldest delta checkpoint when a new delta checkpoint is made.
4. Click **Apply** or **OK**.

30.3.2 Archiving or deleting checkpoints

To reduce clutter and free up disk space, you might need to archive or delete old checkpoints periodically. The number of checkpoints that are stored to disk add up when automatic delta checkpoints are enabled and checkpoint depth is high.

Two locations in the product installation hold information for configuration repository checkpoints:

- ▶ *profile_root/config/cells/cell_name/repository/checkpoints* subdirectories hold checkpoint metadata
- ▶ *profile_root/checkpoints* subdirectories hold checkpoint contents

These locations contain subdirectories, one for each checkpoint. Full checkpoints have the user-specified checkpoint name when delta checkpoints are named *Delta-sequence_number*. The file name also contains the time of creation.

There is no *automatically archive* function provided by WebSphere Application Server V8.5, but you can use any archive tools to archive these two location files.

To delete checkpoints use the delete option on the administrative console Repository checkpoints page as previously shown in Figure 30-2 on page 1062. To access the delete option complete the following steps:

1. Select **System administration** → **Extended repository service** → **Repository checkpoints** → *checkpoint_name*.
2. Click the **Delete** button.

30.3.3 Restoring checkpoints

To restore the checkpoints, refer to Figure 30-2 on page 1062, and complete the following steps:

1. Click **System administration** → **Extended repository service** → **Repository checkpoints**.
2. Select one *checkpoint_name* you want to restore and then click **Restore**.
3. Logout and Login again.

Delta checkpoints must be restored in descending sequence number order only. Selecting multiple checkpoints for restoration is not supported because it can only restore one checkpoint at each time.

30.3.4 Configuring change audit

You can also use a delta checkpoint to audit what changes were made to the configuration. A delta checkpoint can be exported as a compressed file. This file contains the before and after versions of configuration files that have been changed.

To export a delta checkpoint:

1. Click **System administration** → **Extended repository service** → **Repository checkpoints**, as previously shown in Figure 30-2 on page 1062.
2. Select the *delta_checkpoint_name*, and click **Export**.
3. On the Export repository checkpoints page, select the name of the compressed file.
4. Save the file to a specified location.
5. Extract files from the exported compressed file, and examine the changes in the configuration.

30.4 Restoring transactions

WebSphere Application Server stores information about transactions that it manages in the form of persistent logs. The default directory for the transaction logs is:

profile_root/tranlog

These logs can be used to recover transactions that did not complete due to, for example, a hardware or software server failure. The WebSphere Application Server recovers transactions during startup. You can also force it to recover transactions by specifying the **-recovery** parameter in the **startServer** command.

The WebSphere Application Server also supports more complex recovery cases when in a clustered environment, referred to as *transactional high availability*. The high availability of the transaction service enables any server in a cluster to recover the transactional work for any other server in the same cluster. For more information about the high availability concept, refer to chapter and section 15.3, “High availability and failover” on page 538.

30.4.1 Restarting an application server in recovery mode

When starting a server with the **-recovery** parameter after a failure, shown in Example 30-9, the transaction service uses recovery logs to complete the recovery process. You must issue the command from the *profile_root/bin* directory of the profile with which the server is associated.

Example 30-9 Start server in recovery mode

```
startServer.sh server1 -recovery
```

Using the additional **-recovery** parameter specifies that the server starts in the special recovery mode. The special recovery mode engages the following actions:

- ▶ Transactional resources complete the actions in their recovery logs. This action frees up any resource locks that the application server held prior to the failure.
- ▶ During the recovery period, only a subset of the application server functions are accessible. The subset includes only those that are necessary for the transactional recovery to proceed.
- ▶ The application server does not accept transactions during recovery.
- ▶ The application server shuts down when the recovery is complete.

30.4.2 Administering the transaction service

You can view or change settings for the transaction service and manage active and prepared transactions. You can configure transaction properties to enable peer recovery of failed application servers in a cluster. You can manage transaction logging to optimize the availability of application servers. The following list describes the configurations that are available and what they manage:

- ▶ Configuring transaction properties for an application server
 - Change the location or default file size of the transaction log files, transaction timeout properties, or heuristic-related properties.
- ▶ Configuring transaction properties for peer recovery
 - Peer recovery for the transaction service enables servers in a cluster to complete outstanding work for a failed cluster member. You can configure and manage the manual and automated peer recovery.
- ▶ Managing active and prepared transactions
 - In some circumstances, you might have to resolve a transaction manually:
 - Manual transactions: Transactions awaiting administrative completion.
 - Retry transactions: Transactions with some resources being retried.
 - Heuristic transactions: Transactions that have completed heuristically.
 - Imported prepared transactions: Transactions that are imported and prepared but not yet committed.

- ▶ **Managing transaction logging for optimum server availability**

The default disk space allocation for the transaction logs is 1M. If all the log space is in use, no more transactions can commit until more log space is made available. You can change the space allocation of transaction log files. It is useful in high workloads scenario. You can also store the transaction log in a highly-available file system to complete or recover transaction more quickly.
- ▶ **Displaying transaction recovery audit messages**

You can configure the server to use the *High Performance Extensible Logging* (HPEL) log and trace infrastructure instead of using SystemOut.log and trace.log. Set the DISABLE_RECOVERY_AUDIT_LOGGING custom property to turn off the transaction recovery message written to the SystemOut.log.
- ▶ **Delaying the cancelling of transaction timeout alarms**

By default, transaction timeout alarms are cancelled prior to the *before completion phase* of the transaction begins. The DELAY_CANCELLING_ALARMS custom property allows the *before completion phase* of the transaction to be encompassed within the transaction timeout period.
- ▶ **Removing entries from the transaction partner log**

As part of the transaction recovery process, the partner log is checked to establish which resources are needed. You can set the REMOVE_PARTNER_LOG_ENTRY custom property to remove certain entries from the partner log, such as a resource.

For more information about the transaction service administration, refer to the following website:

http://www14.software.ibm.com/w ebapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=tjta_administer

30.4.3 Transactional high availability

In a clustered environment, a running instance of a server can recover the failed transactions of another server from the same cluster. There is no need to restart the failed server to recover its transactions. Having another server in the cluster to be able to recover transactions is useful when a hardware failure is the cause of the original server unavailability. Transactional high availability allows for the needed time to recover, especially when the hardware fault is extensive. For more information, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=cjta_trans_ha

There are two possible ways to recover and release the locks of another server in a cluster:

- ▶ **Automated peer recovery**

If an application server fails, WebSphere Application Server automatically selects another server to perform peer recovery processing on its behalf. Apart from enabling high availability and configuring the recovery log location for each cluster member, no additional WebSphere Application Server configuration steps are required to use this mode. This is the default recovery option for WebSphere Application Server installations.
- ▶ **Manual peer recovery**

If an application server fails, an administrator can use the administrative console to select a particular server in the cluster to perform recovery processing on behalf of the failed server. To use this recovery mode, additional configuration steps are required.

Configuration and usage of the clustered environment transaction recovery is out of the scope for this book. However, a well documented article on this topic is available from IBM developerWorks at the following website:

http://www.ibm.com/developerworks/websphere/techjournal/0504_beaven/0504_beaven.html

30.5 Recovery node with `addNode -asExistingNode` command

The WebSphere Application Server `addNode` command is now extended with the additional `-asExistingNode` parameter to simplify the recovery of nodes in a distributed environment. This parameter allows the following abilities:

- ▶ Recover an existing managed node of a deployment manager.
- ▶ Move a node to a product installation on a different computer at the same or different path.
- ▶ Create a cell from a template cell.

30.5.1 Considerations when using the `-asExistingNode` command

- ▶ To make sure your application runs properly in your target environment, update the virtual hosts (host aliases) to include the target host name of the application server node.
- ▶ If the node uses a Secure Sockets Layer (SSL) certificate, the default SSL certificate of the node does not contain the name of the target machine. Change the default certificate to contain the host name of the new node to make sure SSL works properly. Refer to the following website for replacing SSL certificates:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=tsec_sslreplaceselfsigcert

- ▶ You might also need to update the configuration of other infrastructure components, such as web servers, that are statically configured to use application servers residing on specific hosts.
- ▶ Some of the `addNode` command parameters are incompatible with the `-asExistingNode` option. Do *not* use `-asExistingNode` with the following parameters:

- `includeapps`
- `includebuses`
- `startingport`
- `portprops`
- `nodeagentshortname`
- `nodegroupname`
- `registerservice`
- `serviceusername`
- `servicepassword`
- `coregroupname`
- `excludesecuritydomains`

30.5.2 Recovering a failed managed node of deployment manager

In WebSphere Application Server, to recover a damaged node in a distributed environment (*without* the backup) complete the following steps:

1. Reinstall WebSphere Application Server in the same directory as previously used.
2. Create a profile with the same managed node name and path.
3. Recover the damaged node using `addNode -asExistingNode`.

4. Synchronize the nodes in the cell.

Figure 30-4 illustrates this procedure after a node1 failure. When the managed node is unavailable, but the node information remains on the deployment manager, use the **-asExistingNode** option to recreate the unavailable node.

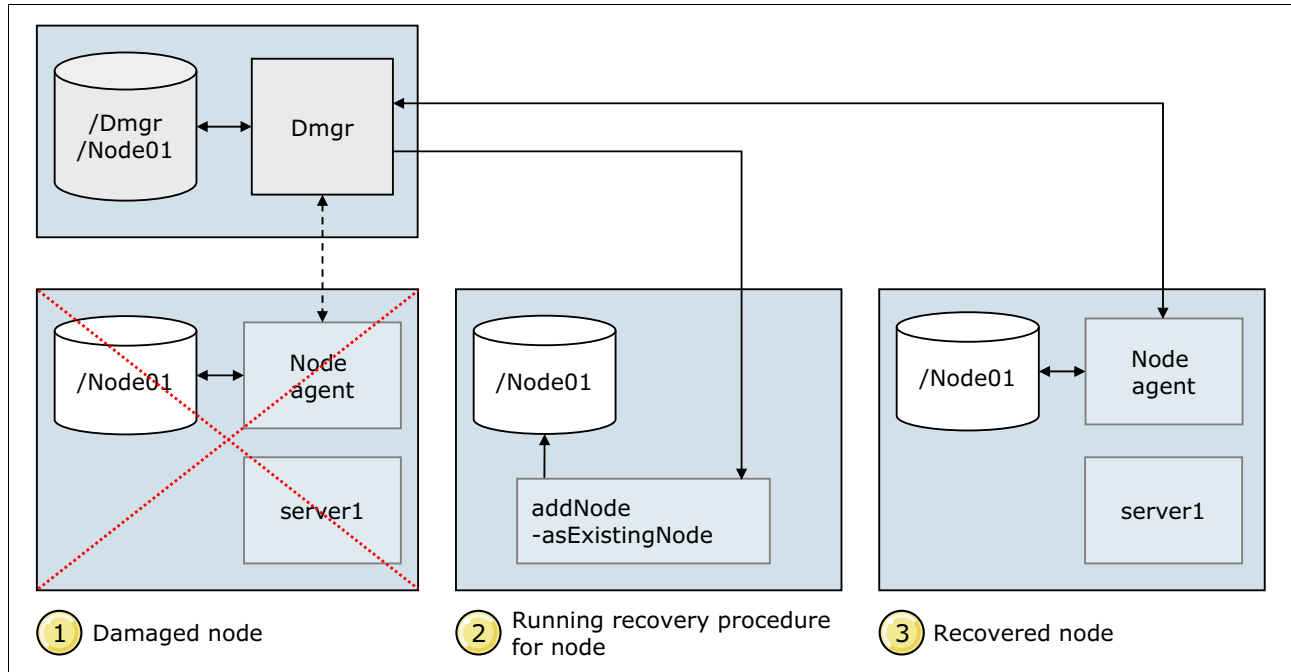


Figure 30-4 Recovering a failed managed node

To accomplish rapid node recovery:

1. Ensure that the existing damaged node is not running. Stop the node agent and any application servers that reside on the node. If the node cannot be stopped using standard commands, stop its processes.
2. Remove the original profile, and create a profile to replace the damaged node and give it the same profile path, profile name, and node name as the unavailable node. Alternatively, you can create the profile on a different computer from the original node, if your original computer is unavailable and you have configured a new one with the same host name.

In this scenario, node was85Node01 within server profile name Node01 stops working. To recover the node, use the following steps:

- Remove the node by using the following command:

```
./manageprofiles -delete -profileName Node01
```

- Replace the damaged node with a new node. Create a profile named Node01 with node name was85Node01. Be sure to use the same directory path and federate the node later. Use the following commands to engage this process:

```
./manageprofiles.sh -create -profileName Node01 -nodeName was85Node01  
-profilePath /opt/IBM/WebSphere/AppServer/profiles/Node01 -federateLater  
true ...
```

3. Run the **addNode** command with the **-asExistingNode** option from the **bin** directory of the damaged node profile. Example 30-10 on page 1069 shows the restoration of the was85Node01 to the environment. The following syntax format is used in Example 30-10 on page 1069:

```
./addNode.sh dmgr_host dmgr_port -asExistingNode -username user_name -password password
```

Example 30-10 Using -asExistingNode to recover a failed Node

```
[root@saw211-RHEL2 bin]# pwd
/opt/IBM/WebSphere/AppServer/profiles/Node01/bin
[root@saw211-RHEL2 bin]# ./addNode.sh saw211-RHEL2 8879 -asExistingNode
-username admin -password admin
ADMU0116I: Tool information is being logged in file
/home/opt/IBM/WebSphere/AppServer/profiles/Node01/logs/addNode.log
ADMU0128I: Starting tool with the Node01 profile
...
ADMU2010I: Stopping all server processes for node was85Node01
ADMU0024I: Deleting the old backup directory.
ADMU0015I: Backing up the original cell repository.
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: was85Node01
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is:
7094
ADMU0300I: The node was85Node01 was successfully added to the was85DmgrCell01
cell.
...
ADMU0003I: Node was85Node01 has been successfully federated.
```

4. Synchronize all of the active nodes in the cell. Your node is fully operational again, as shown in Figure 30-5.

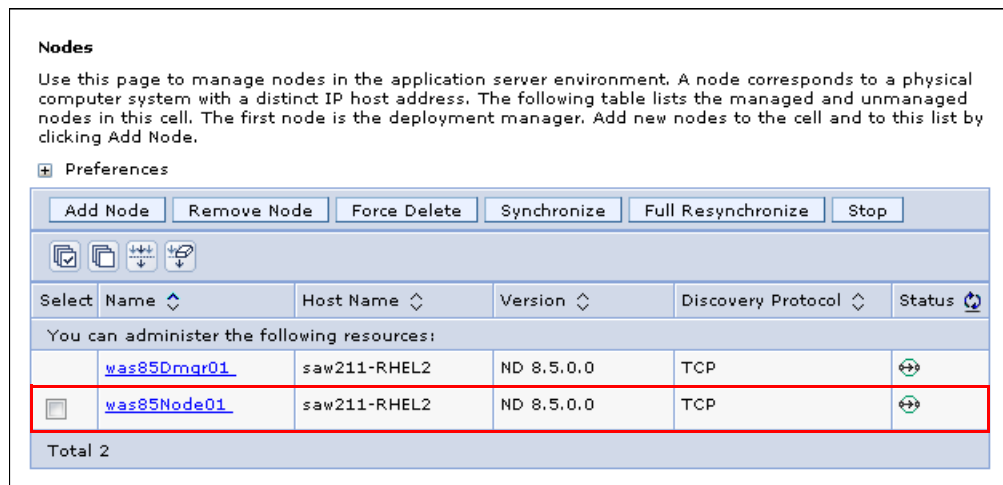


Figure 30-5 Node view after recovering a failed node

30.5.3 Moving a node to a different system

You can also use the `-asExistingNode` option to move the node to a different machine. The procedure overview is shown in Figure 30-6 on page 1070.

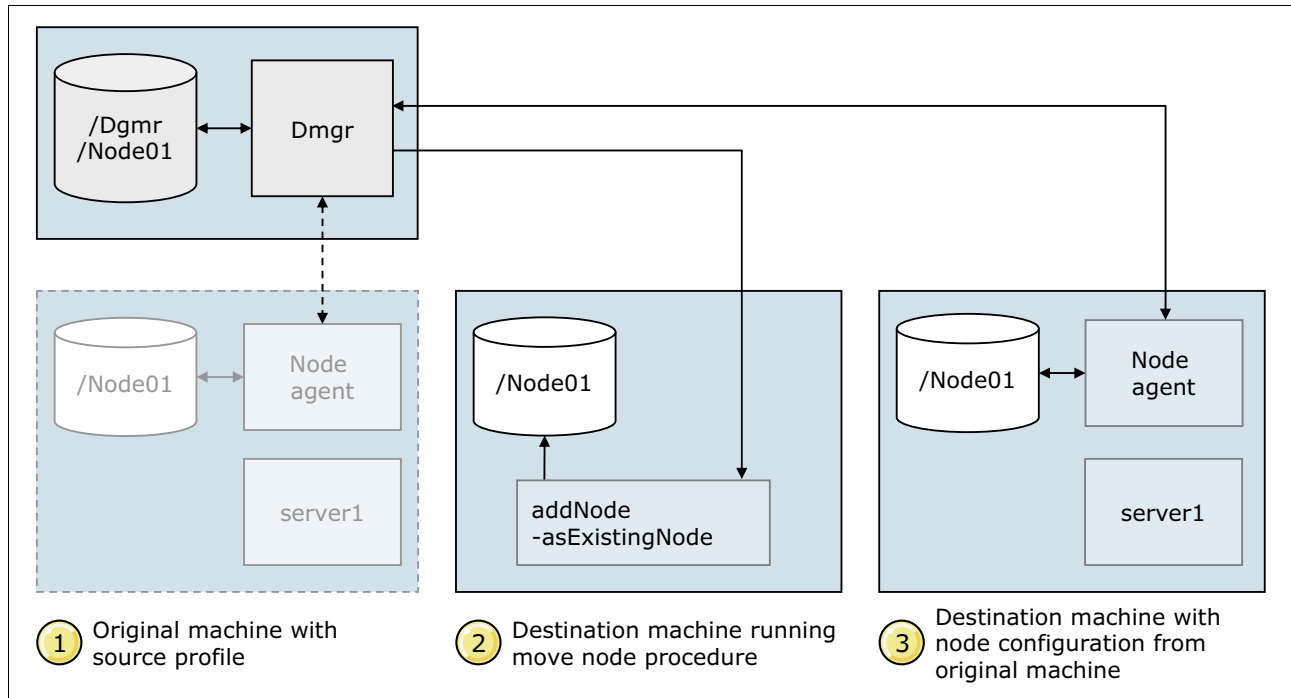


Figure 30-6 Moving a node to a destination machine

In Figure 30-6:

1. Ensure that the node you want to move, with the source profile, is not running. Stop the node agent and any application servers that reside on the node.
2. On the deployment manager, change the host name of the node to target machine host name using the **AdminTask.changeHostName** command.
3. Install WebSphere Application Server on the new machine at the same directory location.
4. Create a profile with the original node name and profile name.
5. Run the **addNode -asExistingNode** command.
6. Synchronize nodes in the network manager cell.

Using **addNode -asExistingNode** to move the node to a different machine involves three different profiles:

- ▶ Deployment manager profile: This profile manages the source profile and is used to manage the destination profile.
- ▶ Source profile: This profile is migrated to a different destination.
- ▶ Target profile: This is the new profile created on the source profile base.

You can use the **addNode -asExistingNode** command to do the following actions:

- ▶ Move a node to a different computer but with same operating system and at the same path.
- ▶ Move a node to a different operating system or with a different path.

In the following scenario, we assume that the source and destination targets use the same profile name and node name, but create a different installation directory and profile path:

1. Stop the node agent or any application server processes on the source profile, and back up the profile. In this example, we use `Node=was85Node01` and `profile=Node01`.

2. Change the host name of the source profile node within the master configuration present at the deployment manager, as shown in Example 30-11. We assume the source machine is saw211-RHEL2 and the destination is saw211-win1.

Example 30-11 Changing the node host name

```
[root@saw211-RHEL2 bin]# pwd
/opt/IBM/WebSphere/AppServer/profiles/Dmgr/bin
[root@saw211-RHEL2 bin]# ./wsadmin.sh -lang jython -username admin -password
admin
WASX7209I: Connected to process "dmgr" on node was85Dmgr01 using SOAP
connector; The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>AdminTask.changeHostName('[-hostName saw211-win1 -nodeName
was85Node01]')
''
wsadmin>AdminConfig.save()
''
```

3. If the target profile has the new product installation and profile paths, include this step (if not, proceed to step 4). This step outlines changing the product installation and profile paths of each node in the variable maps on the deployment manager configuration before moving the node to the target computer. The following process outlines the necessary actions for this step:
 - a. In a deployment manager administrative console, click **Environment** → **WebSphere variables**.
 - b. On the WebSphere Variables page, select the **node scope** as Node=was85Node01 and then click the **WAS_INSTALL_ROOT** variable.
 - c. On the Settings page for the WAS_INSTALL_ROOT variable, change the Value setting to specify the new product installation path, and save the change.
 - d. On the WebSphere Variables page, with the **node scope** selected, click the **USER_INSTALL_ROOT** variable.
 - e. On the Settings page for the USER_INSTALL_ROOT variable, change the Value setting to specify the new profile installation path, and save the change.
 - f. Repeat these steps as needed to change the product installation and profile paths of each node so that the paths are correct for the target computer.
4. Log in to the destination target machine, and do the following steps:
 - a. Install WebSphere Application Server in a directory that has the same name as the product installation directory on the source profile target.
 - b. Create a custom profile that has the same profile name and node name as the one that you want to move, and select to **federate the node later** (if you move the node with the same profile path, keep the profile directory the same):


```
manageprofiles.bat -create -profileName Node01 -profilePath
C:\IBM\WebSphere\AppServer\profiles\Node01 -nodeName was85Node01 -hostName
saw211-win1 -federateLater true ...
```
 - c. Change your working directory to the newly created profile bin directory:


```
C:\IBM\WebSphere\AppServer\profiles\Node01\bin
```
 - d. Run the **addNode** command with the **-asExistingNode** option to replace the application server node with the node that you want to move.

```
./addNode.sh dmgr_host dmgr_port -asExistingNode -username user_name
-password password
```

The results are shown in Example 30-12.

Example 30-12 Successfully move node to a different target

```
C:\IBM\WebSphere\AppServer\profiles\Node01\bin>addNode.bat saw211-RHEL2 8879
-asExistingNode -username admin -password admin
ADMU0116I: Tool information is being logged in file
          C:\IBM\WebSphere\AppServer\profiles\Node01\logs\addNode.log
ADMU0128I: Starting tool with the Node01 profile
...
ADMU2010I: Stopping all server processes for node was85Node01
ADMU0024I: Deleting the old backup directory.
ADMU0015I: Backing up the original cell repository.
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: was85Node01
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is:
2524
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: Cluter1_Member1
ADMU0506I: Server name: nodeagent
ADMU0300I: The node was85Node01 was successfully added to the
was85DmgrCell01 cell.
...
ADMU0003I: Node was85Node01 has been successfully federated.
```

5. Synchronize the nodes using the deployment manager console by clicking **System Administration** → **Nodes**, selecting the unsynchronized nodes, and clicking **Synchronize**.
6. Your node is fully operational again in the new target, as shown in Figure 30-7. You can remove the source profile directory and its backup.

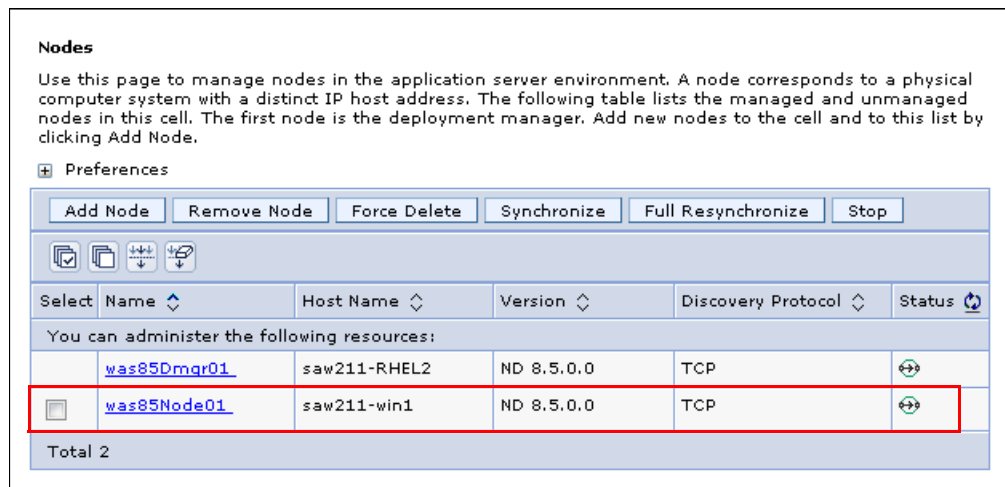


Figure 30-7 Nodes view after move node by asExistingNode

Restriction for moving nodes:

- ▶ Applications that use Scheduler only work with the same host name. After you move a node, reschedule any scheduled tasks.
- ▶ You cannot move nodes between product installations on z/OS and non-z/OS operating systems.
- ▶ Previously installed JCA adapters are not stored as part of the WebSphere configuration. After you move a node, reinstall them.

30.5.4 Recreating a cell from a template

Note: This procedure requires a good knowledge of WebSphere Application Server configurations. You need to update several files, generated by the server, to migrate them to the new environment.

Assuming you have your deployment manager cell environment set up, complete the following steps:

1. Back up the deployment manager profile using the **backupConfig** command.
2. Copy the backup file to a target where your deployment manager will reside.
3. On each new target environment to be provisioned, install WebSphere Application Server and create a deployment manager profile or a managed node profile, depending on your environment topology.
4. Restore the new target deployment manager profile configuration using the **restoreConfig** command and customize each node configuration.
5. Update the deployment manager host name using the **changeHostName** command of the **AdminTask** object through **wsadmin** in local mode. If the profile path or the product installation path changed, update the variables.xml file of the deployment manager node with the new value. Update additional properties files as needed, for example, wsadmin.properties and soap.client.props.
6. Customize each node configuration on the deployment manager profile from the old source environment properties to new target environment properties. For example, change the following settings:
 - Host name: **changeHostname** command of **AdminTask** object.
 - Ports: Ports page of node agent on administrative console.
 - Product installation directory: WAS_INSTALL_ROOT variable.
 - Profile directories: USER_INSTALL_ROOT variable.
 - Security configuration.
7. Log in to each new target node, and run the following command from each node *profile_root/bin* directory:

```
./addNode.sh dmgr_host dmgr_port -asExistingNode -username user_name -password password
```
8. To enable servers for each node to run properly:
 - a. Start the node.
 - b. Update the virtual hosts (host aliases).
 - c. Start servers of the node.
9. If the cell uses a SSL certificate, replace the SSL certificates as described in 30.5.1, “Considerations when using the -asExistingNode command” on page 1067.

10. Synchronize all of the nodes from the deployment manager.

Restrictions when creating a cell using the `-asExistingNode` command:

- ▶ Applications that use Scheduler only work with the same host name. After you move a node, reschedule any scheduled tasks.
- ▶ You must assess whether different resources, such as data sources, are required for each environment.
- ▶ Previously installed JCA adapters are not stored as part of the WebSphere configuration. After you move a node, reinstall them.

For more information about the `-asExistingNode` command, refer to the following website:

http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-nd-dist&topic=tagt_addNode_asExistingNode



Troubleshooting

Problems within an application server environment can take many forms, including poor performance, application unavailability, or unexpected results. The first step in resolving a problem is to isolate and understand it.

In this chapter, we introduce tools and techniques that can be used to analyze and correct problems. The following topics are included in this chapter:

- ▶ Overview
- ▶ WebSphere Application Server logs
- ▶ Tools for collecting and analyzing diagnostic data
- ▶ Troubleshooting scenarios

31.1 Overview

Problem determination and troubleshooting use two basic actions to solve a problem:

- ▶ **Data collection:** Reproduce the problem and collect diagnostic data. Data collection includes Java log files, trace logs, thread dumps, heap dumps, and system dumps.
- ▶ **Analysis and isolation:** Analyze both the results and the diagnostic data until the root cause is found and isolated.

To perform an analysis, begin collecting diagnostic data. Error messages in a log file, for example, might be gathered and analyzed. The error messages might indicate that there was a database connection issue. You might recognize immediately from the diagnostic data that the user ID that tried to authenticate to the database is not authorized to do so. You might have to further analyze the data and eventually discover that there is a network communication problem between the application and database tiers. As you move further in the analysis, you begin to eliminate possible causes for the problem until you find the actual root cause, isolating it from other non-pertinent error messages.

One useful technique, for problem determination, is to categorize and describe the problem based on diagnostic data. The following list gives some questions to get started with this technique:

- ▶ What are the symptoms of the problem?
For example, application server start failures
- ▶ Where does the problem occur?
For example, Web server plug-in
- ▶ When does the problem occur?
For example, during peak system load
- ▶ Has the problem begun after a configuration change?
For example, a new application deployment
- ▶ Can the problem be reproduced?
Enable tracing and collect diagnostic data

31.2 WebSphere Application Server logs

WebSphere provides several useful logs. The following list gives the available logs with an overview description:

- ▶ **Java Virtual Machine (JVM) logs**
Created by redirecting the **System.out** and **System.err** streams of the JVM to independent log files:
 - One set of the JVM logs for each application server and all of its applications is located by default in the following directory:
`<profile_root>/<profile_name>/logs/<server_name>`
 - SystemOut.log and SystemErr.log

- ▶ Process logs

Contain two output streams (stdout and stderr) that are accessible to native code running in the process:

- One set for each application server
- native_stderr.log and native_stdout.log

- ▶ IBM Service logs

The IBM service log contains the WebSphere Application Server messages that are written to the System.out stream. The log also contains some special messages that hold extended service information that is normally not of interest, but can be important when analyzing problems.

- One per profile (node)
- activity.log

31.2.1 Server log files

WebSphere Application Server log files are located in the following folder:

```
<was_root>/profiles/<profile_name>/logs
```

Each application server instance has its own log folder containing their own set of log and trace files. This folder is found using the following location syntax:

```
<was_root>/profiles/<profile_name>/logs/<appserver>
```

Under each application server's folder are several logs. Here are the available logs:

- ▶ The SystemOut.log and SystemErr.log are the standard JVM output and error logs.
- ▶ The startServer.log and stopServer.log contain information logged by the server as it starts and shuts down.
- ▶ The trace.log contains output from a diagnostic trace, if tracing is enabled.
- ▶ The native_stdout.log and the native_stderr.log are used by the operating system to log out of memory exceptions and verbose garbage collection data if verbose garbage collection is enabled.

Configuring JVM logs

The JVM logs can be configured from the administrative console. For both log files, systemOut and systemErr, you can specify the following attributes:

- ▶ The path to their location
- ▶ File formatting (basic or advanced)
- ▶ Log file rotation (by file size or time interval)
- ▶ The maximum number of historical files to store on the file system

To access the JVM logs configuration, from the administrative console, navigate to **Troubleshooting** → **Logs and Trace** → *server_name* → **JVM Logs**.

Enabling verbose garbage collection

Verbose garbage collection is an option provided by the JVM run time. It is often recommended that you have verbose garbage collection enabled permanently in production. The overhead cost is quite small and the benefits of having it enabled when issues happen are considerable.

The verbose garbage collection feature must be enabled for each application server. From the administrative console, navigate to **Servers** → **All servers** → **server_name** → **Server Infrastructure** → **Java and Process Management** → **Process definition** → **Java Virtual Machine** and then select the **Verbose Garbage Collection** option. Save the configurations, synchronize the nodes, and restart the modified application servers.

Example 31-1 shows a snippet of a verbose garbage collection log, after it is enabled for the application server, in the administrative console.

Example 31-1 Snippet of a verbose garbage collection log

```
<exclusive-start id="38" timestamp="2012-06-28T19:29:58.285" intervalms="718.340">
  <response-info timems="0.008" idlems="0.008" threads="0" lastid="0A01CE00"
  lastname="main" />
</exclusive-start>
<af-start id="39" totalBytesRequested="40" timestamp="2012-06-28T19:29:58.285"
intervalms="718.303" />
<cycle-start id="40" type="scavenge" contextid="0"
timestamp="2012-06-28T19:29:58.286" intervalms="718.304" />
<gc-start id="41" type="scavenge" contextid="40"
timestamp="2012-06-28T19:29:58.286">
  <mem-info id="42" free="198926216" total="234881024" percent="84">
    <mem type="nursery" free="0" total="33554432" percent="0" />
    <mem type="tenure" free="198926216" total="201326592" percent="98">
      <mem type="soa" free="188860296" total="191260672" percent="98" />
      <mem type="loa" free="10065920" total="10065920" percent="100" />
    </mem>
    <remembered-set count="26568" />
  </mem-info>
</gc-start>
<allocation-stats totalBytes="22519864" >
  <allocated-bytes non-tilh="839696" tilh="21680168" />
  <largest-consumer threadName="main" threadId="0A01CE00" bytes="22421696" />
</allocation-stats>
<gc-op id="43" type="scavenge" timems="57.326" contextid="40"
timestamp="2012-06-28T19:29:58.343">
  <scavenger-info tenureage="9" tilratio="50" />
  <memory-copied type="nursery" objects="301744" bytes="13055172"
bytesdiscarded="3240" />
  <finalization candidates="654" enqueued="74" />
  <references type="soft" candidates="3390" cleared="0" enqueued="0"
dynamicThreshold="32" maxThreshold="32" />
  <references type="weak" candidates="14601" cleared="0" enqueued="0" />
</gc-op>
<heap-resize id="44" type="expand" space="nursery" amount="3014656" count="1"
timems="0.050" reason="excessive time being spent scavenging"
timestamp="2012-06-28T19:29:58.343" />
<gc-end id="45" type="scavenge" contextid="40" durationms="57.509"
timestamp="2012-06-28T19:29:58.343">
  <mem-info id="46" free="220507040" total="236388352" percent="93">
    <mem type="nursery" free="21580824" total="35061760" percent="61" />
    <mem type="tenure" free="198926216" total="201326592" percent="98">
      <mem type="soa" free="188860296" total="191260672" percent="98" />
      <mem type="loa" free="10065920" total="10065920" percent="100" />
    </mem>
    <pending-finalizers system="73" default="1" reference="0" classloader="0" />
```

```
<remembered-set count="24218" />
</mem-info>
</gc-end>
```

For more information about working with native garbage collection logs, refer to 31.4.3, “Out of Memory exceptions in WebSphere Application Server” on page 1109.

Diagnostic trace

Diagnostic trace provides detailed information about how the application server components run within a managed process. This trace allows administrators to examine processes in the application server and diagnose various issues.

Tracing can be started while the server is running using Runtime Diagnostic Trace or when the server is started using Configuration Diagnostic Trace. The trace outputs can be directed to either a file or a memory buffer.

Tracing has a significant impact on performance and must only be enabled temporarily for problem determination purposes.

To enable diagnostic traces, use the deployment manager’s administrative console, navigating to **Troubleshooting** → **Application servers** → *server_name* → **Diagnostic trace service**.

You can configure the following parameters for diagnostic tracing:

- ▶ Trace Output:
 - None
 - Memory buffer
 - File
- ▶ Trace Output Format:
 - Basic
 - Advanced
 - Log analyzer
- ▶ Change log detail levels. These are used to set up trace strings.

For more information about WebSphere Application Server V8.5 tracing, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/ttrb_trcover.html

31.2.2 JVM log interpretation

The JVM logs contain print data written by applications. They can be written in the following formats:

- ▶ Basic Format: The format used in earlier versions of WebSphere Application Server.
- ▶ Advanced Format: Extends the basic format by adding information about an event, when possible.

Both basic and advanced formats use many of the same fields and formatting techniques. The following is a list of those formats and the various fields associated with those formats:

► **TimeStamp**

The timestamp is formatted using the locale of the process where it is formatted. It includes a fully-qualified date (for example YYYYMMDD), 24 hour time with millisecond precision, and a time zone.

► **ThreadId**

An eight-character hexadecimal value generated from the hash code of the thread that issued the message.

► **ThreadName**

The name of the Java thread that issued the message or trace event.

► **ShortName**

The abbreviated name of the logging component that issued the message or trace event. This is typically the class name for WebSphere Application Server internal components, but can be some other identifier for user applications.

► **LongName**

The full name of the logging component that issued the message or trace event. This is typically the fully-qualified class name for WebSphere Application Server internal components, but can be some other identifier for user applications.

► **EventType**

A one-character field that indicates the type of message or trace event. Message types are in upper case. Possible values are in Table 31-1.

Table 31-1 Message types

| Message type | Description |
|---------------------|---|
| F | A Fatal message. |
| E | An Error message. |
| W | A Warning message. |
| A | An Audit message. |
| I | An Informational message. |
| C | An Configuration message. |
| D | A Detail message. |
| O | A message that was written directly to System.out by the user application or internal components. |
| R | A message that was written directly to System.err by the user application or internal components. |
| Z | A placeholder to indicate the type was not recognized. |

► **ClassName**

The class that issued the message or trace event.

► **MethodName**

The method that issued the message or trace event.

- ▶ Organization
The organization that owns the application that issued the message or trace event.
- ▶ Product
The product that issued the message or trace event.
- ▶ Component
The component within the product that issued the message or trace event.

The following syntax is an example of the basic format structure, as it appears in a log:

```
<timestamp><threadId><shortName><eventType>[className] [methodName] <message>
```

Example 31-2 shows an output SystemOut.log snippet configured for basic format.

Example 31-2 SystemOut.log with basic file formatting

```
[6/28/12 10:10:27:777 EDT] 00001135 impl          I   CWLRB5604I: [06/28/12
10:10:27:777 EDT] Freeing IVTStep2 batch data stream: inputStream
[6/28/12 10:10:27:778 EDT] 00001135 impl          I   CWLRB5604I: [06/28/12
10:10:27:778 EDT] Freeing IVTStep2 batch data stream: outputStream
[6/28/12 10:10:27:778 EDT] 00001135 impl          I   CWLRB5854I: [06/28/12
10:10:27:778 EDT] Job Step [XDCGIVT:00006,IVTStep2]: Metric = clock Value = 00:0
0:00:006
[6/28/12 10:10:27:779 EDT] 00001135 impl          I   CWLRB5854I: [06/28/12
10:10:27:779 EDT] Job Step [XDCGIVT:00006,IVTStep2]: Metric = retry Value = 0
[6/28/12 10:10:27:780 EDT] 00001135 impl          I   CWLRB5844I: [06/28/12
10:10:27:780 EDT] Job Step Batch Data Stream [XDCGIVT:00006,IVTStep2,outputStream
]: Metric = skip Value = 0
```

The following syntax is an example of the advanced format structure, as it appears in a log:

```
<timestamp><threadId><eventType><UOW><source=longName>[className] [methodName] <Orga
nization><Product><Component>[thread=threadName] <message>
```

Example 31-3 shows an output SystemOut.log snippet configured for advanced format.

Example 31-3 SystemOut.log with advanced file formatting

```
[6/29/12 16:32:31:318 EDT] 00000001 I UOW=null source=com.ibm.ws.config.ModelMgr
org=IBM prod=WebSphere component=Application Server thread=[main]
      WSVR0801I: Initializing all server configuration models
[6/29/12 16:32:34:391 EDT] 00000001 I UOW=null
source=com.ibm.ws.xd.agent.websphere.runtime.component.XDAComponentImpl org=IBM
prod=WebSphere component=Application Server thread=[main]
      CWXDA0001I: XDA service XDAComponentImpl initialized successfully.
[6/29/12 16:32:34:449 EDT] 00000001 I UOW=null
source=com.ibm.ws.grid.endpointselector.GAPAgentComponent org=IBM prod=WebSphere
component=Application Server thread=[main]
      CWLRS6000I: GAP (Grid Application Placement) Component has initialized
successfully on process ManagedProcess.
[6/29/12 16:32:34:513 EDT] 00000001 I UOW=null
source=com.ibm.ws.sib.utils.ras.SibMessage org=IBM prod=WebSphere
component=Application Server thread=[main]
      [:] CWSIU0000I: Release: WAS85.SIB Level: gm1216.02
```

31.2.3 Logging modes

Two modes of logging and tracing exist in WebSphere Application Server V8.5:

- ▶ Basic mode

The default mode, is the existing log and trace framework from prior releases of WebSphere Application Server.

- ▶ High Performance Extensible Logging (HPEL) mode

This is a new log and trace framework. HPEL mode must be explicitly enabled. After HPEL mode is enabled, the JVM logs (typically SystemOut.log and SystemErr.log), the trace log (typically trace.log), and the service log (typically activity.log) are no longer written to. Instead, log and trace content is written to a log data or trace data repository in a proprietary binary format. If configured to do so, this content can also be written to a text log file. Text log file names have the following format: TextLog_<yy.mm.dd>_<hh.mm.ss>, where "TextLog_" is a fixed prefix, <yy.mm.dd> is a date (year, month, date) of the first record in the file, and <hh.mm.ss> is the time (hour, minute, second).

Example: TextLog_12.07.16_09.02.09.log

Disabling the writing of this same text log file results in the largest possible performance benefit of HPEL. A log viewing tool, LogViewer, is provided to allow for viewing, filtering, monitoring, and formatting the log and trace data in the repositories.

Figure 31-1 shows the files used by the basic mode and HPEL mode log and trace facilities. When enabled, the HPEL text log file stores content from Java trace (optional), Java logs, System.out, and System.err. You can disable the HPEL text log in cases where it is not needed as indicated by the dotted lines.

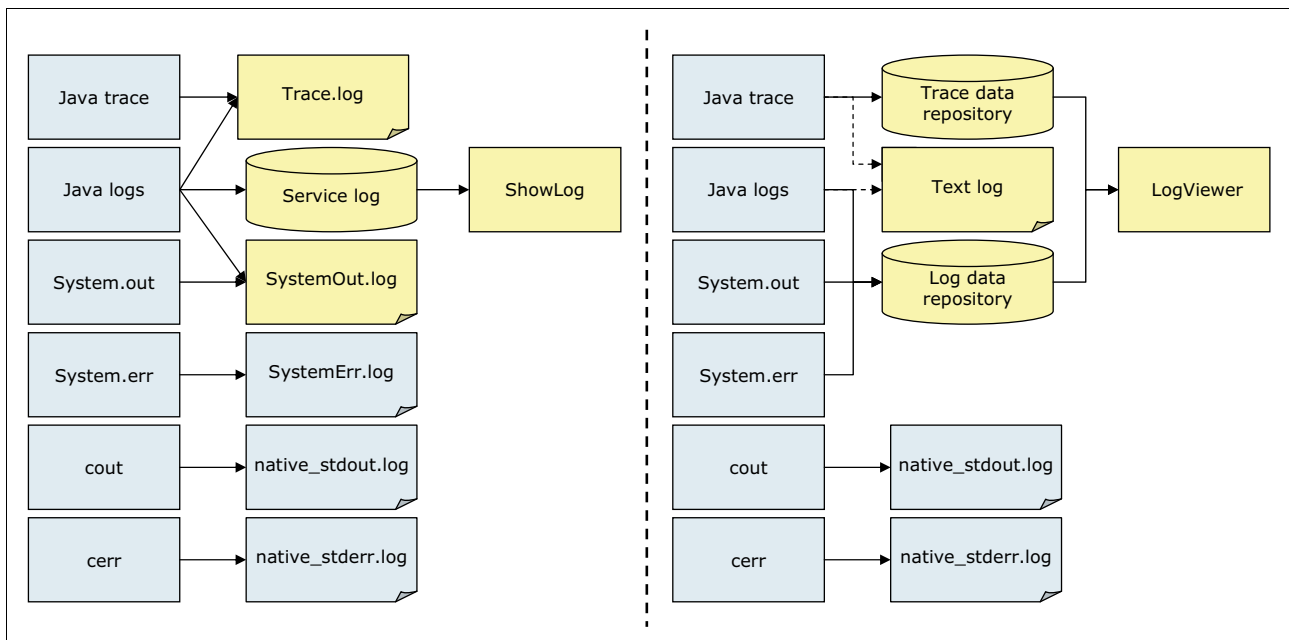


Figure 31-1 Basic mode and HPEL mode log and trace

For more information about WebSphere Application Server V8.5 logging and trace modes, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/c_trb_HPELCompat.html

31.2.4 High Performance Extensible Logging

Performance Extensible Logging (HPEL) is a log and trace facility that is provided as a part of WebSphere Application Server V8.5.

HPEL provides a convenient mechanism for storing and accessing log, trace, System.err, and System.out information produced by the application server or your applications. It is an alternative to the basic log and trace facility, which provided the JVM logs, diagnostic trace, and service log files commonly named SystemOut.log/SystemErr.log, trace.log, and activity.log.

HPEL provides a log data repository, a trace data repository, and a text log file, as shown in Figure 31-2.

Note: All the data that is written to log and trace repositories are parsed and formatted to be stored in a text log file. For this reason, consider disabling the log file as soon as possible to enhance server performance.

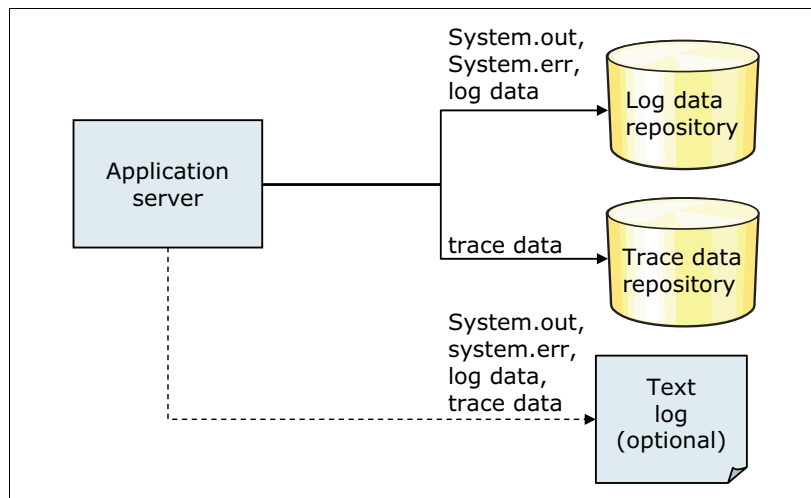


Figure 31-2 HPEL repositories

HPEL log data repository

The log data repository is a storage facility for log records. Log data is typically intended to be reviewed by administrators. This includes any information applications or the server write to System.out, System.err, or java.util.logging at level Detail or higher (including Detail, Config, Info, Audit, Warning, Severe, Fatal, and some custom levels).

HPEL trace data repository

The trace data repository is a storage facility for trace records. Trace data is typically intended for use by application programmers or by the WebSphere Application Server support team. Trace records include any information applications or the server that can write to java.util.logging at levels below level Detail (including Fine, Finer, Finest, and some custom levels).

HPEL text log

The text log file is a plain text file for log and trace records. The text log file is provided for convenience. Primarily so that log content can be read without having to run the LogViewer command-line tool to convert the log data repository content to plain text.

The text log file does not contain any content that is not also stored in either the log data repository or trace data repository. You can disable the text log to enhance server performance. The text log can be configured to record trace content for debugging convenience.

Log and trace performance

HPEL was designed and tested to significantly outperform the existing basic log and trace facility. One result is that the application server can run with trace enabled yet causing less impact to performance than tracing the same components using basic logging. Another result is that applications that frequently write to the logs might run faster with HPEL. A number of factors contribute to the overall performance of HPEL logging and tracing.

Log and trace repositories are not shared across processes

Synchronizing activities between processes causes a degradation in performance to all processes involved. With HPEL, each server process has its own log data repository, trace data repository, and text log file. Because these files are not shared across processes, the server runtime environment does not need to synchronize with other processes when writing to these destinations.

Log and trace data is buffered before being written to disk

Writing large blocks of data to a disk is more efficient than writing the same amount of data in small blocks. HPEL provides buffer log and trace data before writing it to disk. By default, log and trace data is stored in an 8 KB buffer before being written to disk. If the buffer is filled within 10 seconds, the buffer is written to disk. If the buffer is not filled within that time it is automatically written to disk to ensure that the logs have the most current information.

Administration of log and trace

HPEL has been designed to be easy to configure and understand. For example, administrators can easily configure how much disk space to dedicate to logs or trace, or how long to retain log and trace records, and leave the management of log and trace content up to the server. As another example all log, trace, System.out, and System.err content can be accessed using one easy-to-use command (LogViewer), avoiding any possible confusion over which file to access for certain content.

Enabling and disabling HPEL

To activate HPEL logging and tracing, in the administrative console, click **Troubleshooting** → **Logs and trace** → **<your_server>** → **Switch to HPEL Mode**, as seen in Figure 31-3 on page 1085.

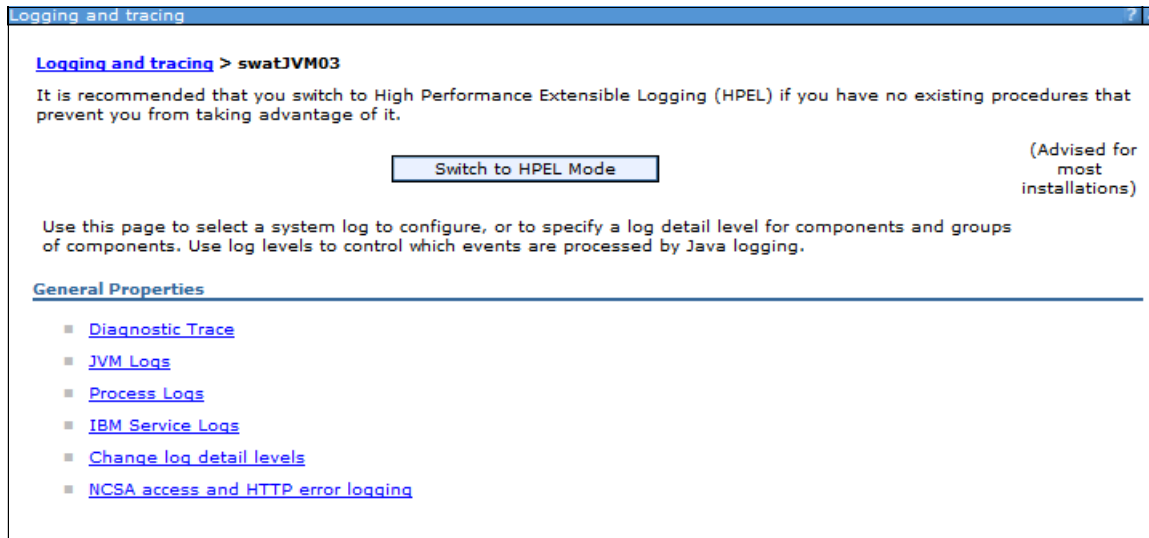


Figure 31-3 Enabling HPEL

To disable HPEL logging and tracing, in the administrative console, click **Troubleshooting** → **Logs and trace** → **<your_server>** → **Change log and trace mode** and click **Switch to Basic Mode**.

Note: A server restart is needed after enabling or disabling HPEL logging and tracing.

HPEL log and trace settings

High Performance Extensible Logging (HPEL) log settings can only be accessed when the server is configured to use HPEL log and trace mode.

To configure HPEL log settings, from the administrative console, click **Troubleshooting** → **Logs and trace** → **server_name** → **Configure HPEL logging**. At this point, the following options will be available for configuration:

- ▶ **Directory path**
Specifies the directory to which log files are written.
- ▶ **Enable log record buffering**
Specifies that the logging system avoids writing to disk each time a log record is created. The logging system creates a buffer that can hold a number of log records, and writes the buffered events when the buffer is full. The logging system also writes the buffered events after a few seconds have passed, even if the buffer is not full.
- ▶ **Start new log file daily at: Time**
Enables the logging framework to close the log file and start a new file at the specified time of day.
- ▶ **Begin cleanup of oldest records**
Specifies the log cleanup settings to be used to automatically purge the oldest log records, or log records that no longer fit in the configured space, from the log repository.
- ▶ **Log record age limit**
Specifies the lifespan, in hours, that log records can remain in the log repository before the log records can be automatically deleted by the server.

- ▶ **Maximum log size**
Specifies the maximum total size, in megabytes, that the server allows the log repository to reach. When the log repository approaches this size limit, the server deletes the oldest records from the log repository to make space for new log records.
- ▶ **Out of space action**
Specifies how the server reacts to an inability to add content to the log repository:
 - **Stop server:** Specify that the server stops when the server is unable to write to the log repository.
 - **Purge old records:** Specify that the server continues to run, and that the oldest log records are immediately removed when the server is unable to write to the log repository.
 - **Stop logging:** Specify that the server continues to run, but that the server cannot continue to write to the log when the server is unable to write to the log repository.

Figure 31-4 shows the options previously listed and available under HPEL log configuration.

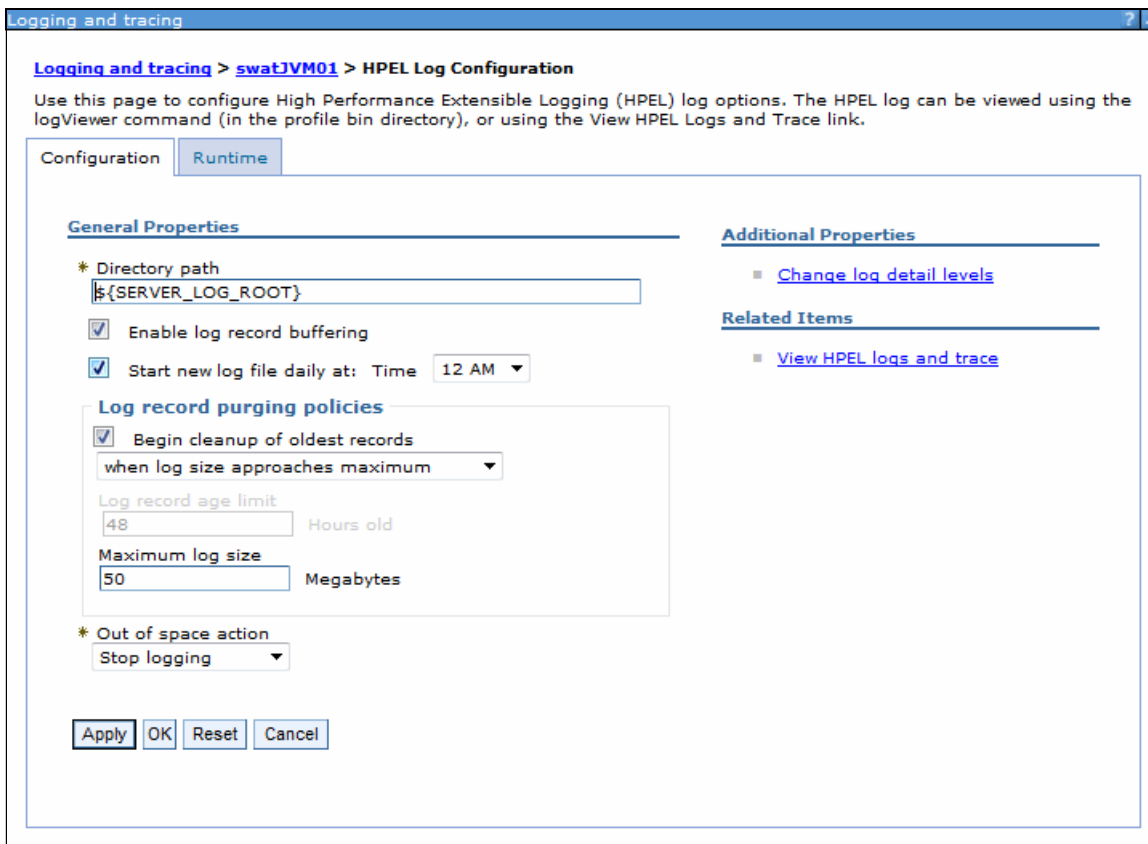


Figure 31-4 HPEL log configuration

To configure HPEL logging and trace settings, from the administrative console, click **Troubleshooting** → **Logs and trace** → **server_name** → **Configure HPEL trace**. The following options are available under Configure HPEL trace:

- ▶ **Choosing the HPEL output format:**
 - **Trace to a directory**
The same set of options that were described for configuring HPEL logging, are applicable to HPEL trace settings when this option is selected.

- Trace to a memory buffer

Specify the memory buffer size in megabytes (MB) and the directory to use for dumping the memory buffer.

To configure HPEL text log configuration settings, from the administrative console, click **Troubleshooting** → **Logs and trace** → *server_name* → **Configure HPEL text log**. The following options are available under Configure HPEL text log:

- ▶ **Enable Text Log**

Specifies that in addition to writing log and trace records in binary format, the logging system writes them in a text format as well:

- When this option is enabled, the same set of options that were described for configuring HPEL logging (page 1085), are applicable to HPEL trace settings when this option is selected.

- **Text Output Format**

Specifies the format to use in the text log file:

- Select Basic to specify a shorter, one-line-per-record format.
- Select Advanced to specify a longer format using full logger name and more details about each record.

- **Include trace records**

Specifies whether trace records are included in the text log file and log records.

Reading from the log data and trace data repositories

Now the data is stored in proprietary binary format instead of text as in basic logging (the only exception is for the text log repository). In this way, the following benefits are realized:

- ▶ There is no more text parsing.
- ▶ More data is available because truncation is not necessary.
- ▶ Data is not formatted unless it is needed.
- ▶ No need to clear log files before server start, for example, to diagnosis a problem.
- ▶ Trace speed is improved and more data can be available, and it has half of the impact that basic tracing has.
- ▶ Provides a common solution between z/OS and distributed platform.
- ▶ Applications running with HPEL run faster than running the same application with basic logging.

To read the log and trace records in this new format, a new command line was introduced called **logViewer**. It reads the data from repositories, formats it, and displays it to the administrator, as shown in Figure 31-5 on page 1088.

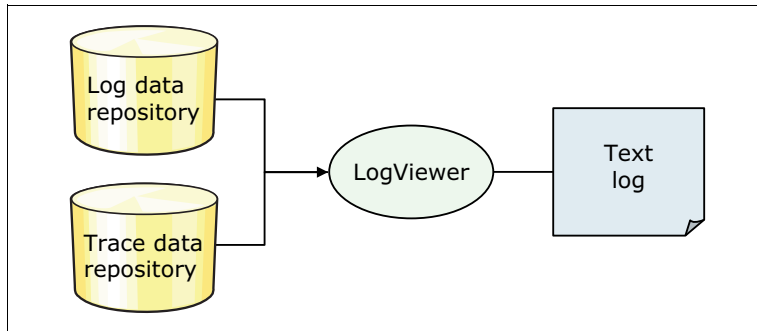


Figure 31-5 logViewer command

Viewing data from HPEL repositories

After configuring HPEL, the data can be viewed using the **logViewer** command from the `bin` subdirectory of the profile, as shown in Example 31-4.

Example 31-4 logViewer output

```

...
[6/28/12 15:27:30:618 EDT] 0000003e CoordinatorIm I   HMGR0228I: The Coordinator
is not an Active Coordinator for core group DefaultCoreGroup. The active
coordinator set is [saw211-RHEL3Cell01\saw211-RHEL3CellManager01\dmgr].
[6/28/12 15:27:30:620 EDT] 0000003e CoordinatorIm I   HMGR0218I: A new core group
view has been installed. The core group is DefaultCoreGroup. The view identifier
is (185:0.saw211-RHEL3Cell01\saw211-RHEL3CellManager01\dmgr). The number of
members in the new view is 7.
[6/28/12 15:27:30:621 EDT] 0000003e CoreGroupMemb I   DCSV8050I: DCS Stack
DefaultCoreGroup at Member saw211-RHEL3Cell01\saw211-RHEL3Node01\swatJVM01: New
view installed, identifier
(185:0.saw211-RHEL3Cell01\saw211-RHEL3CellManager01\dmgr), view size is 7 (AV=7,
CD=7, CN=7, DF=8)
[6/28/12 15:27:30:692 EDT] 0000004e ViewReceiver I   DCSV1033I: DCS Stack
DefaultCoreGroup at Member saw211-RHEL3Cell01\saw211-RHEL3Node01\swatJVM01:
Confirmed all new view members in view identifier
(185:0.saw211-RHEL3Cell01\saw211-RHEL3CellManager01\dmgr). View channel type is
View|Ptp.
[6/28/12 15:27:35:992 EDT] 00000058 NGUtil$Server I   ASND0002I: Detected server
swatJVM02 started on node saw211-RHEL4Node01
[6/28/12 15:27:38:066 EDT] 00000091 ServerStopped I   ODCF0009I: Marked server
/cell/saw211-RHEL3Cell01/node/saw211-RHEL4Node01/server/swatJVM02 STARTED due to
transaction: remote - initial contribution from
saw211-RHEL3Cell01\saw211-RHEL4Node01\swatJVM02
Operation Complete
Processed 346 records in 0.229 seconds (1,510.917 records per second).
  
```

Example 31-4 showed the messages from the most recent server run, obtained by issuing the following command on a Linux test environment:

```
<WAS_PROFILE_ROOT>/bin/logViewer.sh -latestInstance
```


There are several options to be used with the **logViewer** command, making it a powerful tool to filter events from log and trace repositories. The following list demonstrates some possibilities that can be used:

- ▶ Showing messages starting at a specific level or higher:
`logViewer -minlevel <message_level>`
- ▶ Showing messages from log and trace for a specific thread:
`logViewer -Thread <thread_id>`
- ▶ Showing messages from log and trace in advanced format:
`logViewer -format advanced`

Example 31-5 shows the advanced format view.

Example 31-5 Advanced format view

```
...
[6/28/12 15:27:19:554 EDT] 00000062 I UOW= source=com.ibm.son.mesh.Peer org=IBM
prod=WebSphere component=Application Server thread=[sonInThreadPool : 3]
      ODCF8532I: Added neighbor ip=9.42.171.209 udp=11008 tcp=11010
ID=e2c0f4018a2ee0d59673e03a38768e08b18444e0
version=0;cellName=saw211-RHEL3Cell01;bridgedCells=[];properties={MEMBER_STARTUP_T
IME=1340911627497, MEMBER_VERSION=4,
memberName=saw211-RHEL3Cell01\saw211-RHEL4Node01\swatJVM02, epoch=1340911630430,
inOdc=1, ODC_PUBLISHER_ONLY=false}, neighbor set is now 6 nodes
0 ip=9.42.171.209 udp=11008 tcp=11010 ID=e2c0f4018a2ee0d59673e03a38768e08b18444e0
version=0;cellName=saw211-RHEL3Cell01;bridgedCells=[];properties={MEMBER_STARTUP_T
IME=1340911627497, MEMBER_VERSION=4,
memberName=saw211-RHEL3Cell01\saw211-RHEL4Node01\swatJVM02, epoch=1340911630430,
inOdc=1, ODC_PUBLISHER_ONLY=false}
1 ip=9.42.170.178 udp=11008 tcp=11010 ID=888ac4b6fef1afa4e7e79f786c1d70f4fcc9b1a1
version=0;cellName=saw211-RHEL3Cell01;bridgedCells=[];properties={MEMBER_STARTUP_T
IME=1340911443846, MEMBER_VERSION=4,
memberName=saw211-RHEL3Cell01\saw211-RHEL5Node01\swatJVM03, epoch=1340911448237,
inOdc=1, ODC_PUBLISHER_ONLY=false}
2 ip=9.42.171.242 udp=11003 tcp=11004 ID=8ac1cdb57102f65d8b662a66e7af796010f62bcf
version=0;cellName=saw211-RHEL3Cell01;bridgedCells=[];properties={MEMBER_VERSION=4
, MEMBER_STARTUP_TIME=1340844138854,
memberName=saw211-RHEL3Cell01\saw211-RHEL3Node01\nodeagent, inOdc=1,
epoch=1340844140738, ODC_PUBLISHER_ONLY=false}
3 ip=9.42.171.209 udp=11003 tcp=11004 ID=6e2777bc38d54873c49bce229a574ac0d4e32305
version=0;cellName=saw211-RHEL3Cell01;bridgedCells=[];properties={MEMBER_STARTUP_T
IME=1340832422435, MEMBER_VERSION=4,
memberName=saw211-RHEL3Cell01\saw211-RHEL4Node01\nodeagent, epoch=1340832424261,
inOdc=1, ODC_PUBLISHER_ONLY=false}
4 ip=9.42.170.178 udp=11003 tcp=11004 ID=3fc77268a35c9a1bf21d154aef896b48cb902cee
version=0;cellName=saw211-RHEL3Cell01;bridgedCells=[];properties={MEMBER_STARTUP_T
IME=1340832433243, MEMBER_VERSION=4,
memberName=saw211-RHEL3Cell01\saw211-RHEL5Node01\nodeagent, epoch=1340832435110,
inOdc=1, ODC_PUBLISHER_ONLY=false}
5 ip=9.42.171.242 udp=11005 tcp=11006 ID=fb3955437989332fbc4478b7aba8bfd84b191a52
version=0;cellName=saw211-RHEL3Cell01;bridgedCells=[];properties={MEMBER_VERSION=4
, MEMBER_STARTUP_TIME=1340827441504,
memberName=saw211-RHEL3Cell01\saw211-RHEL3CellManager01\dmgr, inOdc=1,
epoch=1340827464976, ODC_PUBLISHER_ONLY=false}.
...
```

```
[6/28/12 15:27:35:992 EDT] 00000058 I UOW=
source=com.ibm.ws.xd.nodedetect.NGUtil$ServerStatusBBSsubscriber org=IBM
prod=WebSphere component=Application Server thread=[BBSon : 0]
ASND0002I: Detected server swatJVM02 started on node saw211-RHEL4Node01
[6/28/12 15:27:38:066 EDT] 00000091 I UOW=
source=com.ibm.ws.odc.nd.ServerStoppedListener org=IBM prod=WebSphere
component=Application Server thread=[Thread-61]
ODCF0009I: Marked server
/cell/saw211-RHEL3Cell01/node/saw211-RHEL4Node01/server/swatJVM02 STARTED due to
transaction: remote - initial contribution from
saw211-RHEL3Cell01\saw211-RHEL4Node01\swatJVM02
Operation Complete
Processed 346 records in 0.227 seconds (1,524.229 records per second).
```

- Showing messages from log and trace from a specific time range:

```
logViewer -startDate <date/time/timezone> -stopDate <date/time/timezone>
```

More information about the **logViewer.sh** command is available at the following website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/trb_logviewer.html

Another way to monitor logs and traces is by using the WebSphere console. Click **Troubleshooting** → **Logs and trace** → **<your_server>** → **View HPEL logs and trace**. Figure 31-6 on page 1091 shows the result.

Logging and tracing > [swatJVM03](#) > Log Viewer

Use this page to view log data from the HPEL repository (group of common binary log files). You can also use this page to filter and search the repository. You can export the custom repository into a compressed file.

Content and Filtering Details

Refresh View Show Only Selected Threads Show All Threads Select Columns ... Export ... Copy to Clipboard Server Instance Info

Viewing log records from server instance June 28, 2012 15:24:00

Number of records to show: First Page Previous Page Next Page

| TimeStamp | Thread ID | Logger | Level | Message |
|----------------------|-----------|--------------|-------|---|
| 6/28/12 15:24:00.848 | 0000000 | gerAdmin | INFC | TRAS0017I : The startup trace state is *=info. |
| 6/28/12 15:24:00.855 | 0000000 | gerAdmin | INFC | TRAS0111I : The message IDs that are in use are deprecated |
| 6/28/12 15:24:00.880 | 0000000 | lerTracker | INFC | com.ibm.ffdc.osgi.ProviderTracker AddingService FFDC1007I : FFDC Provider Installed: com.ibm.ffdc.util.provider.FfdcOnDirProvider@26c184b4 |
| 6/28/12 15:24:00.939 | 0000000 | .ModelMgr | INFC | WSVR0800I : Initializing core configuration models |
| 6/28/12 15:24:01.494 | 0000000 | :aDataMgr | INFC | WSVR0179I : The runtime provisioning feature is disabled. All components will be started. |
| 6/28/12 15:24:01.526 | 0000000 | :aDataMgr | WAR | WSVR0174W : A duplicate component has been ignored. Ignore the WS_JaxWsCommonContainer com.ibm.ws.container.binding.ws.impl.WSEndPointManagerComponentImpl [8825] [processtypes:Adjunct] component for the com.ibm.wsspi.extension.applicationserver-startup extension point within the com.ibm.ws.soa.sca.container.ws bundle. |
| 6/28/12 15:24:01.606 | 0000000 | lerTracker | INFC | com.ibm.ffdc.osgi.ProviderTracker AddingService FFDC1007I : FFDC Provider Installed: com.ibm.ws.ffdc.impl.FfdcProvider@d3f8ee33 |
| 6/28/12 15:24:01.786 | 0000000 | :Initializer | AUD | ADMN0015I : The administration service is initialized. |
| 6/28/12 15:24:02.273 | 0000000 | :erviceImpl | INFC | PLGC0057I : The plug-in configuration service started successfully. |
| 6/28/12 15:24:02.297 | 0000000 | onentImpl | INFC | CWPKI0001I : SSL service is initializing the configuration |
| 6/28/12 15:24:02.303 | 0000000 | :SManager | INFC | CWPKI0044I : FIPS security mode is : No FIPS property found. |
| 6/28/12 15:24:02.306 | 0000000 | :SKeyStore | WAR | CWPKI0041W : One or more key stores are using the default password. |
| 6/28/12 15:24:02.321 | 0000000 | igManager | INFC | CWPKI0027I : Disabling default hostname verification for HTTPS URL connections. |
| 6/28/12 15:24:02.397 | 0000000 | :ticModule | INFC | CWPKI0014I : The SSL component's FFDC Diagnostic Module com.ibm.ws.ssl.core.SSLDiagnosticModule registered successfully: true. |
| 6/28/12 15:24:02.397 | 0000000 | onentImpl | INFC | CWPKI0002I : SSL service initialization completed successfully |
| 6/28/12 15:24:02.402 | 0000000 | onfigHome | INFC | com.ibm.wsspi.rasdiag.DiagnosticConfigHome setStateCollectionSpec RASD0012I : Updating State Collection Spec from Uninitialized Value to *, *=0 |
| 6/28/12 15:24:02.416 | 0000000 | :c.PMIImpl | AUD | CWPMI1001I : PMI is enabled |
| 6/28/12 15:24:02.754 | 0000000 | .ModelMgr | INFC | WSVR0801I : Initializing all server configuration models |

Figure 31-6 HPEL log and trace viewed in WebSphere console

For more information about WebSphere Application Server V8.5 HPEL, refer to the following web sites:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/trb_usinghpel.html

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/trb_logtracehpel.html

http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/topic/com.ibm.iea.was_v8/was/8.0/ProblemDetermination.html?dmuid=20110621121814592967

31.2.5 Cross Component Trace

Cross Component Trace (XCT) annotates the logs so that log entries that are related to a request that is serviced by more than one thread, process, or even server are identified as belonging to the same unit of work.

XCT helps identify the root cause of problems across components, which provides the following benefits:

- ▶ Enables administrators and support teams to follow the flow of a request from end-to-end. XCT follows as a request traverses thread or process boundaries, travels between stack products, and the WebSphere Application Server.
- ▶ Helps to resolve questions about which component is responsible for a request that fails.

Administration of XCT

XCT is a function built into the WebSphere Application Server log and trace framework. When enabled, XCT annotates the logs so that log entries that are related to a request that is serviced by more than one thread, process, or even server are identified as belonging to the same unit of work. Applications built using distributed architectures, such as Service Oriented Architecture, will benefit from XCT. XCT helps facilitate problem determination across multiple services on different systems.

The following list identifies the different XCT modes:

- ▶ Fully disabled.
- ▶ With XCT request IDs added to existing log and trace records.
- ▶ With XCT request IDs added to existing log and trace records and XCT log records added to log files.
- ▶ With XCT request IDs added to existing log and trace records, XCT log records added to log files, and data snapshots captured.

The following list identifies the various XCT request IDs:

- ▶ XCT request IDs are identifiers added to log and trace records. These records are produced by the server when the server is configured to use High Performance Extensible Logging (HPEL).
- ▶ XCT adds the same request ID to every log or trace record as long as the log or trace record is a part of the same request. The request ID is added regardless of which thread or Java virtual machine (JVM) produces the log or trace entry.
- ▶ When XCT is used with the HPEL log and trace infrastructure, you can view request IDs with the logViewer tool when logs are output in advanced format.

Example 31-6 shows a log record with an XCT request ID in the log file (shown rendered in advanced format).

Example 31-6 Log record with an XCT request ID

```
[3/18/11 14:50:17:391 EDT] 00000018 W U0W= source=com.ibm.somelogger.QuickLogTest  
org= prod= component= thread=[WebContainer : 1] requestID=BJrcVPo+Yk4-AAAAAA8zAA  
hello world
```

Note that the request ID is shown previously on a separate line, but in the log files it is actually on the same line as the rest of the log record header.

The following list identifies qualities about the XCT log records:

- ▶ XCT log records are typically added to the logs to:
 - Demarcate the beginning and ending of work for a particular request on a particular thread.
 - Demarcate when work is about to be transferred to another thread or process, or to indicate when work returned from another thread or process.

- Demarcate when work moves from major component to major component, even if work continues on the same thread. For example, to show transfer of control from application server code to application code.

Example 31-7 is example of a XCT log record in the log file.

Example 31-7 XCT log record in the log file

```
[3/18/11 14:50:17:391 EDT] 00000031 XCT I BEGIN BJrcVPo+Yk4-AAAAAA8zAA
0000000000-ccccccccc2 HTTPCF(OutboundRequest /index.html
RemoteAddress(127.0.0.1) RequestContext(36001645))
```

- ▶ XCT log records are composed of:
 - XCT type (BEGIN / END)
 - XCT parent correlator ID (for example, 0000000000-ccccccccc2)
 - XCT current correlator ID (for example, BJrcVPo+Yk4-AAAAAA8zAA)
 - XCT annotations (for example, HTTPCF(OutboundRequest /index.html RemoteAddress(127.0.0.1) RequestContext(36001645))

The following list identifies XCT tools:

- ▶ The HPEL logViewer tool is able to filter log and trace records by request ID.
- ▶ Tools, such as the XCT Log Viewer, can also take advantage of XCT log records or XCT request IDs, or both, when rendering log and trace content. The XCT Log Viewer is available as a tool add-on for the IBM Support Assistant.

Using Cross-Component Trace to troubleshoot applications

Administrators using WebSphere Application Server need to use log and trace files to determine whether their applications and the server are running correctly.

Depending on the nature of your applications, multiple threads within an application server might be used to handle requests, such as HTTP requests or JMS requests. Some requests might be handled by more than one application server, such as when one application server makes a request to another application server for a web services request.

You can use XCT to augment your log and trace files with correlation information. This correlation information clarifies which threads and which application server processes participated in the handling of each request.

To enable XCT, from the administrative console, click **Troubleshooting** → **Logs and trace** → **server_name** → **Change log detail levels**. The following options are available under Change log detail levels:

- ▶ Disable logging and tracing of potentially sensitive data

The application server has a list of loggers which are known to potentially write sensitive information to the log and trace when enabled. For example, enabling certain HTTP related loggers at FINEST level can result in confidential user-specified information from HTTP requests being stored in the trace files. If you want the server to avoid enabling these loggers at levels which are known to be used for potentially sensitive information, check the **Disable logging and tracing of potentially sensitive data** option. When the server is started, or when the log detail level specification is modified at run time, the server will compare the list of loggers and levels specified in the log detail level specification to the list of loggers and levels in the sensitive logger list. At that point, the server will update the log detail level specification as needed.

► Change Log Detail Levels

Enter a log detail level that specifies the components, packages, or groups to trace.

If you select the **Runtime tab**, and expand **Components and Groups**, the list of components, packages, and groups are displayed. This display contains all the components that are registered on the running application server and in the static list.

► Enable log and trace correlation

Enables or disables log and trace correlation to be tracked for the application server.

Options include:

- Include request IDs in log and trace records

Enable Cross Component Trace (XCT) to include request IDs in log and trace files. This enables you to see which log and trace entries, in all threads and application server processes, are related to the same request.

- Include request IDs in log and trace records and correlation log records

Enable XCT to create correlation log records when you want to log how requests branch between threads and processes. You can also see extra information about each request.

- Include request IDs in log and trace records, create correlation log records, and capture data snapshots as appropriate.

Enable XCT to capture data snapshots when you want to store entire request and response bodies to the file system.

Figure 31-7 on page 1095 shows the previously listed functions under the Change log details page.

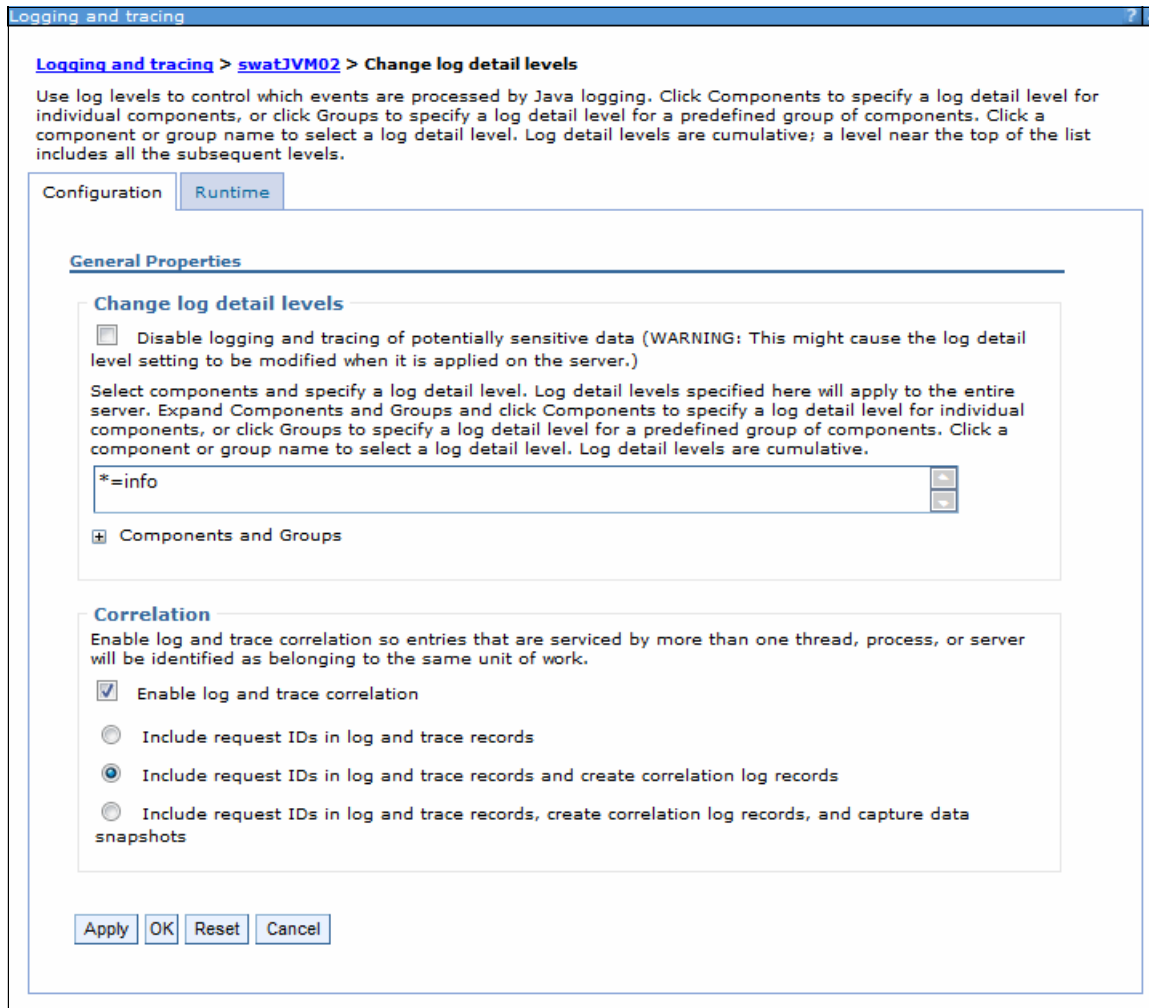


Figure 31-7 Change log detail levels and enable XCT

After XCT is enabled, you can use XCT request ID information to track requests.

Ensure you are using High Performance Extensible Logging (HPEL) log and trace mode, as basic mode log and trace does not store request IDs.

Filter your logs to look for important information, such as errors or warnings using the HPEL LogViewer command-line tool. Output your logs using advanced format so that you can see the request ID information in the logs. For example, use the following command (on Linux):

```
<WAS_PROFILE_ROOT>/bin/logViewer.sh -minLevel WARNING -format advanced
```

When you find the log entries that are of interest to you, note the request ID associated with those entries.

Filter your logs by request ID using the HPEL LogViewer command-line tool and using the request IDs you noted in the previous step:

```
<WAS_PROFILE_ROOT>/bin/logViewer.sh -includeExtensions -requestID=<requestID>
```

31.2.6 Sensitive log and trace guard

The sensitive log and trace guard is a feature that helps administrators prevent sensitive information from being exposed in log and trace files.

The sensitive log and trace guard uses an internal list of allowable levels for sensitive loggers. It specifies the lowest level at which listed loggers can generate log or trace data without containing potentially sensitive data. You can also add your own loggers to the list that the sensitive log and trace guard will block.

To enable the sensitive log and trace guard feature, using the deployment manager's administrative console, navigate to **Troubleshooting** → **Logs and trace** → **server_name**.

Click **Change log detail levels**, and select the **Disable logging and tracing of potentially sensitive data** option to enable sensitive log and trace guard.

For more information about this feature, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.doc/ae/containter_hpel_sensitiveguard.html

31.2.7 Javacores and Heapdumps

You can use the Java runtime environment to create dump and core files to help with troubleshooting. You can use the administrative console to trigger the creation of these dumps and core files.

The Java virtual machine (JVM) is capable of producing Java dump and core files to aid in troubleshooting. You can use heap dump and system dump files to help you diagnose memory-related problems, such as memory leaks. Use Java core files to help you diagnose problems where the CPU is persistently 100% busy, when threads are hanging, or where threads are in a deadlock.

Note: The process of generating dump and core files can have a noticeable performance impact on your system that can last for many seconds or minutes. If necessary, use your test and development environments to better understand the impact of generating dump and core files.

The following procedure describes how you use the deployment manager's administrative console to generate javacores and heapdumps:

1. In the navigation pane, click **Troubleshooting** → **Java dumps and cores**.
2. Select the server or servers you need to collect a dump or core from.
3. Click **System Dump**, **Java Core**, or **Heap Dump** depending on your need.

The system dump, heap dump, or Java core is created and stored in the profile root directory of the server from which you requested the dump or core.

Note: Generating a heap dump or a system dump is not supported for a non-IBM JVM.

31.2.8 HTTP Plug-in Log

The HTTP plug-in records messages to a log file that is stored under the WebSphere HTTP Plug-in directory on the Web server machine. The following syntax shows the default location:

```
<plugin_root>/logs/<web_server_name>/http_plugin.log
```

The log level can be set to different values depending on the amount of detail that you want written to the log. Setting the log level to *Trace* causes all steps in the HTTP request process to be logged.

Note: Setting the plug-in log level to Trace produces a lot of output and can cause an impact in server performance. Input/output happens very frequently and can also cause the plug-in log file to grow quickly, even causing file truncation.

For more information about the WebSphere HTTP Plug-in log, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.base.doc/ae/rwsv_plugincfg.html

31.3 Tools for collecting and analyzing diagnostic data

WebSphere Application Server provides features to assist in the detection and therefore in the capture of information for problem determination. Moreover, IBM has a comprehensive set of instructions on how to gather data for nearly all types of issues that can impact a WebSphere environment. IBM provides the IBM Support Assistant, a tool with great value for problem determination analysis.

This section gives an overview on these features and procedures for collecting and analyzing diagnostic data.

31.3.1 Hang detection policy

The *hang detection* option for WebSphere Application Server is turned on by default. You can configure a hang detection policy to accommodate your applications and environment so that potential hangs can be reported. This provides earlier detection of failing servers. When a hung thread is detected, WebSphere Application Server notifies you so that you can troubleshoot the problem.

A common error in Java Platform, Enterprise Edition (Java EE) applications is a hung thread. A *hung thread* can result from a simple software defect, such as an infinite loop, or a more complex cause, such as a resource deadlock. System resources, such as CPU time, might be consumed by this hung transaction when threads run unbounded code paths, such as when the code is running in an infinite loop. Alternately, a system can become unresponsive even though all resources are idle, as in a deadlock scenario. Unless a user or a monitoring tool reports the problem, the system might remain in this degraded state indefinitely.

Using the hang detection policy, you can specify a time that is too long for a unit of work to complete. The thread monitor checks all managed threads in the system such as web container threads and object request broker (ORB) threads. Unmanaged threads, which are threads created by applications, that are not monitored.

When WebSphere detects that a thread is active longer than the time defined by the thread monitor threshold, the application server takes the following actions:

- ▶ Logs a warning in the WebSphere Application Server log that indicates the name of the thread that is hung and how long it was already active. The following message is written to the log:

```
[6/29/12 14:39:22:921 EDT] 0000009d ThreadMonitor W   WSVR0605W: Thread
"WebContainer : 1" (00000099) has been active for 139,950 milliseconds and may
be hung.  There is/are 1 thread(s) in total in the server that may be hung.
```

- ▶ Issues a Java Management Extensions (JMX) notification.

This notification enables third-party tools to catch the event and take appropriate action. Some common actions are triggering a JVM thread dump of the server or issuing an electronic page or email.

- ▶ Triggers changes in the performance monitoring infrastructure (PMI) data counters.

These PMI data counters are used by various tools, such as the Tivoli Performance Viewer, to provide performance analysis.

In some situations, false alarms for hung threads can also happen. If the work actually completes, a second set of messages, notifications, and PMI events is produced to identify the false alarm. The following message is written to the log:

```
WSVR0606W: Thread "WebContainer : 1" (00000099) was previously reported to be hung
but has completed.  It was active for approximately 139,950 milliseconds.  There are
0 threads in total in the server that still may be hung.
```

If the thread monitor determines that too many false alarms are issued (determined by the number of pairs of hang and clear messages), it can automatically adjust the threshold.

The thread hang detection option is enabled by default. You can take the following steps to adjust the hang detection policy values or to disable hang detection completely:

1. From the administrative console, click **Servers** → **Application Servers** → *server_name*
2. Under Server Infrastructure, click **Administration** → **Custom Properties**
3. Click **New**.
4. Add the following properties:

- `com.ibm.websphere.threadmonitor.interval`

The frequency, in seconds, at which managed threads in the selected application server are interrogated:

- `com.ibm.websphere.threadmonitor.threshold`

The length of time, in seconds, in which a thread can be active before it is considered hung.

- `com.ibm.websphere.threadmonitor.false.alarm.threshold`

The number of times that false alarms can occur before automatically increasing the threshold.

- `com.ibm.websphere.threadmonitor.dump.java`

Set to true to cause a javacore to be created when a hung thread is detected and a WSVR0605W message is printed.

- `com.ibm.websphere.threadmonitor.dump.stack`

Set to true to cause a stack trace to be printed when a hung thread is detected and a WSVR0605W message is printed.

5. Click **Apply**.
6. Save the changes. Make sure a file synchronization is performed *before* restarting the servers.
7. Restart the Application Server for the changes to take effect.

31.3.2 Memory leak detection policy

The leak detection policy for the WebSphere Application Server is turned off by default. You can configure a leak detection, prevention, and action policy to accommodate your applications and environment so that potential memory leaks are reported and acted upon. Leak detection, prevention, and proactive fixing provides protection and resiliency for servers that face persistent out of memory errors.

Memory leaks come in various types and are noted in the following list:

► **Class loader memory leak**

Many memory leaks manifest themselves as class loader leaks, which are normally caused by the application code or JRE triggered code.

Retaining a reference to a single object from a web application pins every class loaded by the web application. These references often remain after a web application reload. With each reload, more classes are pinned which leads to an out of memory error.

► **JRE triggered leaks**

Memory leaks occur when Java Runtime Environment (JRE) code uses the context class loader to load an application singleton. These singletons can be threads or other objects that are loaded by the JRE using the context class loader.

If the web application code triggers the initialization of a singleton or a static initializer, the following conditions apply:

- The context class loader becomes the web application class loader.
- A reference is created to the web application class loader. This reference is never garbage collected.
- Pins the class loader, and all the classes loaded by it, in memory.

► **Application triggered leaks**

Application code sometimes retains references to objects that are no longer required and used. Therefore, these objects continue to exist and to take up memory space and cannot be cleared by the Garbage Collector.

Categories of application triggered leaks are as follows:

- Custom ThreadLocal class
- Web application class instance as ThreadLocal value
- Web application class instance indirectly held through a ThreadLocal value
- ThreadLocal pseudo-leak
- ContextClassLoader and threads created by web applications
- ContextClassLoader and threads created by classes loaded by the common class loader
- Static class variables
- JDBC driver registration: RMI targets

Approaches to memory leak detection typically involve examination of both Java Virtual Machine Tool Interface (JVMTI) or performance monitoring infrastructure (PMI) counters. These are used to watch for slow growth in Java or native heap usage.

WebSphere Application Server Version 8.5 provides a top down pattern to handle memory leak situations. The application server watches for suspect patterns in application code at run time, and operates in the following manner:

- ▶ Detection

WebSphere Application Server checks for known causes of memory leaks and issues warnings when an application leak is detected.

- ▶ Prevention (on by default)

Applies only to JRE triggered leaks. These leaks are prevented by initializing singletons at server startup, when the application server class loader is the context class loader.

- ▶ Action

Take proactive action to fix memory leaks. These actions have reasonable defaults and are configured on a case-by-case basis.

WebSphere Application Server has some means of protection against memory leaks when stopping or redeploying applications. If leak detection, prevention, and action are enabled, WebSphere Application Server monitors application and module activity, then performs diagnostic actions to detect and fix leaks when an application or an individual module stops. This feature helps in increasing application up time with frequent application redeployments without cycling the server.

Therefore, you can configure WebSphere Application Server to detect, prevent, and take action, if possible, on classloader memory leaks using the memory leak detection policy. If a classloader memory leak is detected, WebSphere Application Server notifies you with informational messages in the log and by taking JVM heapdumps so that you can troubleshoot the problem. Optionally, you might also choose to have WebSphere Application Server mitigate, and if possible, fix the memory leak using reflection and other techniques.

For information about how to configure the memory leak detection policy, refer to the following information center website:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.express.doc/ae/ttrb_configmemleak.html

31.3.3 MustGather for troubleshooting

The IBM mustGather for the WebSphere Application Server website is an essential guide for troubleshooting. In regards to problems with the IBM WebSphere Application Server, it provides a complete set of instructions for collecting troubleshooting data.

You will find instructions for, the following list, about collecting data by component for all WebSphere Application Server releases between 5.1 and 8.5:

- ▶ Classloader
- ▶ Crash
- ▶ DB Connections/Connection Pooling
- ▶ High Availability (HA)
- ▶ Java SDK
- ▶ Performance, hang, or high CPU
- ▶ Out of Memory

The information available in the MustGather website, will help you gather data from problematic environments and help familiarize you with the troubleshooting process.

This process greatly improves the accuracy of data analysis, as it will collect a complete set of information from problematic servers and applications.

Visit (and bookmark) the MustGather for WebSphere Application Server website at following website:

<http://www-01.ibm.com/support/docview.wss?uid=swg21145599>

31.3.4 IBM Support Assistant

IBM Support Assistant is a free troubleshooting application that helps you research, analyze, and resolve problems using various support features and tools. IBM Support Assistant enables you to find solutions yourself using the same troubleshooting techniques used by the IBM Support team. It also allows you to organize and transfer your troubleshooting efforts between members of your team or to IBM for further support.

IBM Support Assistant helps you diagnose problems with applications that are deployed to WebSphere Application Server. You can track and organize troubleshooting activity, automate the collection of MustGather files, and use tools that analyze artifacts to understand the root cause of errors.

The IBM Support Assistant can be downloaded from the following website:

<http://www-01.ibm.com/software/support/isa/#isawb>

You can customize IBM Support Assistant by installing the problem determination tools that you need to troubleshoot various problems. For example, when the JVM runs out of memory, application threads hang, the JVM crashes, or if performance declines, you can use analysis tools to diagnose these problems.

Among the many options available from the IBM Support Assistant, you will find the following tools:

► Java Troubleshooting

Use these diagnostic tools to help troubleshoot problems related to memory management, performance, hangs, crashes and more in your Java applications deployed to WebSphere Application Server:

– IBM Monitoring and Diagnostic Tools for Java - Memory Analyzer

Used for analyzing system dumps and Java heap dumps:

- Troubleshoot memory leaks
- Understand the architecture of your Java application through footprint analysis
- Improve performance by tuning memory footprint and optimizing Java collections and cache usage
- Customize analysis with additional plug-ins and reports

– IBM Monitoring and Diagnostic Tools for Java - Garbage Collection and Memory Visualizer (GCMV)

Used for Analyzing and visualizing verbose GC logs:

- Monitor and fine-tune Java heap size and garbage collection performance
- Flag possible memory leaks
- Size the Java heap correctly

- Select the best garbage collection policy
- IBM Monitoring and Diagnostic Tools for Java - Dump Analyzer

Used for analyzing system dumps produced by IBM JVMs to diagnose typical problems:

 - Out of Memory
 - Deadlocks
 - Java Virtual Machine (JVM) crashes
 - Java Native Interface (JNI) crashes
- IBM Monitoring and Diagnostic Tools for Java - Health Center

Used for monitoring the status of a running IBM Java Virtual Machine (JVM):

 - Identify if native or heap memory is leaking
 - Discover which methods are taking most time to run
 - Pin down I/O bottlenecks
 - Visualize and tune garbage collection
 - View lock contentions
 - Analyze unusual WebSphere Real Time events
- IBM Thread and Monitor Dump Analyzer for Java (TMDA)

Used for analyzing Java core files to help you identify threading problems:

 - Hangs
 - Deadlocks
 - Resource contention
 - Bottlenecks
- ▶ WebSphere Application Server Troubleshooting Tools

Use these tools to help you troubleshoot problems that are specific to WebSphere Application Server. For example, you can use these tools to analyze HTTP request/response times, JDBC connections, and web services configurations:

 - Database Connection Pool Analyzer for IBM WebSphere Application Server

Used for identifying Java Database Connectivity (JDBC) problems:

 - Connectivity problems
 - JDBC connection leaks
 - IBM Trace and Request Analyzer for WebSphere Application Server

Used for analyzing HTTP server plug-in traces and WebSphere Application Server traces:

 - Identify delays between HTTP requests and responses
 - Identify delays and hangs during application execution
 - IBM Web Server Plug-in Analyzer for WebSphere Application Server

Used for analyzing WebSphere Application Server plug-in configurations to help you find improper or ill-advised settings that can result in runtime problems:

 - Detecting configurations that can cause outages or performance degradation
 - Identifying request or response failures
 - Tracking HTTP return codes and URI failures
 - Tracking clusters and cluster members
 - Graphically visualizing runtime environment topologies
 - IBM Web Services Validation Tool for WSDL and SOAP

Used for validating SOAP messages and WSDL schemas to identify potential problems prior to deployment.

- WebSphere Application Server extensions for Dump Analyzer
Used for analyzing WebSphere Application Server JVM system dumps to diagnose WebSphere specific problems.

Figure 31-8 shows the IBM Support Assistant with its add-ons for problem analysis.

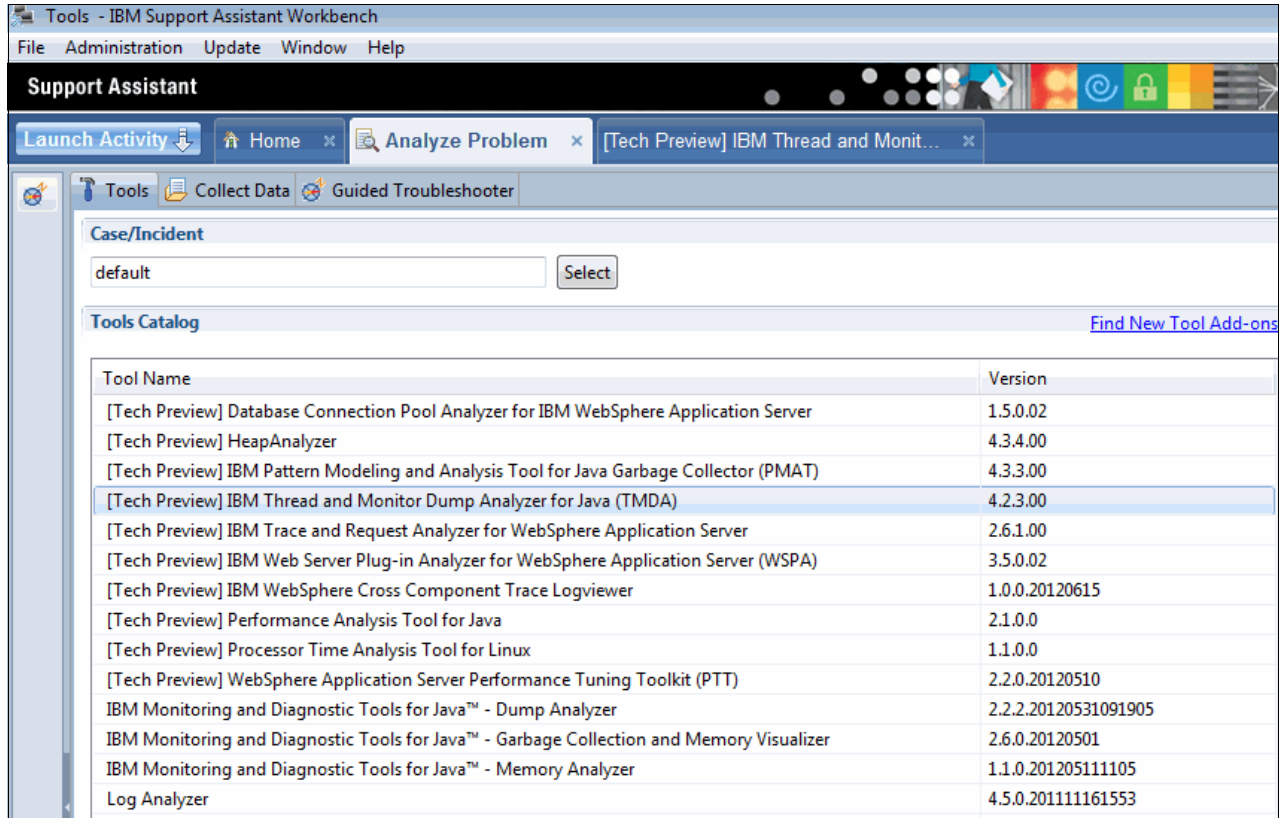


Figure 31-8 IBM Support Assistant

For more information about the IBM Support Assistant add-ons, refer to the following website:
<http://www-01.ibm.com/support/docview.wss?rs=0&uid=swg27013116>

31.4 Troubleshooting scenarios

Now that you have seen what are the most common issues and how to collect information to further diagnose them, this section shows you simulated scenarios of common issues. This section helps you understand what steps need to be taken to identify the root cause of the problem.

31.4.1 Hung threads

This section describes how to troubleshoot hung threads. These can occur due to many reasons, and WebSphere Application Server monitors long running threads to warn about possible issues. After the issue is detected, either by the warning messages in the log files or by other monitoring tools, the first step is to take Java cores. A Java core is a snapshot of all running threads in one specific application server along with the stack trace for each one of them.

Creating Java cores

The following steps outline how to create a Java core using two different ways:

Using a command line:

1. Log into the server, which has the application server experiencing the issue.
2. Locate the process ID for the java process of that application server.
3. Issue the command `kill -3 PID`. Where *PID* is the process ID of the java process.

Using the administration console:

1. From the Web Administration Console, go to **Troubleshooting** → **Java dumps and cores**.
2. Select the application server you want to create a Java core, and click **Java core**.

By default, the java cores are created in the *profile_home* directory with the naming convention of *javacore.<YEAR><MONTH><DAY>.<TIME>.<PID>.<SEQUENTIAL>.txt*, for example: *javacore.20120628.171251.5424.0001.txt*. The exact location of the log file is also shown in the *native_stderr.log* log file. Example 31-8 shows this process and syntax.

Example 31-8 Location and naming of core file

```
JVMDUMP039I Processing dump event "user", detail "" at 2012/06/28 17:12:51 - wait.
```

```
JVMDUMP032I JVM requested Java dump using  
'/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/javacore.20120628.171251.5424.0001  
.txt' in response to an event
```

```
JVMDUMP010I Java dump written to  
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/javacore.20120628.171251.5424.0001.  
txt
```

```
JVMDUMP013I Processed dump event "user", detail "".
```

It is important that you create a few Java cores in regular intervals to be able to detect if a thread is still hung or not. By comparing the java core files and matching with warning messages in the logs, you can detect if a thread is still hung or not and what is the likely cause.

Diagnosing hung thread issues

To diagnose hung thread issues:

1. After a thread is detected hung or if you suspect that something is hung, the first step is to immediately take java cores as described in the section “Creating Java cores”. The minimum number of java cores recommended to take is 3 (with an interval of 2 minutes from each other).
2. Identify from the SystemOut.log file, what thread is reported as hung. Example 31-9 shows an example of a warning message saying that the thread WebContainer: 0 is hung:

Example 31-9 Thread WebContainer hung warning message

```
[6/28/12 18:28:19:227 EDT] 00000070 ThreadMonitor W WSVR0605W: Thread  
"WebContainer : 0" (000000a1) has been active for 146,658 milliseconds and may  
be hung. There is/are 1 thread(s) in total in the server that may be hung.
```

3. Download the java core files to a computer with IBM Support Assistance (ISA) installed. Make sure you have the IBM Thread and Monitor Dump Analyzer for Java (TMDA) installed in ISA.
4. From ISA, Click **Launch Activity** → **Analyze problem**
5. In the Tools Catalog, select **IBM Thread and Monitor Dump Analyzer for Java (TMDA)**, as shown in Figure 31-9.

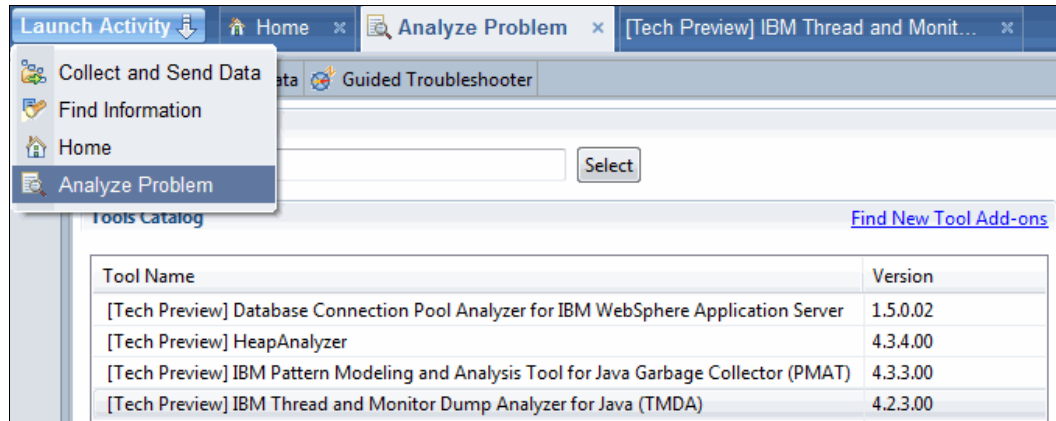


Figure 31-9 Launching TMDA

6. In TMDA, open the Java core files by clicking **File** → **Open Thread Dumps**.
7. Select the Java core file, and click **Open**.
8. Select the Java core file from the list, and click the **Thread Detail icon**, as show in Figure 31-10.

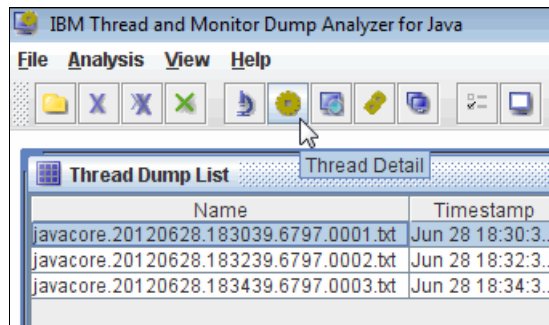


Figure 31-10 Viewing thread details

9. From the list of threads, select the thread identified as hung from step 2 on page 1104. On the right side of the window, you will see the thread details, as shown in Figure 31-11 on page 1106.

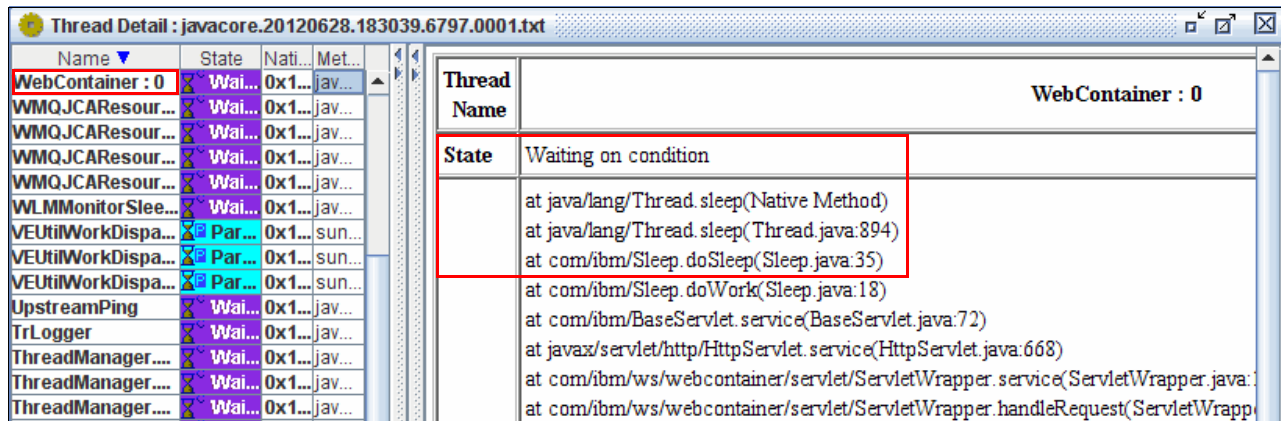


Figure 31-11 Thread details

10. From the window shown in Figure 31-11, you can see the stack trace of the hung thread and identify what is the component causing it. In this case, this is a sample application that issued a sleep call, as shown in Figure 31-11.

31.4.2 High CPU

This section describes how to troubleshoot High CPU usage issues. With this kind of issue, the CPU usage for the process stays high over a long period of time, indicating some abnormality in the infrastructure or in the application.

The high CPU usage analysis is done by identifying what thread or threads are consuming most of the CPU time and then correlating those threads to the Java code they were executing. By doing this, you can pin point what parts of the code are consuming the CPU. High CPU usages can also be caused by excessive execution of the Java garbage collector, most likely because it needs tuning in the algorithm used or in the JVM memory size:

In this example, we use the Linux platform. Complete the following steps to collect the information for diagnosis: If not enabled, enable verbose garbage collection logging as described in section “Enabling verbose garbage collection” on page 1077:

1. Download the `linperf.sh` script from the following MustGather website under Collect Data Manually:

<http://www-01.ibm.com/support/docview.wss?uid=swg21115785>
2. Upload the `linperf.sh` script to the server you are diagnosing and give it execution permission.
3. Identify the Process ID of the top CPU consumer java process. That can be done using the `top` command.
4. As root user, run the script against that PID using `linperf.sh PID`. Make sure you run the script from a directory that is writeable. Also make sure you have enough room in the file system to accommodate the three java core files that are created during the execution of the script.
5. The `linperf.sh` script creates a packaged file with the information it collected from the server. The file gets created in the same directory as `linperf.sh` script was executed and the package is named `linperf_RESULTS.tar.gz`.
6. Download the file `linperf_RESULTS.tar.gz` to the computer with ISA. Also download the Java core files created by the script.

7. Extract the `linperf_RESULTS.tar.gz`, and open the file `topdashH.PID.out`. Where `PID` is the PID of the process in question. What you see is a sequence of `top` commands, which includes the thread information in them (`ps -H` command gives that information).
8. Locate the top CPU consumer threads by looking at the PIDs in the sequence of `top` iterations that consistently shows high CPU usage. Take a note of their PID, it is the first column. Alternatively, use the command line `grep -A 2 "PID USER" topdashH.PID.out` to filter and get the first two lines of all `top` iterations.
9. Convert the thread PID numbers from decimal to hexadecimal using a simple calculator, such as the Windows or Linux calculator. Put the calculator in programmer view, paste the value in decimal that you noted down, and click **Hex**. The value is converted from decimal to hexadecimal.
10. With the top consumer thread PIDs in hexadecimal values ready, we need to investigate in the Java core files. We identify what were those threads doing at the time the Java core was created. Use IBM Thread and Monitor Dump Analyzer for Java (TMDA) in a similar way that we did for diagnosing hung threads.
11. Open the Java core in TMDA following the same instructions as in “Diagnosing hung thread issues” on page 1104.
12. From the list, sort using the *NativeID* column and locate the threads you noted down earlier.

After you select the thread, the stack trace shows on the right side of the window, which gives you a hint of what is taking most of the CPU processing.

Note: It is important that the `linperf.sh` command is executed when the process is consuming high CPU; otherwise, the Java core files taken do not reflect the problem of the occurrence.

In a simulated scenario, we identified the PID 9777 as the top CPU consumer and it is a Java process. Figure 31-12 shows a `top` output.

```
top - 17:27:56 up 23 days, 8:52, 4 users, load average: 27.71, 25.14, 23.41
Tasks: 138 total, 3 running, 135 sleeping, 0 stopped, 0 zombie
Cpu(s): 99.3%us, 0.7%sy, 0.0%ni, 0.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 4098868k total, 3934612k used, 164256k free, 283028k buffers
Swap: 5156824k total, 100k used, 5156724k free, 1598060k cached
```

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
|------|------|----|----|------|------|-----|---|------|------|-----------|-------------|
| 9777 | root | 15 | 0 | 808m | 715m | 93m | S | 99.5 | 17.9 | 167:48.00 | java |
| 1 | root | 15 | 0 | 2160 | 608 | 528 | S | 0.0 | 0.0 | 0:01.08 | init |
| 2 | root | RT | -5 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | migration/0 |
| 3 | root | 34 | 19 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.45 | ksoftirqd/0 |
| 4 | root | 10 | -5 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.02 | events/0 |

Figure 31-12 Top command output

The `linperf.sh` was executed against that PID and displayed the output shown in Example 31-10.

Example 31-10 linperf.sh execution

```
[root@server tmp]# ./linperf.sh 9777
Fri Jun 29 14:40:59 EDT 2012 MustGather>> linperf.sh script starting...
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Script version: 2011.05.03.
Fri Jun 29 14:40:59 EDT 2012 MustGather>> PROBLEMATIC_PID is: 9777
Fri Jun 29 14:40:59 EDT 2012 MustGather>> SCRIPT_SPAN = 240
Fri Jun 29 14:40:59 EDT 2012 MustGather>> JAVACORE_INTERVAL = 120
```

```

Fri Jun 29 14:40:59 EDT 2012 MustGather>> TOP_INTERVAL = 60
Fri Jun 29 14:40:59 EDT 2012 MustGather>> TOP_DASH_H_INTERVAL = 5
Fri Jun 29 14:40:59 EDT 2012 MustGather>> VMSTAT_INTERVAL = 5
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Collecting user authority data...
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Collection of user authority data
complete.
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Creating output files...
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Output files created:
Fri Jun 29 14:40:59 EDT 2012 MustGather>>     vmstat.out
Fri Jun 29 14:40:59 EDT 2012 MustGather>>     ps.out
Fri Jun 29 14:40:59 EDT 2012 MustGather>>     top.out
Fri Jun 29 14:40:59 EDT 2012 MustGather>>     topdashH.9777.out
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Starting collection of top data...
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Collection of top data started.
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Starting collection of top dash H
data...
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Collection of top dash H data started
for PID 9777.
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Starting collection of vmstat data...
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Collection of vmstat data started.
Fri Jun 29 14:40:59 EDT 2012 MustGather>> Collecting a ps snapshot...
Fri Jun 29 14:41:00 EDT 2012 MustGather>> Collected a ps snapshot.
Fri Jun 29 14:41:00 EDT 2012 MustGather>> Collecting a javacore...
Fri Jun 29 14:41:00 EDT 2012 MustGather>> Collected a javacore for PID 9777.
Fri Jun 29 14:41:00 EDT 2012 MustGather>> Continuing to collect data for 120
seconds...
...
Fri Jun 29 14:45:05 EDT 2012 MustGather>> Collecting other data. This may take a
few moments...
Fri Jun 29 14:45:05 EDT 2012 MustGather>> Collected other data.
Fri Jun 29 14:45:05 EDT 2012 MustGather>> Preparing for packaging and cleanup...
Fri Jun 29 14:45:12 EDT 2012 MustGather>> Compressing output files into
linperf_RESULTS.tar.gz
vmstat.out
ps.out
top.out
screen.out
dmesg.out
whoami.out
df-hk.out
topdashH.9777.out
Fri Jun 29 14:45:25 EDT 2012 MustGather>> Cleaning up...
Fri Jun 29 14:45:25 EDT 2012 MustGather>> Clean up complete.
Fri Jun 29 14:45:25 EDT 2012 MustGather>> linperf.sh script complete.

Fri Jun 29 14:45:25 EDT 2012 MustGather>> Output files are contained within ---->
linperf_RESULTS.tar.gz. <----
Fri Jun 29 14:45:25 EDT 2012 MustGather>> The javacores that were created are NOT
included in the linperf_RESULTS.tar.gz.
Fri Jun 29 14:45:25 EDT 2012 MustGather>> Check the <profile_root> for the
javacores.
Fri Jun 29 14:45:25 EDT 2012 MustGather>> Be sure to submit
linperf_RESULTS.tar.gz, the javacores, and the server logs as noted in the
MustGather.

```

```
[root@server tmp]#
```

The topdashH.9777.out file consistently showed PID 15811 as the top CPU consumer, shown in Example 31-11.

Example 31-11 top thread CPU consumers

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
|-------|------|----|----|------|------|-----|---|------|------|---------|-----------------|
| 15811 | root | 25 | 0 | 792m | 590m | 93m | R | 87.5 | 14.8 | 1:05.56 | WebContainer : |
| 9780 | root | 18 | 0 | 792m | 590m | 93m | S | 6.4 | 14.8 | 0:00.64 | Signal Reporter |
| -- | | | | | | | | | | | |
| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
| 15811 | root | 25 | 0 | 792m | 591m | 93m | R | 97.7 | 14.8 | 1:10.45 | WebContainer : |
| 9777 | root | 15 | 0 | 792m | 591m | 93m | S | 0.0 | 14.8 | 0:00.00 | java |
| -- | | | | | | | | | | | |
| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
| 15811 | root | 25 | 0 | 792m | 592m | 93m | R | 90.9 | 14.8 | 1:15.00 | WebContainer : |
| 15918 | root | 18 | 0 | 792m | 592m | 93m | S | 6.8 | 14.8 | 0:00.34 | WebContainer : |
| -- | | | | | | | | | | | |
| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
| 15811 | root | 25 | 0 | 792m | 592m | 93m | R | 96.9 | 14.8 | 1:19.85 | WebContainer : |
| 9781 | root | 15 | 0 | 792m | 592m | 93m | S | 0.2 | 14.8 | 0:25.29 | JIT Compilation |

After converting the PID number 15811 to hexadecimal (3DC3) and opening the Java core files in TMDA, you can see the result. It shows the code from the sample application as being executed at the time that the Java core was created, as shown in Figure 31-13.

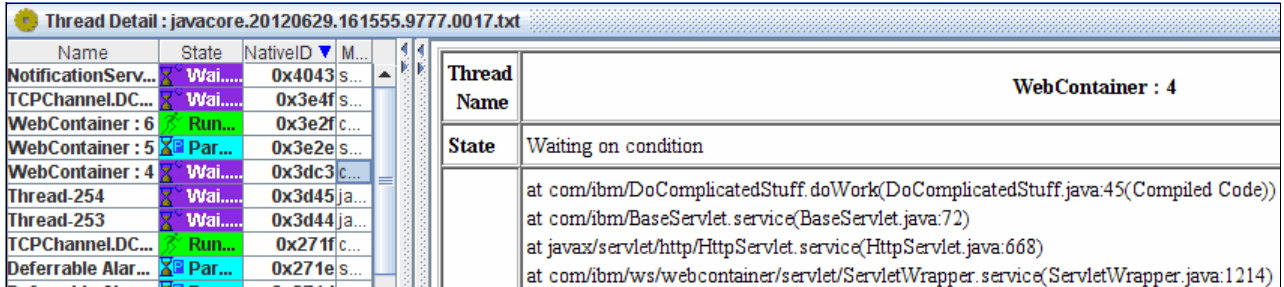


Figure 31-13 Java thread

The conclusion for the analysis in this example is that the code in `com.ibm.DoComplicatedStuff.doWork` is the most likely culprit for the high CPU usage.

31.4.3 Out of Memory exceptions in WebSphere Application Server

An Out of Memory exception can be caused by numerous reasons. A memory leak can be indicated by the steady increase in the memory usage without de-allocation, as shown in Figure 31-14 on page 1110 in IBM Pattern Modeling and Analysis Tool in ISA. It is expected that the Java heap usage fluctuates up and down because the de-allocation of memory is only triggered when the heap usage reaches certain levels.

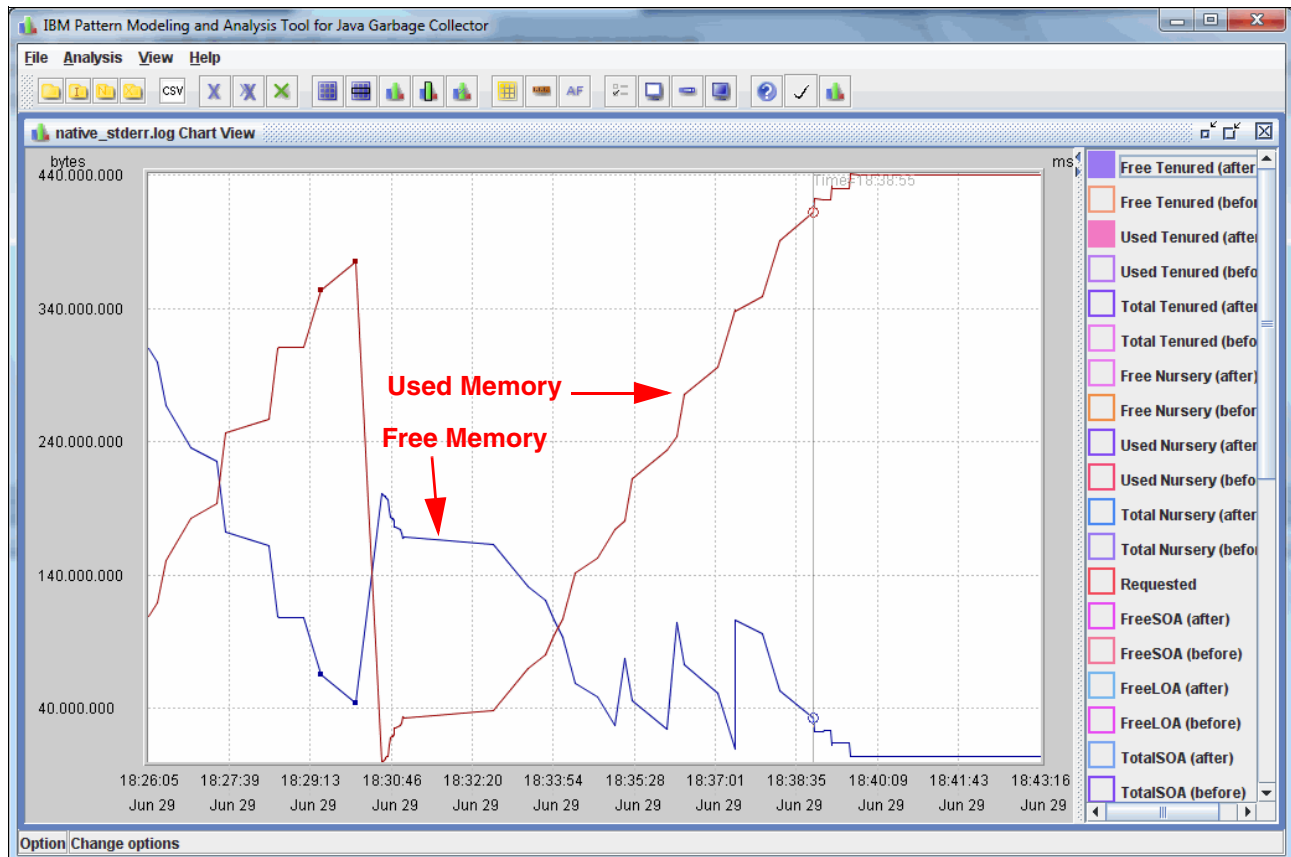


Figure 31-14 Chart graph of a memory leak

Figure 31-14 is similar to a saw, with slow increase in the memory (objects allocation) and then a sudden decrease (garbage collection). If in every cycle, the heap usage after garbage collector execution shows a steady increase, it can also indicate a memory leak condition.

Another type of out of memory issue can be caused by *heap fragmentation*. Heap fragmentation is caused by the allocation and de-allocation of large objects in a short amount of time. The heap will have enough space to accommodate new large objects, but that space is not contiguous, causing an Out of Memory exception. In such cases, a steady increase pattern is not seen in the chart, so then the analysis tools will help in determining if heap fragmentation is the problem.

An out of memory condition is most commonly detected when an Out Of Memory exception is shown in the SystemErr.log file, as shown in Example 31-12. Out Of Memory exceptions are sometimes referred to as OOM.

Example 31-12 Out Of Memory exception in SystemErr.log

```
[6/29/12 18:40:31:919 EDT] 0000008f SystemErr      R java.lang.OutOfMemoryError:
Java heap space
[6/29/12 18:40:31:922 EDT] 0000008f SystemErr      R      at
com.ibm.AllocateObject.doWork(AllocateObject.java:45)
[6/29/12 18:40:31:924 EDT] 0000008f SystemErr      R      at
com.ibm.BaseServlet.service(BaseServlet.java:72)
[6/29/12 18:40:31:925 EDT] 0000008f SystemErr      R      at
javax.servlet.http.HttpServlet.service(HttpServlet.java:668)
```

```

[6/29/12 18:40:31:927 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.servlet.ServletWrapper.service(ServletWrapper.java:1214)
[6/29/12 18:40:31:928 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.servlet.ServletWrapper.handleRequest(ServletWrapper.java:7
74)
[6/29/12 18:40:31:929 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.servlet.ServletWrapper.handleRequest(ServletWrapper.java:4
56)
[6/29/12 18:40:31:929 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.servlet.ServletWrapperImpl.handleRequest(ServletWrapperImp
1.java:178)
[6/29/12 18:40:31:929 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.filter.WebAppFilterManager.invokeFilters(WebAppFilterManag
er.java:1027)
[6/29/12 18:40:31:933 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.webapp.WebApp.handleRequest(WebApp.java:3703)
[6/29/12 18:40:31:936 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.webapp.WebGroup.handleRequest(WebGroup.java:304)
[6/29/12 18:40:31:939 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.WebContainer.handleRequest(WebContainer.java:962)
[6/29/12 18:40:31:943 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.WSWebContainer.handleRequest(WSWebContainer.java:1662)
[6/29/12 18:40:31:946 EDT] 0000008f SystemErr      R      at
com.ibm.ws.webcontainer.channel.WCChannelLink.ready(WCChannelLink.java:195)
[6/29/12 18:40:32:031 EDT] 0000008f SystemErr      R      at
com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.handleDiscrimination(HttpInbo
undLink.java:458)
[6/29/12 18:40:32:032 EDT] 0000008f SystemErr      R      at
com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.handleNewRequest(HttpInboundL
ink.java:522)
[6/29/12 18:40:32:032 EDT] 0000008f SystemErr      R      at
com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.processRequest(HttpInboundLin
k.java:311)
[6/29/12 18:40:32:032 EDT] 0000008f SystemErr      R      at
com.ibm.ws.http.channel.inbound.impl.HttpInboundLink.ready(HttpInboundLink.java:28
2)
[6/29/12 18:40:32:599 EDT] 0000008f SystemErr      R      at
com.ibm.ws.tcp.channel.impl.NewConnectionInitialReadCallback.sendToDiscriminators(
NewConnectionInitialReadCallback.java:214)
[6/29/12 18:40:32:600 EDT] 0000008f SystemErr      R      at
com.ibm.ws.tcp.channel.impl.NewConnectionInitialReadCallback.complete(NewConnectio
nInitialReadCallback.java:113)
[6/29/12 18:40:32:601 EDT] 0000008f SystemErr      R      at
com.ibm.ws.tcp.channel.impl.AioReadCompletionListener.futureCompleted(AioReadComp
letionListener.java:165)
[6/29/12 18:40:32:602 EDT] 0000008f SystemErr      R      at
com.ibm.io.async.AbstractAsyncFuture.invokeCallback(AbstractAsyncFuture.java:217)
[6/29/12 18:40:32:603 EDT] 0000008f SystemErr      R      at
com.ibm.io.async.AsyncChannelFuture$1.run(AsyncChannelFuture.java:205)
[6/29/12 18:40:32:604 EDT] 0000008f SystemErr      R      at
com.ibm.ws.util.ThreadPool$Worker.run(ThreadPool.java:1783)

```

After an Out Of Memory exception is detected, WebSphere Application Server by default creates three log files. The files are one Java Core, one Portable Heap Dump (or simply Heap

dump) and one System Dump (native platform core dump), as shown in the native_stderr.log (Example 31-13).

Example 31-13 native_stderr.log showing an Out Of Memory exception and dumps

```
JVMDUMP039I Processing dump event "systhrow", detail "java/lang/OutOfMemoryError"
at 2012/06/29 18:39:48 - please wait.
JVMDUMP032I JVM requested System dump using
'/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/core.20120629.183948.19153.0001.dmp'
in response to an event
JVMDUMP010I System dump written to
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/core.20120629.183948.19153.0001.dmp
JVMDUMP032I JVM requested Heap dump using
'/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/heapdump.20120629.183948.19153.0002.phd'
in response to an event
JVMDUMP010I Heap dump written to
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/heapdump.20120629.183948.19153.0002.phd
JVMDUMP032I JVM requested Java dump using
'/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/javacore.20120629.183948.19153.0003.txt'
in response to an event
JVMDUMP010I Java dump written to
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/javacore.20120629.183948.19153.0003.txt
JVMDUMP032I JVM requested Snap dump using
'/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/Snap.20120629.183948.19153.0004.trc'
in response to an event
JVMDUMP010I Snap dump written to
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/Snap.20120629.183948.19153.0004.trc
JVMDUMP013I Processed dump event "systhrow", detail "java/lang/OutOfMemoryError".
```

Using these files together with the native_stderr.log, which contains the verbose garbage collection output (if enabled), you can have a first look. These files and logs help diagnose what is accumulating in the memory and causing the Out of Memory exception.

The following steps help identify the likely cause of the Out of Memory Exception:

1. Download the portable heap dump file (.phd) and the native_stderr.log file to a computer with ISA installed. The System Dump file (.dmp) is only needed in case the Out Of Memory exception is being caused by something outside the Java Heap, such as the JVM itself or native libraries.
2. Open the native_stderr.log file in the ISA tool IBM Monitoring and Diagnostic Tools for Java - Garbage Collection and Memory Visualizer.
3. The tool provides views of the heap usage and reports possible issues and recommendations, as shown in Figure 31-15 on page 1113.

[Tuning recommendation](#)

[Version](#)

[VM settings](#)

[System information](#)

[Summary](#)

[Free heap \(before collection\)](#)

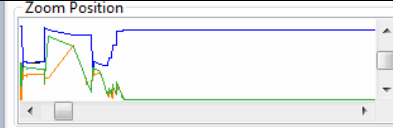
[Free heap \(after collection\)](#)

[Heap size](#)

Tuning recommendation

⊗ The garbage collector seems to be compacting excessively, compaction is an expensive operation and leads to long pause times. On average 56% of each pause was spent compacting the heap. Compaction occurred on 46% of collections. Compaction will occur if the heap is too small or fragmented or if the heap is resized. It will also occur frequently if -Xcompactgc is specified on the command line. If fragmentation is causing compaction you should consider increasing the heap size. If compaction is occurring when the heap is resized consider fixing the heap size by setting -Xmx and -Xms to the same value as this will prevent the heap being resized automatically leading to more predictable pause times. If you have -Xcompactgc enabled and are keen to minimise pause times you may wish to remove that option from your command line. Further information about the command line options can be found in the [Diagnostics Guide](#).

⊗ The Java Heap has been exhausted, leading to an out of memory error. You should consider increasing the Java Heap size using -Xmx if space allows. You can analyse the usage of the Java Heap for a memory leak by using the ISA Tool Add-on, IBM Monitoring and Diagnostic Tools for Java - Memory Analyzer.



Zoom Position

Magnification

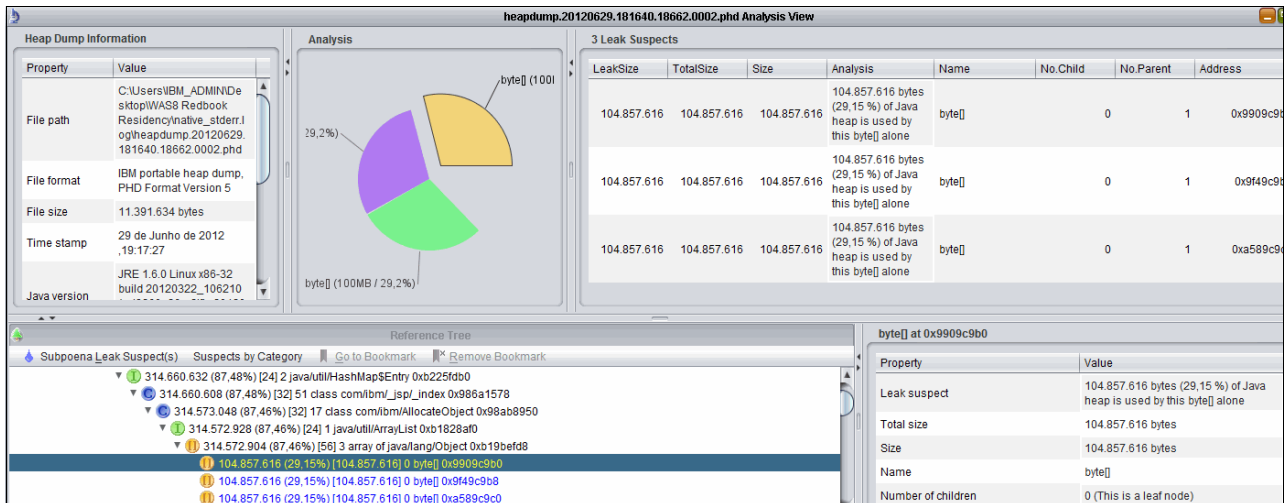
X Axis Zoom Magnification Factor: 55.51

Y Axis Zoom Magnification Factor: 16666.66

Reset Zoom

Figure 31-15 IBM Monitoring and Diagnostic Tools for Java - Garbage Collection and Memory Visualizer

- To understand what objects are getting filled up in memory, use a tool to process the contents of the Portable Heap Dump and allow navigation and drill-down, such as IBM Heap Analyzer or IBM Monitoring and Diagnostic Tools for Java - Garbage Collection and Memory Analyzer. Figure 31-16 shows an example of IBM Heap Analyzer.



Heap Dump Information

| Property | Value |
|--------------|--|
| File path | C:\Users\IBM_ADMIN\Documents\topi\WAS8 Redbook\Residency\native_slderr\log\heapdump.20120629.181640.18662.0002.phd |
| File format | IBM portable heap dump, PHD Format Version 5 |
| File size | 11,391,634 bytes |
| Time stamp | 29 de Junho de 2012, 19:17:27 |
| Java version | JRE 1.6.0 Linux x86-32 build 20120322_106210 |

Analysis

byte[] (100)

29.2%

byte[] (100MB / 29.2%)

3 Leak Suspects

| LeakSize | TotalSize | Size | Analysis | Name | No.Child | No.Parent | Address |
|-------------|-------------|-------------|--|--------|----------|-----------|------------|
| 104.857.616 | 104.857.616 | 104.857.616 | 104.857.616 bytes (29.15%) of Java heap is used by this byte[] alone | byte[] | 0 | 1 | 0x9909c9b0 |
| 104.857.616 | 104.857.616 | 104.857.616 | 104.857.616 bytes (29.15%) of Java heap is used by this byte[] alone | byte[] | 0 | 1 | 0x9f49c9b8 |
| 104.857.616 | 104.857.616 | 104.857.616 | 104.857.616 bytes (29.15%) of Java heap is used by this byte[] alone | byte[] | 0 | 1 | 0xa589c9c0 |

Reference Tree

Subpoena Leak Suspect(s)

- 314.660.632 (87.48%) [24] 2 java.util.HashMap\$Entry 0xb225fdb0
 - 314.660.608 (87.48%) [32] 51 class com.ibm.jsp._index 0x986a1578
 - 314.573.048 (87.46%) [32] 17 class com.ibm/AllocateObject 0x98ab8950
 - 314.572.928 (87.46%) [24] 1 java.util.ArrayList 0xb1828af0
 - 314.572.904 (87.46%) [56] 3 array of java/lang/Object 0xb19befd8
 - 104.857.616 (29.15%) [104.857.616] 0 byte[] 0x9909c9b0
 - 104.857.616 (29.15%) [104.857.616] 0 byte[] 0x9f49c9b8
 - 104.857.616 (29.15%) [104.857.616] 0 byte[] 0xa589c9c0

byte[] at 0x9909c9b0

| Property | Value |
|--------------------|--|
| Leak suspect | 104.857.616 bytes (29.15%) of Java heap is used by this byte[] alone |
| Total size | 104.857.616 bytes |
| Size | 104.857.616 bytes |
| Name | byte[] |
| Number of children | 0 (This is a leaf node) |

Figure 31-16 IBM Heap Analyzer

For more information about Out of Memory analysis, visit the following links at IBM DeveloperWorks:

http://www.ibm.com/developerworks/websphere/library/techarticles/0606_poddar/0606_poddar.html

http://www.ibm.com/developerworks/websphere/library/techarticles/0608_poddar/0608_poddar.html



Additional material

This book refers to additional material that can be downloaded from the Internet as described in the following sections.

Locating the web material

The web material associated with this book is available in softcopy on the Internet from the IBM Redbooks web server at:

<ftp://www.redbooks.ibm.com/redbooks/SG248056>

Alternatively, you can go to the IBM Redbooks website at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG248056.

Using the web material

The additional web material that accompanies this book includes the following files:

- ▶ SG248056.zip
Compressed code samples
- ▶ ClassloaderTestV1.ear, ClassloaderTestV2.ear, ClassloaderTestV3.ear, and VersionCheckerV2.jar
Sample test applications used in *Chapter 22, “Understanding class loaders” on page 789.*

Downloading and extracting the Web material

Create a subdirectory (folder) on your workstation, and download the contents of the web material file into this folder. Extract the `sg248056.zip` file into this folder to access the sample application files.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *WebSphere Application Server V7 Messaging Administration Guide*, SG24-7770
- ▶ *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*, SG24-8022
- ▶ *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580
- ▶ *WebSphere Application Server V8 Security Guide*, SG24-7971
- ▶ *WebSphere Application Server V6 Scalability and Performance Handbook*, SG24-6392
- ▶ *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422
- ▶ *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580
- ▶ *Rational Application Developer for WebSphere Software V8 Programming Guide*, SG24-7835
- ▶ *WebSphere Application Server V6: Diagnostic Data*, REDP-4085
- ▶ *Getting Started with WebSphere Application Server Feature Pack for Service Component Architecture*, REDP-4633
- ▶ *WebSphere Virtual Enterprise Best Practices*, REDP-4461

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Online resources

These web sites are also relevant as further information sources:

- ▶ Command assistance simplifies administrative scripting in WebSphere Application Server
http://www.ibm.com/developerworks/websphere/library/techarticles/0812_rhodes/0812_rhodes.html
- ▶ Properties-based configuration:
http://www.ibm.com/developerworks/websphere/techjournal/0904_chang/0904_chang.html
- ▶ Sample Scripts for WebSphere Application Server:
<http://www.ibm.com/developerworks/websphere/library/samples/SampleScripts.html>

- ▶ IBM HTTP Server performance tuning:
http://publib.boulder.ibm.com/httserv/ihsdiag/ihs_performance.html
- ▶ WebSphere Application Server v8.5 Information Center:
<http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp>
- ▶ *Java Diagnostics Guide*:
<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/topic/com.ibm.java.doc.diagnostics.60/homepage/plugin-homepage-java6.html>
- ▶ IBM Pattern Modelling and Analysis Tool for Java Garbage Collector (PMAT):
<http://www.alphaworks.ibm.com/tech/pmat>
- ▶ IBM Monitoring and Diagnostic tools for Java:
<http://www.ibm.com/developerworks/java/jdk/tools/>
- ▶ Case study: Performance tuning for WebSphere Application Server V7 and V8:
http://www.ibm.com/developerworks/websphere/techjournal/0909_blythe/0909_blythe.html
- ▶ Planning for Availability in the Enterprise:
http://www.ibm.com/developerworks/websphere/techjournal/0312_polozoff/polozoff.html#sec1
- ▶ IBM File System Locking Protocol Test for WebSphere Application Server:
http://www-01.ibm.com/support/docview.wss?rs=180&context=SSEQTP&q1=transaction+log+failover&uid=swg24010222&loc=en_US&cs=utf-8&lang=en
- ▶ IBM Support Assistant:
<http://www.ibm.com/software/support/isa/>
- ▶ Complete list of ISA available add-ons:
<http://www.ibm.com/support/docview.wss?uid=swg27013116>
- ▶ IBM Education Assistant module for The IBM Garbage Collection and Memory Visualizer:
http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.was_v7/was/7.0/ProblemDetermination/WASv7_GCMV0overview/player.html
- ▶ Garbage collection in WebSphere Application Server V8, Part 1:
http://www.ibm.com/developerworks/websphere/techjournal/1106_bailey/1106_bailey.html
- ▶ Garbage collection in WebSphere Application Server V8, Part 2:
http://www.ibm.com/developerworks/websphere/techjournal/1108_sciampacone/1108_sciampacone.html
- ▶ *ITCAM for WebSphere Application Server 7.2*:
http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfad.doc_7101/itcam_ecam_installation_72_85.pdf
- ▶ Classify the Application Control Region in WLM OMVS rules:
<http://www-03.ibm.com/support/techdocs/atmsastr.nsf/WebIndex/TD102730>
- ▶ Java Information Center:
<http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Ftools%2Fjavadump.html>

- ▶ IBM Extended Cache Monitor for IBM WebSphere Application Server technology:
http://www.ibm.com/developerworks/websphere/downloads/cache_monitor.html
- ▶ WebSphere z/OS Optimized Local Adapters:
<http://www-03.ibm.com/support/techdocs/atmsastr.nsf/WebIndex/WP101490>
- ▶ *Introducing the IBM Support Assistant for WebSphere Application Server on z/OS*:
<http://www-03.ibm.com/support/techdocs/atmsastr.nsf/WebIndex/WP101575>
- ▶ IBM Support Assistant:
<http://www-01.ibm.com/software/support/isa/download.html>
- ▶ Java Information Center:
http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.diagnostics.60%2Fdiag%2Funderstanding%2Fmemory_management.html
- ▶ JSR 289 SIP Servlet API 1.1 Specification:
<http://www.jcp.org/aboutJava/communityprocess/final/jsr289/index.html>
- ▶ Overview of the purpose of OSGi:
<http://www.youtube.com/watch?v=J2wq0Y603-Q>
- ▶ OSGi home page:
<http://www.osgi.org/About/HomePage>
- ▶ IBM SDK Java Technology Edition V7 Information Center:
<http://publib.boulder.ibm.com/infocenter/java7sdk/v7r0/index.jsp>
- ▶ Demystifying class loading problems:
http://www.ibm.com/developerworks/java/library/j-dclp1/?S_TACT=106AH10W&S_CMP=NC
- ▶ Samples and Tutorials with SCA using IBM Rational Application Developer:
http://publib.boulder.ibm.com/infocenter/radhelp/v9/index.jsp?topic=%2Fcom.ibm.sca.tools.doc%2Ftopics%2Fsca_tools_intro.html
- ▶ OSGi alliance:
<http://www.osgi.org>
- ▶ Tool to build your OSGi applications, Ant:
<http://ant.apache.org>
- ▶ Tool to build your OSGi applications, Maven:
<http://maven.apache.org>
- ▶ IBM AIX Toolbox for Linux Applications official download website:
<http://www-03.ibm.com/systems/power/software/aix/linux/toolbox/date.html>
- ▶ IBM developerWorks, "Transactional high availability and deployment considerations in WebSphere Application Server V6":
http://www.ibm.com/developerworks/websphere/techjournal/0504_beaven/0504_beaven.html
- ▶ IBM Support Assistant:
<http://www-01.ibm.com/software/support/isa/#isawb>

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services



Redbooks

WebSphere Application Server V8.5 Administration and Configuration Guide for the Full Profile

(2.0" spine)
2.0" <-> 2.498"
1052 <-> 1314 pages



WebSphere Application Server V8.5 Administration and Configuration Guide for the Full Profile



**Learn about
WebSphere
Application Server
V8.5.5**

**Configure and
administer a
WebSphere system**

**Deploy applications in
a WebSphere
environment**

This IBM Redbooks publication provides system administrators and developers with the knowledge to configure an IBM WebSphere Application Server Version 8.5 runtime environment, to package and deploy applications, and to perform ongoing management of the WebSphere environment. As one in a series of IBM Redbooks publications and IBM Redpapers publications for V8.5, the entire series is designed to give you in-depth information about key WebSphere Application Server features.

WebSphere Application Server V8.5 provides two runtime profiles. Every WebSphere Application Server package includes both profile types. The run time traditionally available with the WebSphere Application Server packages is referred to as the full profile. The application serving run time provided with this profile is composed of a wide spectrum of runtime components that are available when the server is started. The full profile provides support for Java Platform Enterprise Edition 6 (Java EE 6) and Enterprise OSGi technologies.

The Liberty profile provides a simplified stand-alone run time for web applications, supporting a subset of the programming model available with the full profile. Any application that runs on the Liberty profile will also run on the full profile.

In this book, we provide a detailed exploration of the WebSphere Application Server V8.5 runtime administration process for the full profile. This book includes configuration and administration information for WebSphere Application Server V8.5 and WebSphere Application Server Network Deployment V8.5 on distributed platforms and WebSphere Application Server for IBM z/OS V8.5.

**INTERNATIONAL
TECHNICAL
SUPPORT
ORGANIZATION**

**BUILDING TECHNICAL
INFORMATION BASED ON
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:
ibm.com/redbooks**

SG24-8056-01

ISBN 0738438537