

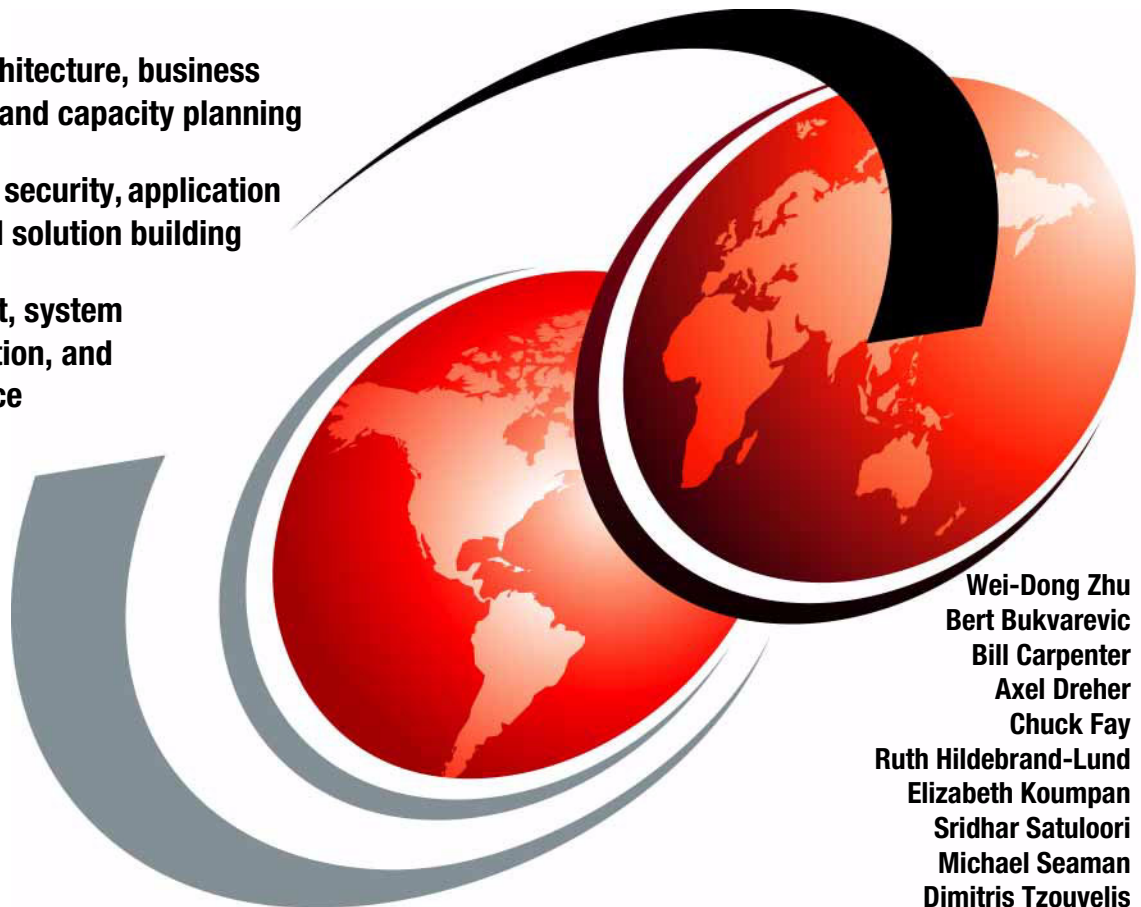


IBM FileNet Content Manager Implementation Best Practices and Recommendations

System architecture, business
continuity, and capacity planning

Repository, security, application
design, and solution building

Deployment, system
administration, and
maintenance



Wei-Dong Zhu
Bert Bukvarevic
Bill Carpenter
Axel Dreher
Chuck Fay
Ruth Hildebrand-Lund
Elizabeth Kouman
Sridhar Satuloori
Michael Seaman
Dimitris Tzouvelis



International Technical Support Organization

IBM FileNet Content Manager Implementation Best Practices and Recommendations

June 2013

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

Second Edition (June 2013)

This edition applies to Version 5, Release 2 IBM FileNet Content Manager (product number 5724-R81).

Contents

| | |
|---|------|
| Notices | xiii |
| Trademarks | xiv |
| Preface | xv |
| Authors | xvi |
| Now you can become a published author, too! | xix |
| Comments welcome | xix |
| Stay connected to IBM Redbooks | xix |
| Summary of changes | xxi |
| June 2013, Second Edition | xxi |
| Chapter 1. Introduction to IBM FileNet Content Manager | 1 |
| 1.1 Industry challenges and IBM solutions benefits | 2 |
| 1.1.1 Industry challenges | 2 |
| 1.1.2 Information lifecycle governance | 4 |
| 1.1.3 Benefits of IBM ECM solutions | 4 |
| 1.2 IBM FileNet P8 Platform | 5 |
| 1.2.1 Platform components | 5 |
| 1.2.2 Enterprise capabilities | 7 |
| 1.3 IBM FileNet Content Manager | 7 |
| 1.3.1 Basic capabilities | 8 |
| 1.3.2 Enterprise foundation | 9 |
| 1.4 IBM FileNet P8 and related products | 11 |
| 1.4.1 Content products | 12 |
| 1.4.2 Ingestion products | 13 |
| 1.4.3 Process products | 14 |
| 1.4.4 Compliance products | 14 |
| 1.4.5 Collaboration products | 16 |
| 1.5 Conclusion | 16 |
| Chapter 2. Solution examples and design methodology | 17 |
| 2.1 P8 Content Manager sample solutions | 18 |
| 2.1.1 Policy document creation | 18 |
| 2.1.2 Processing insurance claims | 20 |
| 2.1.3 Archiving SAP invoices | 23 |
| 2.1.4 Email capture for compliance | 25 |
| 2.1.5 Knowledge management through collaboration | 27 |

| | | |
|---|--|-----------|
| 2.2 | Design methodology | 28 |
| 2.2.1 | Defining ECM strategy | 29 |
| 2.2.2 | Requirements analysis | 30 |
| 2.2.3 | Functional design | 33 |
| 2.2.4 | System architecture design | 33 |
| 2.2.5 | Repository design | 34 |
| 2.2.6 | Security model design | 34 |
| 2.2.7 | Application design | 35 |
| 2.2.8 | Test planning | 35 |
| 2.2.9 | Deployment | 35 |
| 2.2.10 | Maintenance planning | 36 |
| 2.3 | Conclusion | 36 |
| Chapter 3. System architecture | | 37 |
| 3.1 | Basic components | 38 |
| 3.1.1 | Additional components | 39 |
| 3.1.2 | Data organization | 40 |
| 3.1.3 | Object stores | 42 |
| 3.1.4 | Storage considerations | 43 |
| 3.1.5 | Workflow systems | 46 |
| 3.1.6 | Management tools | 46 |
| 3.1.7 | Bulk Import Tool | 48 |
| 3.1.8 | Hardware layout | 49 |
| 3.1.9 | Setting up a sandbox or demo environment | 49 |
| 3.1.10 | Using Information Center and other product documentation | 50 |
| 3.2 | Scalability | 52 |
| 3.2.1 | Horizontal scalability | 54 |
| 3.2.2 | Vertical scalability | 54 |
| 3.2.3 | Clustering | 54 |
| 3.2.4 | P8 domain and object store scaling | 56 |
| 3.2.5 | Scaling Content Search Services | 57 |
| 3.3 | Virtualization | 57 |
| 3.3.1 | A virtualized IBM FileNet Content Manager system | 60 |
| 3.4 | Shared infrastructure | 62 |
| 3.4.1 | Communication between the engines | 62 |
| 3.4.2 | Data segregation | 65 |
| 3.4.3 | Levels of data segregation | 66 |
| 3.4.4 | Degree of sharing | 68 |
| 3.4.5 | Cloud deployments | 69 |
| 3.5 | Geographically distributed systems | 69 |
| 3.5.1 | Site, virtual server, and server configuration | 69 |
| 3.5.2 | Distributed content caching model | 73 |
| 3.5.3 | Request forwarding | 74 |

| | |
|---|-----------|
| 3.5.4 Distributed workflow systems | 77 |
| 3.5.5 Use cases for distributed systems | 77 |
| 3.6 Conclusion | 79 |
| Chapter 4. Repository design | 81 |
| 4.1 Repository design goals | 82 |
| 4.2 Object-oriented design | 82 |
| 4.2.1 Design approaches | 83 |
| 4.2.2 Design processes | 85 |
| 4.3 Repository naming standards | 87 |
| 4.3.1 Display name | 87 |
| 4.3.2 Symbolic name | 87 |
| 4.3.3 Uniqueness | 88 |
| 4.3.4 Taxonomy | 88 |
| 4.3.5 Consistency | 89 |
| 4.3.6 Object stores | 89 |
| 4.3.7 Storage areas | 89 |
| 4.3.8 Document, custom object, and folder classes | 90 |
| 4.3.9 Property templates | 90 |
| 4.3.10 Choice lists | 91 |
| 4.4 Populating a repository | 91 |
| 4.4.1 Generic object system properties | 93 |
| 4.4.2 Creating design elements | 95 |
| 4.5 Repository organizational objects | 95 |
| 4.6 Global configuration database (GCD) | 97 |
| 4.7 Repository design objects | 98 |
| 4.7.1 Object stores | 98 |
| 4.7.2 Storage areas | 100 |
| 4.7.3 Document classes | 102 |
| 4.7.4 Folder classes | 106 |
| 4.7.5 Custom object classes | 109 |
| 4.7.6 Custom root classes | 110 |
| 4.7.7 Property templates | 111 |
| 4.7.8 Choice lists | 112 |
| 4.7.9 Annotations | 113 |
| 4.7.10 Document lifecycles | 113 |
| 4.7.11 Events and subscriptions | 115 |
| 4.7.12 Marking sets | 116 |
| 4.8 Repository content objects | 117 |
| 4.8.1 Folder objects | 117 |
| 4.8.2 Other objects | 125 |
| 4.9 Storage media | 125 |
| 4.9.1 Catalog | 126 |

| | |
|---|------------|
| 4.9.2 Database stores | 127 |
| 4.9.3 File stores | 127 |
| 4.9.4 About storage policies. | 131 |
| 4.9.5 Using fixed storage devices | 133 |
| 4.10 Considerations for multiple object stores | 133 |
| 4.11 Retention management and automatic disposal | 137 |
| 4.11.1 Retention management. | 137 |
| 4.11.2 Automatic disposition | 139 |
| 4.11.3 Retention update. | 139 |
| 4.12 P8 Content Manager searches | 140 |
| 4.12.1 User-invoked searches | 141 |
| 4.12.2 Content-based search. | 141 |
| 4.12.3 Searches for repository maintenance | 146 |
| 4.12.4 CBR query optimization | 148 |
| 4.13 Conclusion. | 149 |
| Chapter 5. Security | 151 |
| 5.1 Access control | 152 |
| 5.2 Authentication | 152 |
| 5.2.1 Use of JAAS | 152 |
| 5.2.2 Directory service users and groups. | 153 |
| 5.2.3 Security context. | 155 |
| 5.3 Authorization | 156 |
| 5.3.1 Access rights. | 156 |
| 5.3.2 Security descriptor | 158 |
| 5.3.3 Default security descriptor. | 161 |
| 5.3.4 Security templates. | 161 |
| 5.3.5 Proxies | 163 |
| 5.3.6 Markings | 167 |
| 5.3.7 The access check | 169 |
| 5.3.8 Auditing | 178 |
| 5.4 Security best practices | 179 |
| 5.4.1 Physical security measures. | 180 |
| 5.4.2 Directory service configuration | 181 |
| 5.4.3 Defining the security approach | 182 |
| 5.4.4 Planning for evolution | 184 |
| 5.4.5 Role-based access control using inheritance | 186 |
| 5.4.6 Using markings | 187 |
| 5.4.7 Effective use of auditing | 188 |
| 5.4.8 Cache management | 188 |

| | |
|---|-----|
| Chapter 6. Application design | 191 |
| 6.1 IBM FileNet P8 applications | 192 |
| 6.1.1 IBM Administration Console for Content Platform Engine | 192 |
| 6.1.2 IBM Content Navigator | 192 |
| 6.2 Application technologies | 193 |
| 6.2.1 Traditional Java thick clients | 193 |
| 6.2.2 Java applets | 194 |
| 6.2.3 Java EE web applications and other components | 194 |
| 6.2.4 .NET components | 195 |
| 6.3 Principles for application design | 195 |
| 6.3.1 Available P8 Content Manager APIs | 196 |
| 6.3.2 Transports available with the APIs | 200 |
| 6.3.3 Minimizing round-trips | 204 |
| 6.3.4 Parallel processing | 206 |
| 6.3.5 Client-side transactions | 207 |
| 6.3.6 Creating a custom AddOn | 208 |
| 6.3.7 Using the JDBC interface for reporting | 209 |
| 6.3.8 Exploiting the active content event model | 210 |
| 6.3.9 Logging | 211 |
| 6.3.10 Creating a data model | 212 |
| | |
| Chapter 7. Business continuity | 217 |
| 7.1 Defining business continuity | 218 |
| 7.2 Defining high availability (HA) | 218 |
| 7.3 Implementing a high availability solution | 221 |
| 7.3.1 Load-balanced server farms | 221 |
| 7.3.2 Active-passive server clusters | 226 |
| 7.3.3 Geographically dispersed server clusters and server farms | 230 |
| 7.3.4 Server cluster products | 231 |
| 7.3.5 Comparing and contrasting farms to clusters | 232 |
| 7.3.6 Inconsistent industry terminology | 233 |
| 7.3.7 Server virtualization and high availability | 234 |
| 7.4 Defining disaster recovery (DR) | 234 |
| 7.4.1 Disaster recovery concepts | 235 |
| 7.5 Implementing a disaster recovery solution | 236 |
| 7.5.1 Replication | 236 |
| 7.5.2 Automated site failover | 241 |
| 7.5.3 Disaster recovery approaches | 242 |

| | |
|---|------------|
| 7.6 Best practices | 245 |
| 7.7 Reference documentation | 250 |
| Chapter 8. Capacity planning with IBM Content Capacity Planner | 253 |
| 8.1 IBM Content Capacity Planner | 254 |
| 8.1.1 Example use cases for IBM Content Capacity Planner | 255 |
| 8.1.2 Capacity planning for new systems. | 256 |
| 8.1.3 IBM Content Capacity Planner output. | 260 |
| 8.1.4 Predictions from a baseline | 261 |
| 8.1.5 Best practices | 263 |
| 8.2 IBM FileNet Disk sizing Tool spreadsheet | 266 |
| 8.3 Performance-related reference documentation. | 267 |
| 8.3.1 Standard product documentation | 268 |
| 8.3.2 Benchmark papers | 268 |
| 8.4 Conclusion. | 268 |
| Chapter 9. Deployment | 271 |
| 9.1 Overview | 272 |
| 9.2 Deployment environments. | 273 |
| 9.2.1 Single stage development environment | 273 |
| 9.2.2 Multi-stage deployment environments | 274 |
| 9.3 Deployment by using a formal methodology | 275 |
| 9.3.1 Release management | 278 |
| 9.3.2 Change management | 281 |
| 9.3.3 Configuration management. | 286 |
| 9.3.4 Testing | 287 |
| 9.4 Deployment approaches | 291 |
| 9.4.1 Cloning | 292 |
| 9.4.2 Custom-scripted export, transform, and import. | 293 |
| 9.4.3 Scripted generation. | 294 |
| 9.5 Deployment based on cloning | 295 |
| 9.5.1 Cloning an object store | 297 |
| 9.5.2 Topology | 298 |
| 9.5.3 Access to the environment | 299 |
| 9.5.4 Post-cloning activities | 299 |
| 9.5.5 Backup changes | 299 |
| 9.6 Deployment by export, transform, and import | 299 |
| 9.6.1 Incremental deployment compared to full deployment | 299 |
| 9.6.2 Reducing the complexity of inter-object relationships. | 301 |
| 9.6.3 Deployment automation | 303 |
| 9.7 FileNet Content Manager deployment. | 303 |
| 9.7.1 FileNet Content Manager export. | 305 |
| 9.7.2 CE-objects transformation. | 307 |

| | |
|---|------------|
| 9.7.3 Content Platform Engine import best practice | 309 |
| 9.7.4 IBM FileNet Deployment Manager | 309 |
| 9.7.5 Exporting and importing other components | 311 |
| 9.8 Conclusion | 313 |
| Chapter 10. System administration and maintenance | 315 |
| 10.1 IBM FileNet Content Manager administrative roles | 316 |
| 10.2 Online help and existing documentation | 317 |
| 10.2.1 Tips for working with the information center | 320 |
| 10.2.2 Other useful documentation | 321 |
| 10.3 Monitoring the environment | 321 |
| 10.4 Capacity monitoring and growth prediction | 326 |
| 10.4.1 IBM System Dashboard for ECM | 327 |
| 10.4.2 Dashboard | 328 |
| 10.4.3 IBM ECM System Monitor | 334 |
| 10.5 Tracing | 336 |
| 10.6 Auditing | 337 |
| 10.7 Managing the logs | 338 |
| 10.7.1 Log location | 338 |
| 10.7.2 Log file size | 339 |
| 10.7.3 Trace logs | 340 |
| 10.7.4 Audit logs | 340 |
| 10.8 System administration tools | 340 |
| 10.8.1 Configuration Manager | 340 |
| 10.8.2 IBM Administration Console for Content Platform Engine | 341 |
| 10.8.3 Consistency checker | 351 |
| 10.8.4 Database tools | 351 |
| 10.8.5 Application server administration tools | 353 |
| 10.8.6 Workflow system tools | 353 |
| 10.8.7 IBM Content Navigator tools | 353 |
| 10.9 Reducing storage costs | 354 |
| 10.9.1 Retention rules | 354 |
| 10.9.2 Using the sweep framework | 355 |
| 10.9.3 Monitoring storage and cache usage | 356 |
| 10.10 Using virus scan software | 359 |
| 10.11 Applying fixes | 360 |
| 10.11.1 Tracking fixes | 360 |
| 10.11.2 Checking compatibility and build numbers | 361 |
| 10.11.3 Reporting issues and downloading fixes | 361 |
| 10.12 Updating security | 362 |
| 10.13 Backup and restore | 364 |
| 10.13.1 System components requiring backup | 364 |
| 10.13.2 Offline backup | 365 |

| | |
|---|------------|
| 10.13.3 Online backup | 366 |
| 10.13.4 System restore | 367 |
| 10.13.5 Application consistency check | 368 |
| 10.14 Task schedule | 369 |
| 10.15 Conclusion. | 370 |
| Chapter 11. Upgrade and migration | 371 |
| 11.1 Terminology. | 372 |
| 11.1.1 Packages | 372 |
| 11.1.2 Package naming conventions | 373 |
| 11.1.3 Installation rules | 374 |
| 11.1.4 Update types | 375 |
| 11.2 Planning for updates | 377 |
| 11.2.1 Getting started. | 378 |
| 11.2.2 Practicing the update | 378 |
| 11.2.3 Documenting the process | 380 |
| 11.3 Upgrading to a new software release | 380 |
| 11.3.1 Staging the upgrade | 381 |
| 11.3.2 Big-bang upgrade | 382 |
| 11.4 Migration best practices | 382 |
| 11.5 Special considerations for upgrade. | 383 |
| 11.5.1 Reference information. | 384 |
| 11.6 Conclusion. | 385 |
| Chapter 12. Troubleshooting | 387 |
| 12.1 A typical P8 Content Manager system | 388 |
| 12.2 Different types of troubleshooting | 390 |
| 12.3 Creating customized best practice guides. | 390 |
| 12.4 General troubleshooting | 393 |
| 12.5 Troubleshooting the installation or upgrade | 394 |
| 12.6 Troubleshooting during application development | 403 |
| 12.7 Troubleshooting functional issues. | 404 |
| 12.7.1 Review the logs. | 404 |
| 12.7.2 Review additional sources for information about issues | 409 |
| 12.8 Troubleshooting production issues | 410 |
| 12.9 Troubleshooting performance issues | 411 |
| 12.9.1 Performance tuning guide | 412 |
| 12.9.2 Gathering performance data | 412 |
| 12.9.3 Slow logon. | 414 |
| 12.9.4 Slow searches. | 417 |
| 12.9.5 Storage performance issues | 418 |
| 12.9.6 Tuning sweep jobs | 418 |

| | | |
|--|--|------------|
| 12.10 | Opening PMRs | 419 |
| 12.10.1 | The IBM software support portal | 419 |
| 12.10.2 | Open a PMR by calling IBM | 419 |
| 12.10.3 | Open a PMR via the web | 419 |
| 12.10.4 | Necessary items when contacting IBM software support | 420 |
| 12.10.5 | IBM Support Assistant (ISA) Workbench | 423 |
| 12.10.6 | Type of fixes that might be provided | 423 |
| 12.10.7 | Rolling up fixes | 424 |
| 12.11 | Conclusion | 425 |
| Chapter 13. IBM FileNet Content Manager solutions | | 427 |
| 13.1 | Solution building blocks | 428 |
| 13.1.1 | Foundation components | 429 |
| 13.1.2 | Content ingestion tools | 441 |
| 13.1.3 | Process management | 444 |
| 13.1.4 | Presentation features | 447 |
| 13.2 | Sample use cases using solution building blocks | 448 |
| 13.2.1 | Policy document creation use case | 449 |
| 13.2.2 | Insurance claim processing use case | 452 |
| 13.2.3 | SAP invoice archiving use case | 455 |
| 13.2.4 | Email capture for compliance use case | 457 |
| 13.2.5 | Knowledge management through collaboration use case | 459 |
| 13.3 | Conclusion | 461 |
| Related publications | | 463 |
| | IBM Redbooks | 463 |
| | Online resources | 464 |
| | Help from IBM | 468 |

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 give 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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites 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>

| | | |
|----------------------------|--------------|---|
| AIX® | IBM® | Rational® |
| ClearCase® | InfoSphere® | Redbooks® |
| Cognos® | Lotus Notes® | Redbooks (logo)  ® |
| DB2® | Lotus® | System p® |
| developerWorks® | Notes® | SystemMirror® |
| Domino® | OpenPower® | Tivoli Enterprise Console® |
| Enterprise Storage Server® | Optim™ | Tivoli® |
| FileNet® | PowerHA® | WebSphere® |
| Global Business Services® | pSeries® | |
| GPFS™ | pureScale® | |

The following terms are trademarks of other companies:

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.

Network Appliance, SnapMirror, SnapLock, NetApp, and the NetApp logo are trademarks or registered trademarks of NetApp, Inc. in the U.S. and other countries.

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

IBM® FileNet® Content Manager Version 5.2 provides full content lifecycle and extensive document management capabilities for digital content. IBM FileNet Content Manager is tightly integrated with the family of IBM FileNet products based on IBM FileNet P8 Platform. IBM FileNet Content Manager serves as the core content management, security management, and storage management engine for the products.

This IBM Redbooks® publication covers the implementation best practices and recommendations for solutions that use IBM FileNet Content Manager. It introduces the functions and features of IBM FileNet Content Manager, common use cases of the product, and a design methodology that provides implementation guidance from requirements analysis through production use of the solution. We address administrative topics of an IBM FileNet Content Manager solution, including deployment, system administration and maintenance, and troubleshooting.

Implementation topics include system architecture design with various options for scaling an IBM FileNet Content Manager system, capacity planning, and design of repository design logical structure, security practices, and application design. An important implementation topic is business continuity. We define business continuity, high availability, and disaster recovery concepts and describe options for those when implementing IBM FileNet Content Manager solutions.

Many solutions are essentially a combination of information input (ingestion), storage, information processing, and presentation and delivery. We discuss some solution building blocks that designers can combine to build an IBM FileNet Content Manager solution.

This book is intended to be used in conjunction with product manuals and online help to provide guidance to architects and designers about implementing IBM FileNet Content Manager solutions.

Many of the features and practices described in the book also apply to previous versions of IBM FileNet Content Manager.

Product name changes: New for Version 5.2, IBM FileNet Business Process Manager has been renamed to IBM Case Foundation.

Authors

This book was produced by a team of specialists from around the world working at the IBM Software Development Lab in Costa Mesa, California.

Wei-Dong Zhu (Jackie) is an Enterprise Content Management (ECM) Project Leader with IBM in Los Angeles, California. She has more than 10 years of software development experience in accounting, image workflow processing, and digital media distribution. Jackie holds a Masters of Science degree in Computer Science from the University of the Southern California. Jackie joined IBM in 1996. She is a Certified Solution Designer for IBM Content Manager and has managed and led the production of many Enterprise Content Management IBM Redbooks publications.

Bert Bukvarevic is an IT Specialist for ECM with IBM in Germany. He has 10 years of experience in the ECM Platform. He has worked at IBM for six years. His area of expertise is a T-shape skill, which means that the IBM FileNet P8 Content Platform is required to collaborate across different technologies. He has written extensively about deployment, upgrades, and migration.

Bill Carpenter is an ECM Architect with IBM in the Seattle area. Bill has had experience in ECM since 1998 as a developer, development manager, and as an architect. He is the author of the book *Getting Started with IBM FileNet P8 Content Manager*. He is also co-author of the first edition of this book and *Developing Applications with IBM FileNet P8 APIs*, a contributing author for IBM developerWorks®, and a frequent conference presenter. He has experience in building large software systems at Fortune 50 companies and has also served as the CTO of an Internet start-up. He has been a frequent mailing list and patch contributor to several open source projects. Bill holds degrees in Mathematics and Computer Science from Rensselaer Polytechnic Institute in Troy, New York.

Axel Dreher is a Managing Consultant working as an ECM Architect and Project Leader with IBM in Villingen-Schwenningen, Germany. He has more than nine years of experience in designing and implementing high-performance, high-volume, and high-availability solutions around the IBM FileNet product suite and has worked at IBM for five years. Axel studied media and computer science and graduated in Computer Science with a Diplom-Informatiker degree from the Fachhochschule Furtwangen, University of Applied Sciences in Germany. He is an IBM Certified Specialist for IBM FileNet Content Manager and IBM Case Foundation. Axel specializes in infrastructure implementation and troubleshooting for IBM FileNet P8 solutions.

Chuck Fay is a Software Architect for Enterprise Content Management systems with IBM in Costa Mesa, California. He has over thirty years of experience in the software industry, as a developer, development manager, and software architect at Xerox, FileNet, and now IBM since its acquisition of FileNet in 2006. He is a co-inventor on four patents and developer of software standards issued by AIIM, IETF, and OMG. Since 2000, he has advised FileNet and IBM clients, as well as engineering, support, and sales representatives, about system deployment architectures to ensure high availability and enable disaster recovery, for IBM FileNet products. He holds an A.B. in Philosophy and an M.S. in Computer Science, both from Stanford University.

Ruth Hildebrand-Lund is a Customer Engagement Specialist with IBM in the US. She has 15 years of experience in the ECM field. She has worked at IBM for six years. Her areas of expertise include planning for, deploying, testing, and managing the FileNet P8 suite of products.

Elizabeth Koumpan is a Senior IT Application Architect with the Business Analytics and Optimization practice of IBM Global Business Services® in Canada. She specializes in business process and enterprise content management systems and has a broad spectrum of functional and technical skills. Elizabeth has led numerous technology assessment projects at insurance companies and global banks as well as implemented and architected business process management (BPM) solutions for their core business processes. Elizabeth has over twenty years of experience in system design and architecture and has been with IBM more than six years. She has been involved in the development of the IBM ECM Methodology framework.

Sridhar Satuloori is a Content Platform Engine Developer with the IBM Software Development Group in Costa Mesa, California. He has worked with the Content Platform Engine development for the past eight years. He has worked on various components of the Content Platform Engine, including content-based retrieval (CBR) with the CSS server, security, and content federation services. Sridhar also worked on the Authentication and Single Sign-On (SSO) feature for the IBM Content Navigator. He leads the maintenance and support activities for the Content Platform Engine, and is experienced in working with the Building Enterprise Edition of Java EE application servers. Sridhar wrote several articles about the CBR with the CSS server and Authentication and SSO features of the IBM Content Navigator. Sridhar holds a Masters degree in Computer Science from IIT Roorkee, India.

Michael Seaman is a Software Architect with IBM in Cambridge, UK. He has worked on ECM products for FileNet and IBM for seventeen years and, in that period, has designed and implemented many of the core features of the P8 product. Prior to that, he had fifteen years of industry experience in the areas of networking and distributed systems. Mike is a co-inventor of a dozen or so software patents and holds a Masters degree in Mathematics from Cambridge University.

Dimitris Tzouvelis is a Project Manager and also ECM Consultant with IBM Global Services in Greece. He has 14 years of experience in the ECM field. He has worked at IBM for six years. His areas of expertise include designing, analyzing, implementing, and managing projects based on the IBM FileNet ECM platform. He holds a degree in Physics from Athens University in Greece and a Master in Business Administration from University of Piraeus in Greece.

We also thank the following people for their contribution during the Redbooks publication production period:

Kenytt Avery
Roger Bacalzo
Kevin Bates
Quynh Dang
Patrick Doonan
Anita Jayaraman
David Keen
Bob Kreuch
Tim Lai
Vincent Le
Shari Perryman
Yvonne Santiago
John Spanoudakis
Shawn Waters
Mike Winter
IBM Software Group, US

Thorsten Poggensee
Marcus Mueller-Westerholt
IBM Global Business Service, Germany

Bruce Taylor
Tony Laino
IBM Software Group, Canada

Thanks to the authors of the first edition of this book:
Wei-Dong Zhu, Dan Adams, Dominik Baer, Bill Carpenter, Chuck Fay, Dan McCoy, Thomas Schrenk, and Bruce Weaver

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:

<http://www.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:

<http://www.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>

Summary of changes

This section describes the technical changes made in this edition of the book and in previous editions. This edition might also include minor corrections and editorial changes that are not identified.

Summary of Changes
for SG24-7547-01
for IBM FileNet Content Manager Implementation Best Practices and
Recommendations
as created or updated on June 7, 2013.

June 2013, Second Edition

This is the second edition of the book. The first edition was published in 2008, covering IBM FileNet Content Manager, Version 4.0.0. It was the first IBM Redbooks publication specifically dealing with the IBM FileNet product family. Since that time, there have been several IBM FileNet Content Manager product releases introducing new features and capabilities, and entirely new products in the IBM FileNet family. Also since that time, several additional IBM Redbooks titles have been published on related topics. Thus, this Redbooks publication now maintains the basic structure of the first edition, but it is both expanded in scope (where there are new recommendations to be described) and reduced in size (where other IBM Redbooks publications or product reference material already covers the information in detail).

All chapters have been revised for this update.



Introduction to IBM FileNet Content Manager

IBM FileNet Content Manager (P8 Content Manager) provides full content lifecycle and extensive document management capabilities for digital content. This chapter introduces P8 Content Manager and describes its features and functionality to demonstrate how P8 Content Manager makes an excellent foundation for enterprise content management solutions.

We discuss the following topics:

- ▶ Industry challenges and IBM solutions benefits
- ▶ IBM FileNet P8 Platform
- ▶ IBM FileNet Content Manager
- ▶ IBM FileNet P8 and related products

1.1 Industry challenges and IBM solutions benefits

Enterprise Content Management (ECM) uses technology to simplify and bring consistency to entering, creating, storing, managing, retaining, and deleting information.

ECM technology alone does not ensure that a company follows good document management practices. To be successful, companies must develop good working practices and embed them into the implementation of the ECM technology.

To facilitate this approach, the ECM technology must have the right flexibility and toolset to implement and maintain these practices, and prevent turning your ECM implementation for managing documents and content into a never-ending development and support project. A successful ECM implementation follows step-by-step processes or a building block approach for designing and implementing the desired solution. Building a solid base allows the solution to be expanded to meet additional business needs as they arise.

Figure 1-1 illustrates the types of building blocks that can be included in a fully featured ECM solution. The IBM portfolio of ECM products and solutions provides the building blocks to help organizations implement ECM successfully.

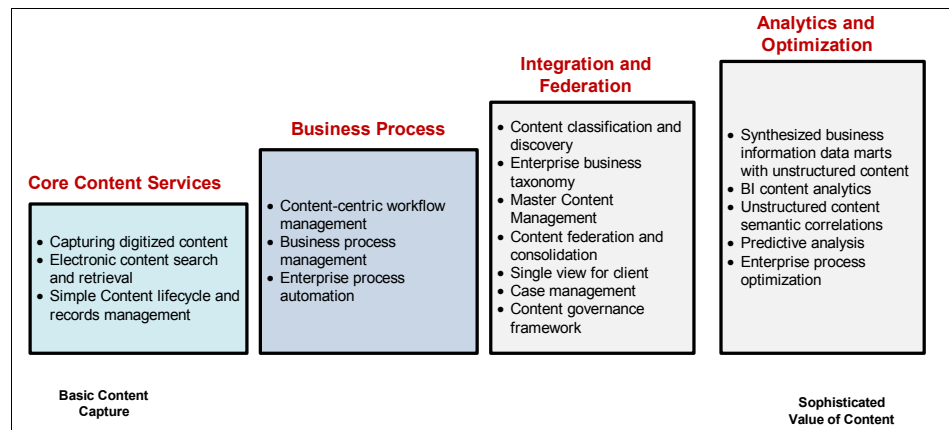


Figure 1-1 ECM implementation

1.1.1 Industry challenges

In the quest for increased efficiency and profitability, organizations strive to incorporate more and more relevant information into their business processes in order to make the right decision at the right time. Corporations want to enable their employees to search, retrieve, and review information in context, to limit

exception handling and manual processing, to reduce costs, and to improve service. It is important to have all the information needed and make the right decision that satisfies clients, partners, suppliers, and shareholders:

► Oil and Gas

According to industry experts, the top business challenge facing the Energy Industry today is regulatory compliance. To be compliant in this highly regulated industry, an organization must ensure that all relevant documents are available whenever and wherever required, vital information is archived, and unstructured content, process workflows, and collaborative workflow systems are managed, all within a central repository.

Refining organizations continue to investigate tools that allow for the revision and markup of CAD drawings while integrating with their ECM repository. The ability to search for drawings based on information contained in title blocks and markups is instrumental to engineering operations.

► Insurance

Saddled with IT infrastructures built years ago, health insurers also lack centralized control capabilities and automated processes. Most processes are inefficient, manual, and paper based, which drains profits at an alarming rate. Health insurers must explore new technologies that empower customers with immediate, real-time access to their policies and claim status via the web. They need customer-centric business models that match their customer needs. This is essential for acquiring new customers and for retaining existing customers.

► Finance

Many financial institutions handle their client requests by using paper-driven processes. Replacing the paper with electronic forms and documents and routing the tasks automatically to the user desktop provide more efficient responses to customer needs.

The progression from immature, chaotic document management practices to a mature, well-managed enterprise content management process starts with the understanding of where you are and where you need to be. You need to decide how far and how fast to move, because transitioning an established culture is much harder than delivering technology.

One of the other challenges facing global companies is that there are no common legal requirements and regulations. So, a company that is operating in different countries has to fulfill different obligations. Therefore, gathering all legal requirements is a fundamental step in implementing an ECM solution.

1.1.2 Information lifecycle governance

It used to be that when IT infrastructures were being put in place, the cost of storage was considered inexpensive, so putting in place policies and tools for removing content was a low priority. But with the volume of data being generated today, the cost of storage is no longer trivial, and neither is the task of managing all this content.

In addition to managing how much data is kept, it is also important to categorize the content appropriately so that it can be retrieved easily and quickly.

Information governance is the discipline of managing information according to its legal obligations and its business value to enable the defensible disposal of data, as well as to meet the needs of clients who are using the client to perform their daily tasks.

A formal Information Governance framework establishes chains of responsibility, authority, and communication. It describes the roles of people involved in the production cycle of content, their responsibilities, the ways in which they interact, and the general rules and policies about the production of content.

Electronic information systems are generally more complex than paper information systems. The requirement to manage the system to ensure the integrity, reliability, and authenticity of the information for the long term is not readily apparent.

Good information governance requires specificity and transparency on the legal and regulatory obligations and the business value of information. This relates to the people tasked with actually managing information, and establishes measurement, policy, and control mechanisms to enable people to carry out their roles and responsibilities. Information Governance is an essential part of any ECM strategy and is necessary to realize the full extent of business benefits derivable from a corporation's information management activities.

1.1.3 Benefits of IBM ECM solutions

The ability to make decisions better and faster is a real competitive advantage that *IBM Enterprise Content Management* (ECM) solutions can help provide. The IBM ECM portfolio of products improves workforce effectiveness by enabling organizations to transform their business processes, access and manage all forms of content, secure and control information related to compliance needs, and optimize the infrastructure required to deliver content anywhere at anytime.

IBM ECM helps organizations make quick, smart, and cost-effective decisions, right at the moment that it matters the most.

IBM ECM solutions provide the following benefits:

- ▶ Active content
Delivering information that is unified, accurate, and in context with critical business processes and policy management
- ▶ Business agility
Providing the right information to the right people at the right time in the right context to enable better decisions faster
- ▶ Enterprise compliance
Managing risk and automating compliance with records management, legal discovery, and intelligent content archiving
- ▶ Content anywhere
Managing content on any system without requiring content migration
- ▶ Pervasive and persuasive
Accessing, collaborating, and influencing business decisions in new ways by making content a first-class source of decision-making insight

1.2 IBM FileNet P8 Platform

The IBM FileNet P8 Platform family of products is part of the IBM ECM product suite. The IBM FileNet P8 Platform includes back-end services, development tools, and applications that address enterprise content and process management requirements.

The IBM FileNet P8 Platform is a unified content, process, and compliance platform that offers maximum flexibility, accelerates application deployment, and lowers the total cost of ownership. It is an integrated platform that provides interoperability to a wide selection of database, operating system, storage, security, and web server environments. It serves as the core content management, security management, and storage management engine for the IBM ECM family of products. It is an integration platform that can be extended by adding additional components as needed and integrating it with other line of business (LOB) applications.

1.2.1 Platform components

The IBM FileNet P8 Platform includes the baseline components for enterprise content management solutions:

- ▶ Content Platform Engine

- ▶ IBM Content Navigator
- ▶ ECM Collaboration Services
- ▶ FileNet Content Search Services
- ▶ FileNet Content Federation Services
- ▶ FileNet Rendition Engine
- ▶ FileNet Content Manager Information Center

These components address enterprise content management and are (except for ECM Collaboration Services and FileNet Content Federation Services) described in Chapter 3, “System architecture” on page 37. Some components are included in the base platform offering and some require additional licensing.

All IBM P8 Platform capabilities are inherited and make a foundation for IBM ECM solutions. Additional components can be added to a system to enable additional capabilities.

The IBM P8 Platform capabilities can be leveraged for a wide range of enterprise scalable solutions, including case management, automated content collection using the IBM Content Collectors and Datacap, Image Manager, IBM Enterprise Records, and Information Lifecycle Governance (ILG) portfolio of products and more. Figure 1-2 illustrates the interaction of some of the ECM portfolio of products with P8 Content Manager.

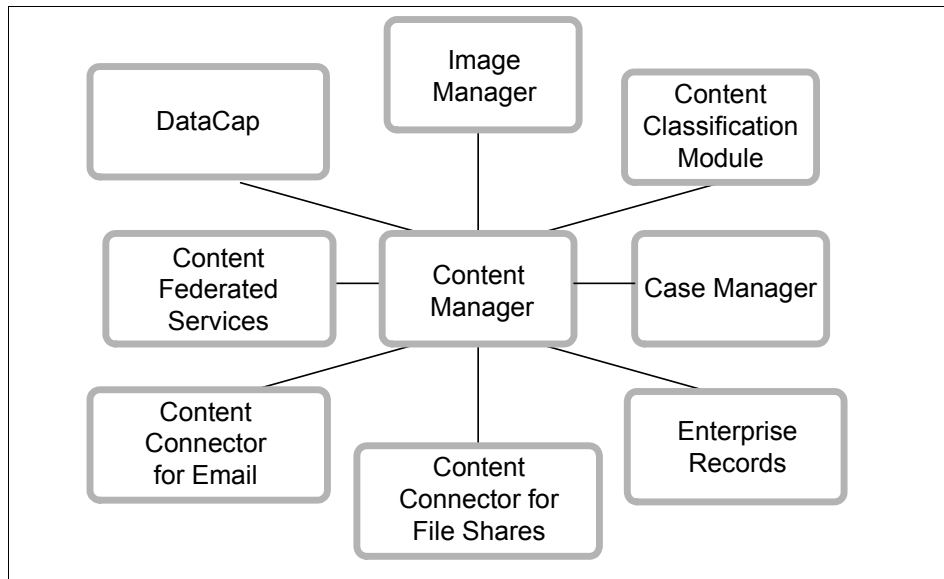


Figure 1-2 Major platform components

1.2.2 Enterprise capabilities

The IBM FileNet P8 Platform family of products provides a variety of enterprise capabilities for organizations, including open and extensible environment, high ingestion and large storage capacity capabilities, and scalable architecture.

Open and extensible

To promote an open and extensible environment, IBM P8 Platform provides APIs for developing custom applications.

The IBM P8 Platform offers Java and .NET APIs that offer objects and methods to handle creating, retrieving, updating, and deleting objects, folders, and documents. Content Management Interoperability Services (CMIS) and Web Services are also supported.

Content Platform Engine provides an extensible Object Model that includes a set of predefined classes and the ability to create new custom classes.

High ingestion and large storage capacity

Content Platform Engine is designed to handle high volumes of information. Performance studies show that P8 Content Manager can handle both high ingestion rates (millions of documents per hour) and large amounts of stored information (over a billion objects in a single repository).

Scalable architecture

Content Platform Engine achieves excellent performance rates with a scalable architecture, offering both vertical and horizontal scalability solutions.

IBM Content Platform can be farmed (scaled horizontally) or can be scaled vertically by either running multiple instances on a single box or configuring a single instance to use multiple servers.

Multiple servers can be added in load-balanced configurations to handle increasing transaction loads. This architecture makes the Content Platform Engine an ideal candidate for large corporations, government agencies, or any client with large information management requirements.

1.3 IBM FileNet Content Manager

IBM FileNet Content Manager provides full content lifecycle and extensive document management capabilities for digital content.

P8 Content Manager combines document management with readily available workflow and process capabilities to automate and drive content-related tasks and activities. It provides a unique *active content* capability to proactively move content and content-related business tasks through a business process without requiring human initiation. In addition, P8 Content Manager streamlines document management tasks by providing mature versioning and parent-child capabilities, approval workflows, and integrated publishing support and is designed for large-scale enterprise content management initiatives:

- ▶ Content Manager is designed for high volume environments.
- ▶ Content Manager is adept at managing active content.
- ▶ Content Manager is tightly integrated with the IBM ECM suite of supporting applications.
- ▶ Content Manager is a library for electronic content. It can store simple objects, such as documents and images, as well as more complex objects, such as workflows, email streams, and corporate records. It is designed as a central repository for any type of electronic information. Some other types of electronic content include audio files, web content, XML files, rich media, and fax. Content Manager provides a rich set of content management services, tools, built-in features, and programmable functionality that system designers can use to meet enterprise content management goals.

1.3.1 Basic capabilities

P8 Content Manager provides enterprise content management services, tools, built-in features, and programmable functionality that system designers can use to meet enterprise content management goals.

P8 Content Manager by default supports ultimate reusability by creating or reusing shared components. Such architecture incorporates different design patterns and eliminates redundancy, allowing a quicker time to delivery, reducing issues in development and lowering the end maintenance costs.

By using P8 Content Manager, the documents are stored in a repository where they are available to all users based on the security model.

P8 Content Manager offers flexible tools that coordinate the revision, routing, and processing of objects stored in the repository. A powerful feature of P8 Content Manager is the support for active content. *Active content* is electronic information that changes according to a document lifecycle or business process. *Document lifecycle* is a feature of P8 Content Manager. It consists of a sequence of steps. *Business processes*, however, are executed by the IBM Case Foundation (previously known as IBM FileNet Business Process Manager). Business processes typically contain branching logic. In essence, both are a series of

steps, with each step representing an event or process that acts on the object content.

P8 Content Manager features event action scripts that can be triggered when objects are created, modified, or deleted in the repository. Event actions can launch workflows or execute Java applications. Events, and the actions that they trigger, are the mechanism that enables active content.

When designing an ECM solution, document retention can be addressed and implemented by using the retention management features, including the sweep framework and event-based retention.

IBM Content Navigator is a ready-to-use, standards-based user interface that delivers intelligence and control of collaborative content across the organization.

1.3.2 Enterprise foundation

P8 Content Manager is a foundational building block for IBM ECM solutions. The tight integration with the IBM P8 Platform components increases operational efficiency by reducing the number of products and vendors across the enterprise.

This foundation can be expanded from basic document management support to enterprise-wide applications by adding additional components and capabilities.

Next, we provide examples of how P8 Content Manager can be used as the foundation for policy document management, invoice processing, and compliance support.

Policy document management

Policy lifecycle management represents one of the common business activities across many organizations. From the creation of policy papers, to their edits, approvals, retrieval, storage, and archiving, policy touches every employee within the organization.

A robust ECM solution can address the challenges associated with policy lifecycle management through the adoption of centralized and standardized processes.

Policy professionals can develop policies by working in a collaborative environment to more effectively execute the tasks involved in policy management. There are a range of common activities that are typically undertaken in the policy development process, including creating a policy document as a draft, and moving it through the typical document revision process for review and approval. The policy document is usually published on the organizational website at the end of the process.

In the policy document management solution:

- ▶ The document becomes active content as it moves from state to state according to the document lifecycle.
- ▶ A single trusted policy document source with versioning control, standard classification, and reliable search results is established across the organization.
- ▶ A role-based access model is established to support secure and efficient policy management publication and access.
- ▶ IBM Case Foundation is used to automate the review and publishing processes.

Invoice processing

A major challenge for the account payable process is the ability to process a large number of paper-based invoices, and place them into the system for tracking and control invoice processing.

The solution must enable the processing of invoice transactions using a structured, automated process. This process should secure content associated with each invoice, providing stakeholders with visibility to the process steps. A content management platform integrated with an accounts payable process also enables the continuous improvement of internal business operations over time by managing each payment case, its associated content, and reporting on the efficiency of the business process.

The solution includes the ability to extract invoice data from paper invoice documents during scanning and indexing, applying advanced lookups against business applications and systems of record, and creating an invoice in the Accounts Payable application. A robust document capture solution, such as that provided by Datacap, supports the required high volume ingestion rates. Within the process, the invoice is used to support day-to-day activities, storing electronic copies of the documents as records of evidence.

Users (whether internal or external) are able to perform work from within the data processing application. The content associated with the data systems is stored in the repository.

The solution provides the following capabilities:

- ▶ Enables the ingestion of both scanned paper and digital content, whether submitted by internal or external users, and the automatic classification of the content.

- ▶ Provides access to the content as an integrated capability of the Accounts Payable interface, ensuring that access to the invoices is seamless to the client.
- ▶ Allows you to add and relate additional content to a specific invoice throughout the invoice lifecycle.
- ▶ Provides notification to clients if an invoice requires special attention.

Compliance support for insurance claim processing

Claims service is a key differentiator for insurance companies. Claims processing is supported by documents, such as medical assessments, police reports, and estimates. Information is processed and stored in electronic format and available for Claims Adjudicators to view, annotate, and redact.

A content ingestion solution provides the scanning, recognition, and indexing capabilities that support efficiently processing both paper and electronic documents and uploading them into an ECM repository. Content Management support for Electronic Claim File provides an enterprise solution for electronically stored claim information and digital media, which are integrated with the core claims administration application. The Claim Adjudicator can retrieve and present, in a single view, claim and policy documents from different sources.

Corporate retention schedules are applied to all claim data and supporting documents. When the required retention period expires, the electronic claim data can be automatically deleted.

The solution is intended to support the following key business functions:

- ▶ A centralized repository for a variety of media types, including email, to support claims adjudication.
- ▶ Interactive process for claims processing, including common content services functions, such as adding, searching, and retrieving data.
- ▶ User access to claims processing from a single user interface.
- ▶ Extensibility to support additional capabilities and capacity.
- ▶ Compliance by ensuring all claim data is automatically disposed of based on any corporate or legal retention requirements.

1.4 IBM FileNet P8 and related products

The IBM FileNet P8 family of products is based on P8 Content Manager and the Content Platform Engine. In addition, there are many other IBM ECM offerings

that integrate with P8 Content Manager. The products can be grouped into the following categories:

- ▶ Content products
Enabling companies to activate content with processes to add value and transform their business.
- ▶ Ingestion products
Used to add documents into a repository by copying, moving, or linking to source documents, as well as index and classify the content.
- ▶ Process products
Automate and optimize complex processes across the enterprise using effective content and compliance.
- ▶ Compliance products
Ensure content is kept as required and deleted when no longer needed.
- ▶ Collaboration products
Provide an environment where people can work together to achieve business goals and streamline processes.

1.4.1 Content products

The portfolio of content products that integrate with P8 Content Manager:

- ▶ FileNet Image Manager
- ▶ IBM Content Federation Services
- ▶ IBM Content Search Services
- ▶ IBM Content Navigator
- ▶ IBM FileNet Integration to Microsoft Office

There are many content products in the IBM ECM portfolio. It is beyond the scope of this book to introduce all of them. However, to help you better understand what these products can do for your corporation, we briefly introduce several of them here.

IBM FileNet Content Federation Services

IBM FileNet Content Federation Services (CFS) enables content stored in different repositories, such as OpenText and Documentum, to be used in P8 Content Manager applications. This capability can be used to give a single access point to content in multiple repositories and to utilize the records management capabilities provided by IBM Enterprise Records. The content can stay in the external repository and, if appropriate, can also be migrated over time into a P8 Content Manager repository.

IBM Content Navigator

IBM Content Navigator is a ready-to-use, open web-based standards user interface that provides a modern user experience for working with all forms of content. It supports all content management use cases, and is configurable to the needs of an organization without requiring a development effort for basic customization.

IBM Content Search Services

IBM Content Search Services (CSS) is a tool for indexing both the content and string metadata of documents to support the use of content-based retrieval.

1.4.2 Ingestion products

The portfolio of ingestion products that integrate with P8 Content Manager:

- ▶ IBM Content Collector
- ▶ IBM Content Federation Services
- ▶ IBM Datacap

To help you better understand what these products can do for your corporation, we briefly introduce two of them here.

IBM Content Collector

IBM Content Collector can be used to ingest content from these sources:

- ▶ Email servers
- ▶ File shares
- ▶ Microsoft Sharepoint servers
- ▶ SAP

Based on the configuration of the connector to the ingestion source, IBM Content Collector, in addition to adding the content to a P8 Content Manager repository, also performs other tasks:

- ▶ Creating stubs to the document in a manner transparent to existing client applications
- ▶ Declaring content as records
- ▶ Deduplicating content, which is especially important for email attachments

IBM Datacap

IBM Datacap is a scanning and indexing solution that is integrated with Content Platform Engine. This product features a complete suite of document indexing capabilities, including automated document identification and text recognition.

Datacap automates the input of data from documents to reduce cost and accelerate document process efficiencies.

1.4.3 Process products

The portfolio of process products that integrate with P8 Content Manager:

- ▶ IBM Case Analyzer
- ▶ IBM Case Manager
- ▶ IBM Forms
- ▶ IBM Case Foundation

To help you better understand what these products can do for your corporation, we briefly introduce them here.

IBM Case Analyzer

IBM Case Analyzer can analyze, monitor, and report information at both the process level and case level. It allows you to identify dynamic trends and monitor the case processing in real time. The captured data can be used by tools, such as IBM Cognos® Real Time Monitor.

IBM Case Manager

IBM Case Manager is a platform for case management with advanced capabilities for rules, events collaboration, social software, and analytics. The Case Manager functionality and user interfaces allow business analysts to easily design and build case solutions.

IBM Forms

IBM Forms provides an electronic alternative to paper forms. With an easy to use visual design environment, the electronic forms support easy data entry, can be used to perform calculations, and can be integrated into business processes.

IBM Case Foundation

IBM Case Foundation (previously known as IBM FileNet Business Process Manager) is tightly integrated with P8 Content Manager to provide tools for defining, optimizing, and processing business processes.

1.4.4 Compliance products

The portfolio of compliance products that integrate with P8 Content Manager:

- ▶ IBM Content Analytics

- ▶ IBM Content Classification
- ▶ IBM eDiscovery Manager and eDiscovery Analyzer
- ▶ IBM Enterprise Records
- ▶ IBM FileNet Compliance Framework
- ▶ IBM Global Retention Policy and Schedule Management

To help you better understand what these products can do for your corporation, we briefly introduce a few of them here.

IBM Content Analytics

IBM Content Analytics is an advanced analytics platform that enables better decision making from the enterprise content regardless of the source or format. Content Analytics solutions can understand the meaning and context of human language and rapidly process information to improve knowledge-driven search and surface new insights from your enterprise content.

IBM Content Classification

IBM Content Classification categorizes and organizes content by combining multiple methods of context-sensitive analysis. Content Classification enables clients to focus on higher-level activities by consistently and accurately automating content-centric categorization decisions.

IBM eDiscovery tools

IBM eDiscovery Manager and IBM eDiscovery Analyzer focus on the needs of collecting and processing unstructured data for legal discovery, automating applying legal holds and collections processes by holding data in place, and collecting evidence from data repositories.

The tools automate the identification of unstructured data by searching for relevant data by keyword or date-range, as well as archive and deduplicate email and attachments, and dynamically classify and declare records of collected emails through integration with IBM Enterprise Records.

eDiscovery Analyzer enables legal professionals to analyze, sort, and cull evidence immediately after it has been collected rather than by performing manual document reviews.

IBM Enterprise Records

IBM Enterprise Records is a complete application that securely manages the declaration, classification, security and access, auditing and monitoring, authenticity, preservation, and destruction of electronic and physical records.

IBM Enterprise Records utilizes unique “Zero Click” technology to reduce the burden and costs associated with proper management of an organization’s records and integrates directly with Content Platform Engine to allow repository objects to be managed as records.

Use IBM Enterprise Records with the other compliance, process, and ingestion tools to provide an end-to-end records management solution that covers the capture, management, and destruction of records.

IBM Global Retention Policy and Schedule Management

IBM Global Retention Policy and Schedule Management is a single, cohesive retention management system with natively integrated workflows and analytics for information governance stakeholders.

1.4.5 Collaboration products

The portfolio of collaboration products that integrate with P8 Content Manager includes ECM Collaboration Services.


IBM Connections (formerly Lotus® Connections) and IBM ECM Collaboration Services provide collaborative authoring and sharing of business content. In particular, a social collaboration environment unlocks enterprise content and makes it accessible across the corporation. Content can be selectively placed into the ECM repository for safekeeping. There is support for many of the typical social collaboration features, such as comments, tagging, and download counters.

1.5 Conclusion

The unified content and process architecture of Content Platform Engine provides inherent capabilities to integrate content with applications to provide integrated support for data transactions and to coordinate business processes.

IBM core ECM provides the foundation for delivering robust solutions across the enterprise.

In the next chapter, we provide detailed examples illustrating the use of P8 Content Manager as part of an ECM solution.



Solution examples and design methodology

IBM FileNet Content Manager (P8 Content Manager) can be used for the implementation of a large spectrum of applications that vary from small departmental systems to large enterprise systems running in a complex environment and integrating with several other systems. In this chapter, we describe common P8 Content Manager solutions, each illustrating P8 Content Manager's features and capabilities. In addition, we introduce the design methodology that guides you from the Enterprise Content Manager (ECM) strategy definition through deployment and administration planning. The methodology is used as the structure for the remaining chapters of the book.

We describe the following topics:

- ▶ P8 Content Manager sample solutions:
 - Policy document management process
 - Invoice archiving
 - Email archiving
 - Insurance claim processing
 - Social Enterprise Content Management
- ▶ Design methodology

2.1 P8 Content Manager sample solutions

In this section, we present five common P8 Content Manager sample solutions:

- ▶ Policy document creation
- ▶ Insurance claim processing
- ▶ SAP invoice archiving
- ▶ Email archiving
- ▶ Training material development

Each solution demonstrates how you can implement P8 Content Manager to solve enterprise content management challenges using a different approach.

The following different approaches are available for the implementation of an ECM system:

- ▶ Simple stand-alone content management application
- ▶ ECM system integrated with the line of business (LOB) application
- ▶ Archiving solutions
- ▶ Social ECM solutions

Typically, a mix of the approaches is used in organizations based on the needs to meet the business requirements and challenges.

The policy document creation sample solution demonstrates a simple document management solution with check-in check-out capabilities and security. The insurance claim processing sample solution demonstrates the integration and record management capabilities of P8 platform. The SAP invoice archiving solution demonstrates the SAP archiving and high availability capabilities of P8 platform and intelligent document recognition (ICR). The Email archiving solution demonstrates the archiving and compliance capabilities of the P8 platform. The social ECM sample solution demonstrates the social features of the P8 platform.

Note: The solutions we present here are simplified versions. In actual installations, the solutions are often more sophisticated than what we describe here. We simplify the scenarios and their solutions for ease of reading and understanding. The important point from this section is to get an idea of what P8 Content Manager can do to help solve your business problems.

2.1.1 Policy document creation

In the first P8 Content Manager solution, a team of authors, reviewers, and managers is responsible for updating policy documents. Those documents define the organization guidelines. Goals must be rigidly reviewed, and all

workers must always have access to the current policy information. The following flow is for the events of this use case:

1. The author creates a new document for revision and assigns a minor version number on the documents.
2. The reviewers and authors collaborated in this document and adding minor versions of the document.
3. The approver approves the final revision of the document and checks in the document as a major version.
4. The major version is available now to all users based on their permissions and security is applied to the document as the final approved version of the document.

This P8 Content Manager solution is implemented with the following features:

- ▶ Content versioning, including major and minor versions
- ▶ Document access controlled by role-based permissions and dependency on document lifecycle status
- ▶ Check-in and check-out capability

Figure 2-1 illustrates the implemented document revision and approval process using P8 Content Manager.

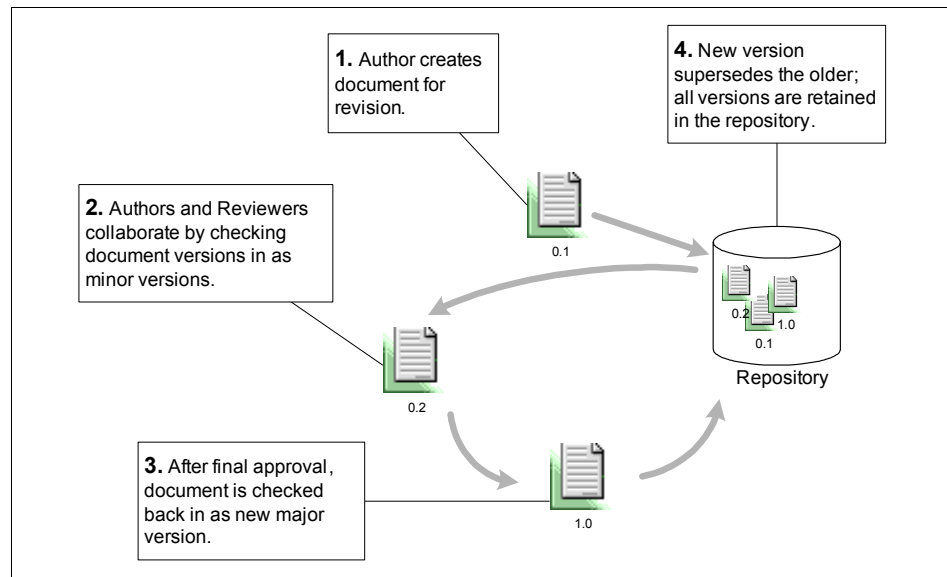


Figure 2-1 Document revision and approval process

Solution description

This solution uses P8 Content Manager features without additional programming. In the design, policy documents are stored in the repository where they are available to all users for reference.

Each policy document goes through a document lifecycle with multiple states. In this implementation, the states are minor and major versions. A *minor version* is a draft document; a *major version* is a completed document that has been approved and released. A security policy is implemented to define the security that applies to documents in the major version state and those in the minor version state. Minor versions can only be viewed and modified by authors, reviewers, and approvers. They are invisible to general users. All users can view major versions, but only authors, reviewers, and managers can modify them.

For simplification and to reflect the majority of actual solutions, this sample solution does not include document retention. When implementing a document revision solution for your environment, you must address your document retention requirements and include them in your solution implementation as necessary.

2.1.2 Processing insurance claims

In this sample P8 Content Manager solution from the insurance industry, the company policy governs how claims are processed. The main system of records for this solution is the insurance company claim processing system and the P8 Content Manager is used as the content repository together with the IBM Enterprise Records for compliance.

The following flow of events is for the sample insurance claim processing solution:

1. A new claim is opened in the Claim Management application and the claim ID is communicated to the customer.
2. The claim management application will invoke a service in the ECM repository, populate all claim-related metadata, including claim ID, propagating the proper retention policy based on the claim type.
3. The claim-related documents arrive from different sources and may include electronic documents, email, electronic forms, and paper documents.
4. Paper documents will be converted to an electronic format and send for processing.
5. Using Datacap optical character recognition (OCR)/intelligent document recognition (ICR) capabilities, the claim number will be extracted from the electronic document and the documents will be stored in the ECM system. If

the claim number cannot be found in the electronic documents, an exception process will be initiated through the ECM system to the Claim Management system.

6. After the document is filed into FileNet Content Manager, a corresponding record object will be created and placed under the IBM Enterprise Records record folder where retention will be aggregated on the claim records folder.
7. A task will be created within the Claim Management application triggered by the document release in the repository.
8. The integration between the Claim Processing application and IBM ECM will allow users to access, view, load, and link the documents to the claim case throughout the Claim Processing lifecycle, according to their established permissions. Also, the ECM system using the Content Federation will present content from other repositories, such as policy documents to the authorized users.
9. The document metadata is updated from the Claim Management application as the claim moves through the process.
10. The claim close event from the Claim Management application will trigger a service request to ECM to start the retention process.
11. After the claim case record and all associated documents will be ready for disposition, the service request will be sent to the Claim Processing application to automatically dispose of structured data in the same time as unstructured content.

This P8 Content Manager solution is implemented with the following features and components:

- ▶ Document scanning and character recognition
- ▶ Integration with LOB application
- ▶ Active content event actions
- ▶ Records management
- ▶ Content Federation Services

Figure 2-2 illustrates the implemented insurance claim processing solution using P8 Content Manager.

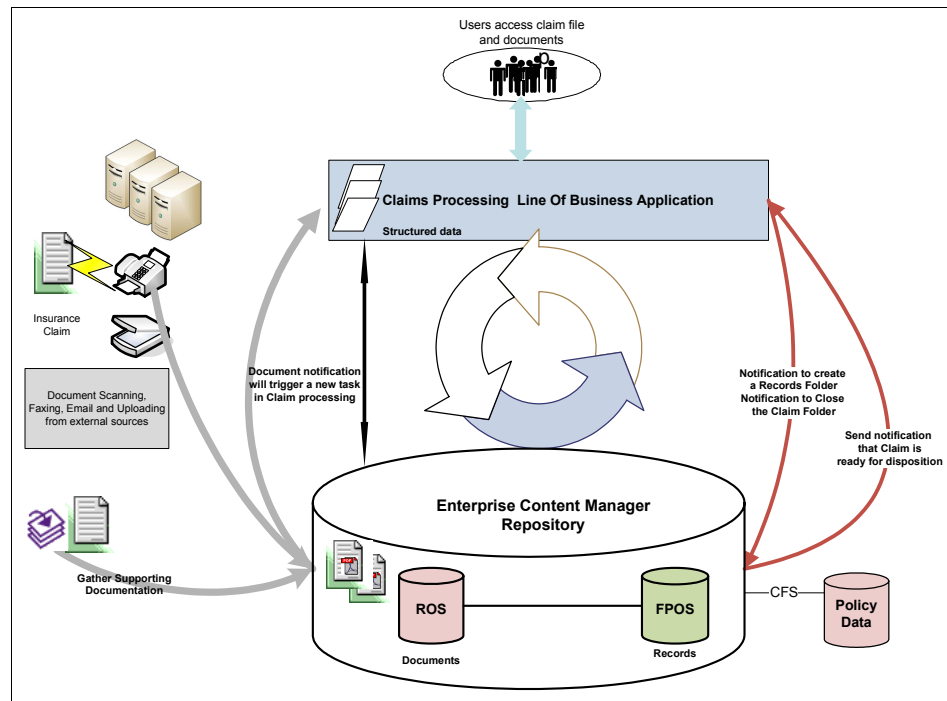


Figure 2-2 Insurance claim processing

Solution description

In this solution, the active content is the insurance claims. The content is moved through a business process in a series of steps implemented in the LOB application (claim processing application).

From the claim processing application, a records folder is opened in the ECM system for each new claim. IBM Datacap converts the paper documents received from the client to electronic documents and stores them into the ECM repository. Datacap also extracts the metadata from the documents using ICR and automatically indexes the documents without user interaction. A record is declared for the documents that are stored in the ECM repository and the retention period is set based on the business requirements. The LOB application utilizes the features of the ECM system through the integration capabilities of the P8 Content Manager. The documents can be retrieved and viewed from different sources that are transparent to the business user.

2.1.3 Archiving SAP invoices

This is a cross-industry sample P8 Content Manager solution. Large organizations have to process a large volume of paper-based invoices. They have to place them in a system to track and control invoice processing.

A major challenge is the large volume of information that must be processed, indexed, and stored. Another challenge is the requirement for fast response and high availability necessary for the accounting department. To address these challenges, the solution includes load-balanced server farms to achieve high ingestion and response rates.

The flow of events for this sample solution follows:

1. Upon invoice arrival, the paper invoices are scanned using IBM Datacap.
2. Data from the invoice is extracted using IBM Datacap OCR/ICR capabilities.
3. Extracted data is verified against the SAP system. If data cannot be verified, an exception handling process is launched.
4. The invoice is created in the SAP system.
5. Invoice images are stored in the P8 Content Manager indexed with the invoice information and linked to the SAP transaction.
6. An invoice's documents can be retrieved and viewed directly from the SAP system
7. Authorized users can search for the invoices using IBM Content Navigator and view the invoice images, as well as directly use SAP to view invoices associated with the transaction record.

This P8 Content Manager solution is implemented with the following features and components:

- ▶ IBM Datacap
- ▶ IBM Content Collector for SAP
- ▶ P8 Content Manager security
- ▶ P8 Content Manager server farm
- ▶ High performance search operation (load balancer)
- ▶ Scalability

Figure 2-3 on page 24 illustrates the implemented invoice processing solution using P8 Content Manager.

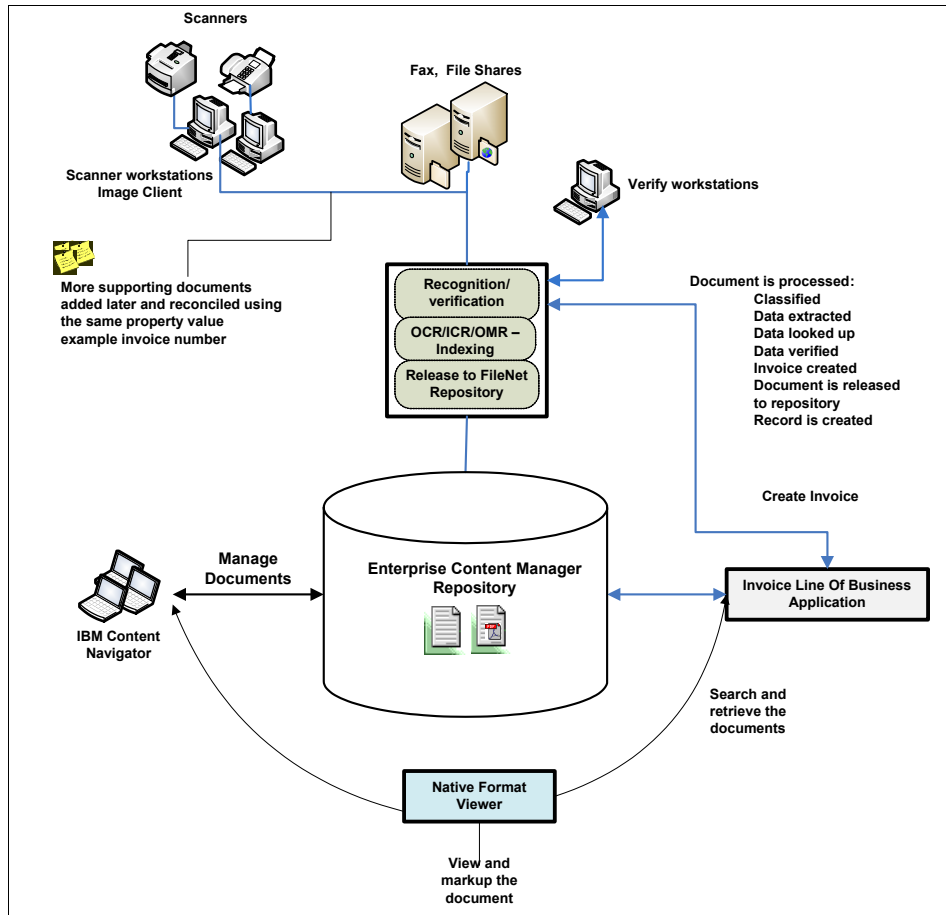


Figure 2-3 SAP archiving solution

Solution description

This solution utilizes the following P8 Content Manager components and capabilities: IBM Datacap Invoice Capture, IBM Content Collector for SAP, and server farms.

IBM Datacap for Invoice Capture

IBM Datacap for Invoice Capture is a solution for accounts payable automation. Extract invoice data using OCR to reduce manual data entry. Apply advanced validation database lookups, calculations, and checksum to verify accurate data recognition. It stores the data and document to IBM Content Manager.

IBM Content Collector for SAP

IBM Content Collector for SAP integrates Filenet Content Manager with the SAP system. It improves the efficiency of SAP business processes by linking content to SAP transactions. It supports both SAP Archivelink and SAP Netweaver Information Lifecycle management protocols.

Server farms

For applications with high-volume loads, P8 Content Manager can be configured as a server farm. A *server farm* employs multiple servers to multiply processing power. In this solution, three P8 Content Manager servers are deployed to spread the document processing load across three separate P8 Content Manager servers. A load balancer spreads the incoming load evenly so that even a high ingestion rate, the load does not overload a single server. In a similar fashion, searches and document retrieval requests are managed by a load balancer on the call center side.

Server farms can also be configured in highly available configurations. Refer to Chapter 3, “System architecture” on page 37 for more details.

2.1.4 Email capture for compliance

This solution addresses recent industry concerns about legal discovery of email messages. It uses IBM Content Collector for Email and P8 Content Manager to capture emails directly from email server journals.

This P8 Content Manager solution is implemented with the following features and components:

- ▶ IBM Content Collector for Email with rule-based automation
- ▶ IBM Enterprise Records

Figure 2-4 on page 26 illustrates the implemented email capture for compliance solution using P8 Content Manager.

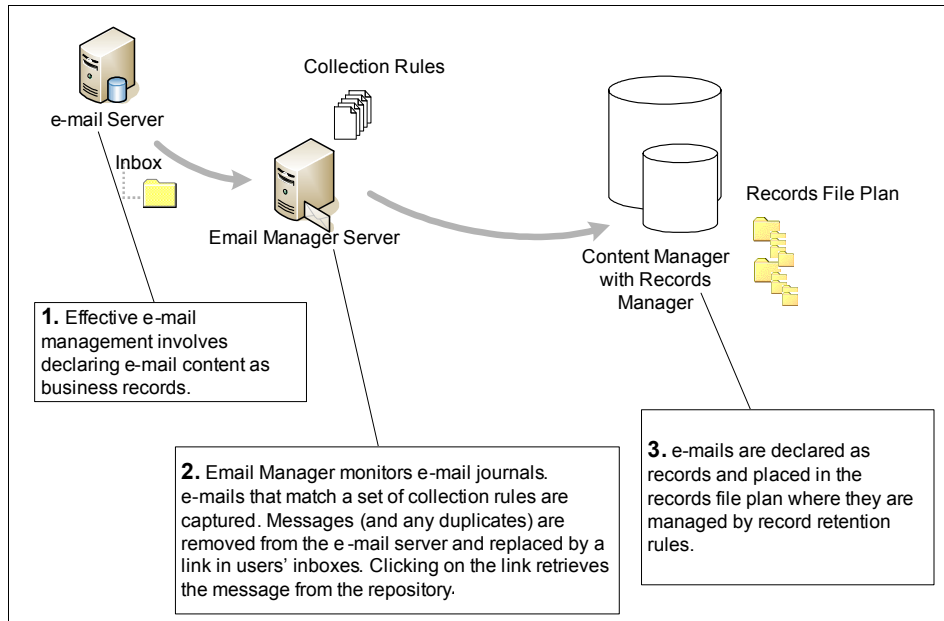


Figure 2-4 Email capture for compliance

Solution description

This solution uses IBM Content Collector to monitor an Exchange Server Journal (IBM Lotus Notes® and other email systems are also supported). The journal contains a copy of all incoming and outgoing messages. IBM Content Collector for Email monitors the journal and searches for messages that meet a set of conditions or rules. Common conditions include:

- ▶ Messages that contain particular keywords
- ▶ Messages to or from a particular set of addresses
- ▶ Messages that pertain to compliance issues raised by the legal department

Messages that meet the set of conditions or rules are treated this way:

- ▶ The message is captured and added to P8 Content Manager.
- ▶ Duplicates of the message (if the message was sent to multiple recipients) are identified. Only one copy is added to the repository.
- ▶ Selected messages based on business rules (for example, emails to a specific address or containing a combination of keywords on the subject or mail body) are classified and declared as an official record subject to legal retention rules.

- ▶ In the user's mailbox, the message is replaced by a stub. When the user clicks the stub, the message is retrieved from the repository and displayed in Outlook as expected.
- ▶ When the records retention period expires, the content is destroyed with no ability to restore it.

2.1.5 Knowledge management through collaboration

In this sample solution, an organization needs to develop and manage the internal training material. They need a solution to provide relevant training information to users according to their job description function and preferences. Organization employees need to collaborate over the training material by starting discussions following updates of a training material and rate the training material. The flow of the events for this use case follows:

- ▶ Authors create training material that might include documents, videos, and audio and stores it in Content Manager.
- ▶ Training material becomes available in a Lotus Connections community for the subject matter experts' (SME) review.
- ▶ SMEs that are participating in that community are collaborating over the material where they can put comments, add tags, and provide feedback over the content.
- ▶ After the training material is finalized, the content becomes available to user communities that require training.
- ▶ In that community, users can add comments and view activity streams, download counts, and recommendation counts.

Figure 2-5 on page 28 illustrates the implemented knowledge management through a collaboration solution using P8 Content Manager.

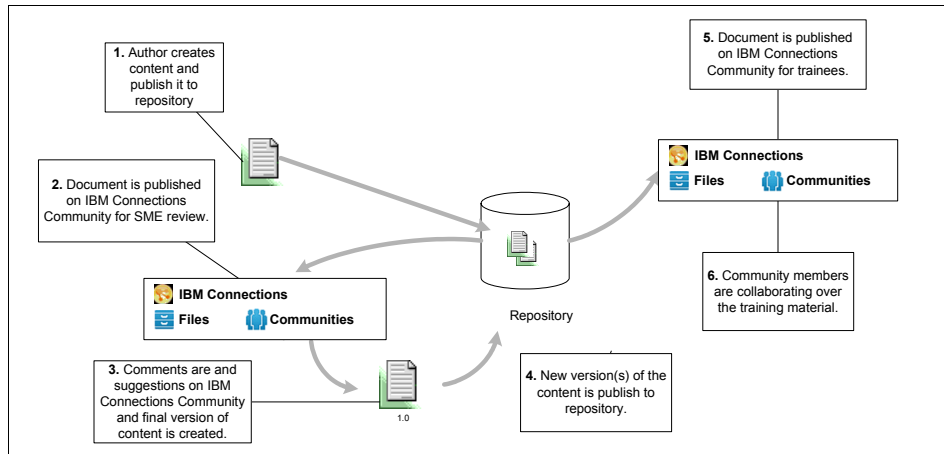


Figure 2-5 Knowledge management through collaboration

Solution description

In this solution, the content that is managed is the training material of the organization. Training material can include documents, presentations, video, and audio. Content authors and SMEs need to collaborate during material preparation. Trainees need to follow the training material, put in comments, and view activity streams, recommendations, and the number of downloads.

In the first step, content is stored in Content Manager along with metadata and is available through the IBM Connections community for review by the SMEs. Using IBM Connections, a SME can add comments, put in recommendations, monitor the activity stream over the content, and upload new versions in the Content Manager repository. When the review process is completed, the finalized content is available on the user community.

Using IBM Connections, a trainee can view new content, add tags, comments, and recommendations, and follow the content. Content Manager provides activity feeds, recommendation counts, download counts, and tag searching, enabling users to locate training content and find relevant content in their social network.

2.2 Design methodology

In the previous section, we present four common P8 Content Manager solutions implemented at client sites. In this section, we introduce the methodology for creating the solution designs to help you with your implementation process.

From our experience, successful design and implementation follow a well-thought-out, repeatable process. Enterprise content management projects are complex, involving the installation, configuration, and customization of a mixture of hardware, software, network resources, content analysis, and process policy. The projects often cross organizational boundaries and involve teams from the IT department, legal department, Quality Assurance department, and others. An organized approach is necessary for a successful implementation.

We outline the recommended design methodology that has been used by many IBM Enterprise Content Management architects in the field and has been applied successfully in many client situations.

The process starts with a top-down approach:

1. Define the ECM strategy.
2. Define the requirements.
3. Create a functional design document from the requirements.
4. Design the systems architecture.
5. Design the P8 Content Manager repository.
6. Design the document security model.
7. Build applications and user interfaces.
8. Create the system and application testing plan.
9. Create a deployment plan.
10. Plan for administration and maintenance.

The remaining chapters of this book address the concepts and recommendations for each step in the methodology. When you design a P8 Content Manager solution, we recommend following the chapters in the book and using our suggestions and recommendations to meet your design challenges.

2.2.1 Defining ECM strategy

The main purpose of an ECM strategy is to establish the decision making principles and standards for the use of content as a business resource. To develop an ECM strategy, a thorough understanding of the business and the underlying information needs, how to structure the information resources to support those needs, and how to manage and maintain the architecture is required.

ECM strategy definition is a critical step in the process of the development of an ECM system. Typically, in the ECM strategy definition phase, you need to define the high-level requirements of the organization for the ECM system and lay out a plan for the successful adoption of the ECM system by the organization. The ECM strategy defines the organization's vision for the ECM system and consolidates the different departments' requirements and directions.

The ECM strategy will establish a common picture of the ECM future that clarifies the business units' directions, aligns actions to the correct directions, and coordinates various initiatives. The organization needs to have a clear understanding and establish an ECM vision to pursue core ECM elements and any adding advanced capabilities.

In our experience, the following elements of the ECM strategy are key:

- ▶ ECM standards
- ▶ Security standards
- ▶ Content taxonomy and classification
- ▶ Content inventory
- ▶ Change management plan for the adoption of the ECM system

ECM requires a change and people management approach that will support every business group and requires a managed business process change and large-scale adoption of the technical solutions. The fundamental challenge of rolling out an ECM system is that it requires the active involvement and participation of all users.

Effective ECM strategy eliminates information silos, enables the foundational ECM elements, such as the enforcement of the information governance and support for business requirements, and enables the integration and automation of business processes.

2.2.2 Requirements analysis

The first step is to gather and analyze the requirements for the development of an ECM system. From our experience, ECM systems evolve and grow according to the organizational needs and business challenges. So, the requirements gathering is an iterative process. It is important to start the requirements gathering as early as possible and aligning those requirements with the ECM strategy. If you miss a critical requirement, the entire system might be flawed. It is important to gather input from all groups that will be involved in using, building, testing, training, or operating the system.

For each iteration of requirements gathering, create a requirements document. As a milestone goal, create and review this document and obtain the appropriate sign-offs on the document.

A requirements document needs to include the following information:

- ▶ Functional requirements - What must the system accomplish?
- ▶ Non-functional requirements - To what must the system adhere?
- ▶ Hardware - What standards or limitations apply to hardware specifications?
- ▶ Software - What standards apply to software specifications?

- ▶ Security - What are the security requirements for the system?
- ▶ Retention - What are the retention requirements for the system?
- ▶ Usability - What ease of use standards apply?
- ▶ Performance - What levels of response times are required?
- ▶ Continuity - What service levels are expected?
- ▶ Documentation - What training and documentation are required?

Typically, requirements gathering is an iterative process. You will start with the functional design, revisit the requirements, complete the requirements analysis, and then revisit the functional design. This process continues until you feel confident that all known requirements have been identified and addressed.

The enterprise content management illustration in Figure 2-6 helps in structuring the discussion among the various people involved in understanding what functionality the solution requires to solve specific business problems.

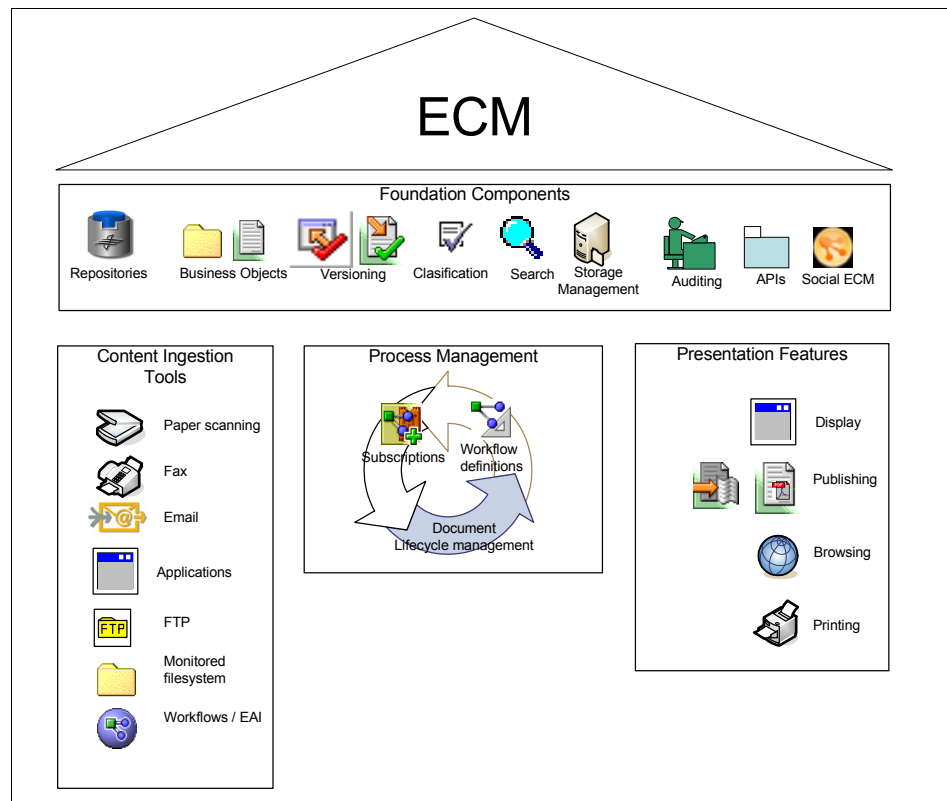


Figure 2-6 A simple input and output diagram to assess functional requirements

For example, based on Figure 2-6 on page 31, you can discuss the following points to help you gather functional requirements:

- ▶ How will content be ingested? Options might be paper scanning, fax, email, other applications, FTP, monitored file system, or workflows.
- ▶ What are the content and workflow management requirements? Considerations can be indexing, validation, the addition of document management, security, binding of documents, usage of entry templates, and check-in and check-out features.
- ▶ What are your presentation and delivery management requirements? Consider content-based search, publishing, and browsing requirements; printing needs; display needs; Simple Mail Transfer Protocol (SMTP) sends; and the requirement for the usage of search templates.
- ▶ What indexing terms will be used to identify and retrieve information?
- ▶ What interfaces are required with existing systems to move information between systems?
- ▶ What is the lifecycle of the document and how does it fit as part of the business process?
- ▶ How will auditing requirements be satisfied?
- ▶ How will security or information confidentiality requirements be met?
- ▶ How will the information be searched across the enterprise?
- ▶ What is the minimum required to describe the metadata?
- ▶ How long will the documents need to be stored in the system?
- ▶ Who will be responsible for the destruction of the documents?
- ▶ What type of storage will be required?

Working from the requirements analysis to a functional design is probably one of the biggest challenges in your project. This activity requires extensive experience and solid knowledge of the P8 Content Manager product. We address this topic in more detail in Chapter 13, “IBM FileNet Content Manager solutions” on page 427.

As you read through each chapter of this book, remember that each chapter provides many of the best practices for a number of scenarios but in a generalized way. Use these best practices and recommendations within the context of the actual functional requirements for your solution; do not apply them as is.

2.2.3 Functional design

A functional design shows the components of the systems and describes how each component handles information objects. The functional design needs to match the user requirements and be presented to the project team for review.

Functional design will define the overall solution architecture, such as taxonomy, and the folder and data structure design. A solution model is created to support sample use cases. Analysis and design are crucial in the whole development cycle. Any mistake in the design phase can be expensive to solve in a later stage.

As mentioned in 2.2.2, “Requirements analysis” on page 30, the functional design step is often part of an iterative solution design process. Chapter 13, “IBM FileNet Content Manager solutions” on page 427 covers this topic in more detail.

2.2.4 System architecture design

The system architecture design (sometimes called a *logical design*) lays out the setup of the hardware and software components. The system architecture is a blueprint for system infrastructure construction. A logical design shows servers (both hardware and software), network connections, storage units, and database instances. When creating your system architecture design, consider the following elements:

- ▶ Server topology
- ▶ Network (LAN and WAN) topology
- ▶ Scalability and continuity
- ▶ Virtualization
- ▶ Shared infrastructure
- ▶ Capacity planning
- ▶ Performance

Although system architecture can be derived from nonfunctional requirements, it can be influenced by the functional requirements. Often, system architecture is dependent on certain decisions made in the functional design.

During this phase, the major architectural decisions must be documented. Architectural decisions are significant because they might directly or indirectly determine whether a system meets its nonfunctional requirements, such as software quality attributes. Harmonizing the architectural decision-capturing process simplifies reusing and learning from architectural decisions made on previous projects. Architectural decisions need to be identified, made, and enforced systematically.

For more information regarding system architecture, business continuity, and capacity planning, refer to the following chapters:

- ▶ Chapter 3, “System architecture” on page 37
- ▶ Chapter 7, “Business continuity” on page 217
- ▶ Chapter 8, “Capacity planning with IBM Content Capacity Planner” on page 253.

2.2.5 Repository design

The repository design is the key design step in a P8 Content Manager project. It specifies the number, type, and structure of the solution repositories. It defines the object classes that will be stored in the repositories, including the metadata, folder storage, security descriptors, and retention requirements for each type of content object.

The repository design typically is tightly linked to the functional design. It affects and is affected by the security design. The repository design must be carefully synchronized with the application and security design.

For more details, refer to Chapter 4, “Repository design” on page 81.

2.2.6 Security model design

You can enforce security through repository design and application design. A dependency exists between application security constraints and the security mechanisms applied on the repository. P8 Content Manager offers a rich set of options for developing a security model. Our suggestion is to utilize the repository security features as much as possible.

ECM Security Framework controls access to the information from the initial classification through to disposition, covering all aspects of information lifecycle management, and simplifies and speeds up ECM planning and execution.

Determine which users and groups will have access and the levels of access that these users and groups will have. Create these users and groups in the authentication provider if they do not exist there.

The security model for ECM needs to be clearly defined and it needs to be understood that security access for documents that are transitory records might differ from the documents that will become business records.

The Security Framework needs to address the principles for security and user roles mapping between different ECM modules. For more details, refer to Chapter 5, “Security” on page 151.

2.2.7 Application design

Application design is mainly derived from the functional design and must be synchronized with the repository and security design. The application design includes user interfaces and custom software components. The design presents the details of application features and functionality and specifies the application programming interface (API) that developers will use to construct the application.

ECM solution design needs to support reusability by creating or using existing shared components. For more details, refer to Chapter 6, “Application design” on page 191.

2.2.8 Test planning

Testing is a required step before the deployment of an ECM solution to the production environment. There are many types of tests that a solution has to pass in order to be a successful one.

The most common types of tests are:

- ▶ User Acceptance Tests (UAT)
- ▶ Regression tests
- ▶ Performance tests and load tests
- ▶ Backup and recovery tests

User acceptance tests verify that the system meets the functional requirements described in the functional design document.

Regression tests verify that the functionality of the solution is not affected after the implementation of new features or the resolution of a defect. It is important to have a full set of regression tests for the ECM solutions and we suggest that those tests are automated.

Performance and load tests verify the system responsiveness and stability under a defined workload. It is a best practice to develop performance tests and execute them before any major release is put on the production environment.

Backup and recovery tests verify that the system can be successfully recovered in case of disaster. You need to perform at least one recovery test a year to verify that your backup and recovery plan is still valid.

2.2.9 Deployment

Deployment is defined as the methodology to move a designed solution from development to production. When planning for deployment, issues related to

release management, change management, testing, and the steps for the actual move need to be considered. It is important to plan for deployment as early as possible, especially at development time, to address many of the challenges that might arise in this area.

For more details, refer to Chapter 9, “Deployment” on page 271. Also, refer to Chapter 11, “Upgrade and migration” on page 371 for upgrade and migration information.

2.2.10 Maintenance planning

Maintenance is related to operational aspects, such as system monitoring, backup and restore, and other tasks. Capacity planning might also be considered as part of your maintenance planning activity.

It is a best practice to have four environments where one of the environments, User Acceptance Testing (UAT), is a replica of the production environment. The replica includes version, configuration, topology, and data to ensure that the applications and any upgrades are sufficiently tested before being deployed to production.

For more details, refer to Chapter 10, “System administration and maintenance” on page 315. Also, refer to Chapter 12, “Troubleshooting” on page 387 for troubleshooting information.

2.3 Conclusion

In this chapter, we introduced five typical use cases where Content Manager can be used. Those use cases are simplified and the actual use cases are much more complicated. The purpose of those examples is to make a quick introduction of Content Manager features and capabilities. The detailed solution building blocks and how they are used for the implementation of those use cases are described in Chapter 13, “IBM FileNet Content Manager solutions” on page 427.

We also described the proposed methodology for the implementation of an ECM system using Content Manager. That methodology is used as a structure for the rest of the book.



System architecture

IBM FileNet Content Manager is a collection of tightly integrated components that are bundled together as a common platform. The broad functionality provided by these integrated components constitutes an enterprise content and process management platform. Some of the key elements of this platform are a metadata repository, a workflow system, an application that can be used by all clients to access content and participate in processes, and a storage framework that supports a wide range of storage devices and platforms.

In this chapter, we introduce the components of an IBM FileNet Content Manager system, discuss the logical and physical layout options, and discuss how to scale the system to meet both local and global business needs.

We discuss the following topics:

- ▶ Basic components
- ▶ Scalability
- ▶ Virtualization
- ▶ Shared infrastructure
- ▶ Geographically distributed systems

For information about the internal system architecture of IBM FileNet Content Manager and the P8 Platform, see *IBM FileNet P8 Platform and Architecture*, SG24-7667.

3.1 Basic components

An IBM FileNet Content Manager system typically includes the following components:

- ▶ Content Platform Engine
A collection of services and components for managing different types of business-related content and automating business processes.
- ▶ FileNet Workplace XT (optional)
A presentation-tier application for working with both content and processes. It also provides administrative tools for designing and managing processes. FileNet Workplace XT is a useful tool for performing a quick validation of an IBM FileNet Content Manager installation, and for reproducing issues seen with custom applications.
- ▶ IBM Content Navigator (optional)
An extensible presentation tier for working with both content and processes. Like FileNet Workplace XT, IBM Content Navigator can also be used to validate an IBM FileNet Content Manager installation and to replicate issues seen with custom applications. For more information about IBM Content Navigator, see *Customizing and Extending IBM Content Navigator*, SG24-8055.
- ▶ Directory service
A Lightweight Directory Access Protocol (LDAP) directory server, frequently the corporate LDAP server, used by IBM FileNet Content Manager to authenticate users and to access authorization information for artifacts managed by the Content Platform Engine.
- ▶ Database server
Hosts the databases used to store the metadata associated with the business content and processes, and configuration information for an IBM FileNet Content Manager implementation.
- ▶ Application server
Hosts the Content Platform Engine and other Java EE applications, including FileNet Workplace XT and IBM Content Navigator, that are used as part of an IBM FileNet Content Manager implementation.
- ▶ Storage
Database, file, or fixed storage areas for storing the business content.

Figure 3-1 on page 39 illustrates a basic layout of an IBM FileNet Content Manager environment.

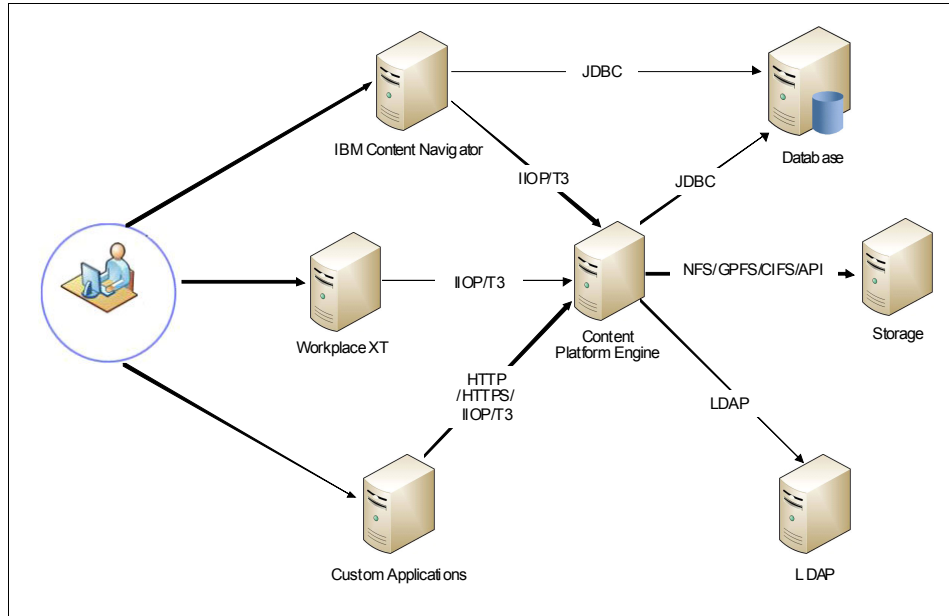


Figure 3-1 Basic components of P8 Content Manager configuration

Although FileNet Workplace XT and Content Navigator are popular application components, they are not absolutely required for a functioning IBM FileNet Content Manager system.

3.1.1 Additional components

In addition to the basic components, an IBM FileNet Content Manager environment can be configured to include the following components:

- ▶ Content Search Services (CSS)
 - Allows you to index and search for text within documents and in the metadata that is used to describe a document
- ▶ Rendition Engine
 - Is used to generate PDF or HTML versions of documents

3.1.2 Data organization

The business content and processes associated with an IBM FileNet Content Manager implementation are organized into a logical grouping called a *P8 domain*. The definition of the P8 domain is stored in a database known as the *global configuration database* (GCD).

The GCD stores information about the following items:

- ▶ Object stores and any workflow systems that have been defined for the P8 domain

A P8 domain can have one or more content stores, known as *object stores*, and one or more *workflow systems*. The role of an object store is to store business-related content. A workflow system is a repository for storing and managing process-related information.

- ▶ Configuration information that is common to all object stores and workflow systems, such as the LDAP configuration used with the environment
- ▶ Components that are available for reuse within the environment, such as fixed content storage devices and marking sets

Fixed content storage devices are a specific class of storage device, for example, IBM Tivoli® Storage Manager that is defined once within a P8 domain, but that can be used by any or all of the object stores to store content.

A *marking set* is a special construct used to enhance the security model that can be applied to an object store.

- ▶ Logical grouping of components to support distributed configurations

Figure 3-2 illustrates the logical relationship between a P8 domain, and its associated configuration information, object stores, and workflow systems.

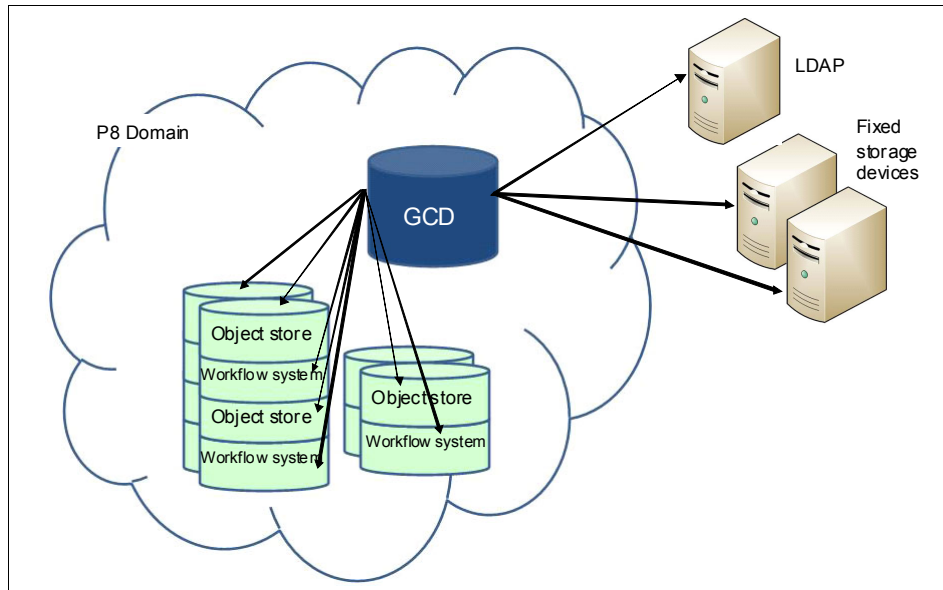


Figure 3-2 P8 domain

Figure 3-3 illustrates the relationship between the components that are used by IBM FileNet Content Manager to instantiate a P8 domain and the software components that comprise the P8 domain.

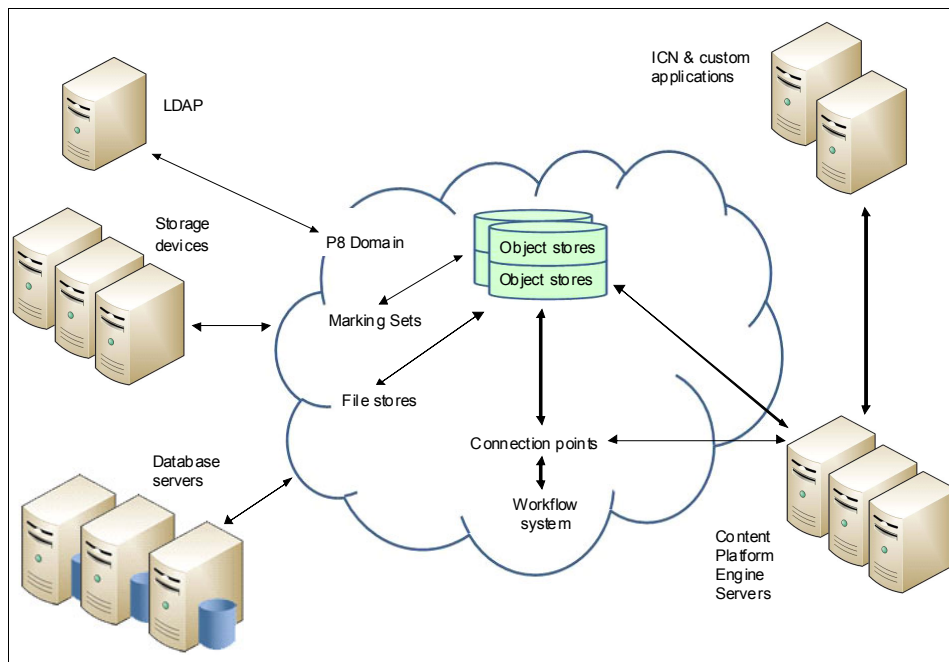


Figure 3-3 Relationship between P8 domain and supporting components

3.1.3 Object stores

An object store manages business content. Each object store manages:

- ▶ A database for metadata
 - The metadata describes the artifacts that are stored in the object store.
- ▶ One or many storage areas that represent the physical storage area location of the business content
 - The storage area can be in a database, a file system, a fixed content device, such as IBM Tivoli Storage Manager, Network Appliance SnapLock, or EMC Centera, or a combination of these options.

A P8 domain can have many object stores. The number of object stores in a domain, and the physical location of the database and storage areas associated with an object store are dependent on your business needs and will be discussed in more detail later in this chapter.

3.1.4 Storage considerations

When creating an object store, you can choose to utilize database storage or storage areas external to the database.

If you choose:

- ▶ Database storage, all metadata and document content is stored in the object store database

Use this option when creating object stores that will have little document content. Object stores that fit into this category include ones that are used to store only configuration information, or file plans for IBM Enterprise Records implementations.

- ▶ External storage, metadata is stored in the object store database, but the document content is stored externally to the database in a shared file system or on a fixed content device, such as IBM Tivoli Storage Manager

For a full list of supported storage types, see the IBM FileNet P8 Hardware and Software Requirements guide:

<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27013654>

Although the type of storage you use is likely to depend on corporate standards, various storage features also drive your decision-making process.

Where and how content is stored are defined using storage policies. Storage policies can be set up at the object store level, the document class level, and on individual documents. For information about defining storage policies, see the following link in the P8 Information Center:

http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.admin.tasks.doc%2Fcontentstores%2Fcs_stp_about_storage_policies.htm

Minimizing storage requirements

IBM FileNet Content Manager provides the following features to help reduce storage requirements:

- ▶ Retention management
- ▶ Deduplication
- ▶ Compression

Retention management

Retention management defines how long content must be kept before it can be deleted. With retention management, a date is set that defines the earliest that content can be deleted. Retention management differs from records management in that it is usually driven by corporate rather than legal requirements.

In IBM FileNet Content Manager v5.1 and earlier releases, support was provided for static retention periods. That is, when a document was added to an object store, you were able to define the minimum period that the document must be kept. Now, IBM FileNet Content Manager also supports dynamic retention.

With dynamic retention, the length of time a document must be kept can be updated at any time by users with the appropriate level of permissions.

Defining retention periods helps with storage management only if you also regularly delete content that has passed its retention requirement. Set up regular jobs to delete content that is no longer needed as described in 4.11.2, “Automatic disposition” on page 139.

Deduplication

Preventing duplicate copies of the same content from being stored in an object store can save a significant amount of space.

A common scenario where this feature is beneficial is when email is archived into an object store via IBM Content Collector. Often, several clients will receive the same email with the same attachments. If the content is stored in a storage area, enabling deduplication will ensure that only one copy of the content is stored. The IBM FileNet Content Manager software tracks the documents associated with the content, and will not delete the content until all the associated documents have been deleted.

Deduplication occurs at the content rather than block level. So, additional space savings can be achieved by also using compression.

Compression

Two types of compression are available:

- ▶ If you are using IBM DB2®, consider enabling row compression on large object store database tables to improve application response times.
- ▶ You can also choose to compress storage areas. The compression is block-based and, depending on the type of files being stored, might not provide a significant savings. The overhead introduced with this capability can affect both upload and download performance, so test this feature in your environment to ensure that any performance impact is offset by the space savings.

Reducing storage costs

You can reduce storage costs by moving less frequently accessed content to less expensive forms of media. You need to define what is “less expensive” for your organization. For example, some organizations have different tiers of storage so that content that is accessed frequently is stored on “tier 1” storage, while less frequently accessed content is stored on “tier 2” storage. The requirement for a document to be on “tier 1” versus “tier 2” content can change over the life of the document.

IBM FileNet Content Manager provides APIs for moving content from one form of storage to another. To take advantage of this capability, make sure the object store design enables you to identify easily the types of documents that can be moved and when they need to be moved.

Securing content

When content is stored, you can encrypt the data prior to it being stored on the underlying storage (file store, fixed content device, or database).

The requirement for data encryption is set at the storage container level and can be enabled or disabled at any time. When the encryption is enabled, any data added to the storage device will be encrypted, but existing content is not affected by enabling or disabling the encryption capability.

The overhead introduced with this capability can affect both upload and download performance.

Data encryption can be used with the other storage features, including data compression and deduplication.

3.1.5 Workflow systems

A *workflow system* manages the process-related data. Like object stores, there can be multiple workflow systems in a single P8 domain, but the database tables used to instantiate a workflow system must be collocated with a specific object store database.

Within the workflow system, the processes run inside of an isolated region that acts as an individual processing space. Workflow systems can contain multiple isolated regions, and each isolated region can contain multiple workflow definitions and related data. And although workflows can use content from any object store in the P8 domain, the workflow processing cannot span multiple isolated regions.

For ease of maintenance, a best practice is to segregate isolated regions into separate workflow systems and to associate each workflow system with a separate object store.

Applications connect to an isolated region using a *connection point*. Connection points are defined at the P8 domain level using the IBM Administration Console for Content Platform Engine (ACCE).

Notes:

- ▶ The concept of a workflow system is no different than the workflow database of earlier IBM FileNet Content Manager releases.
- ▶ Prior to IBM FileNet Content Manager v5.2, the process-related data can be collocated with an object store, or it can be stored in a separate database.
- ▶ Clients upgrading from IBM FileNet Content Manager v5.1 and earlier releases are not required to merge process databases with object store databases.

For more information about using business processes with IBM FileNet Content Manager, see *Introducing IBM FileNet Business Process Manager*, SG24-7509.

3.1.6 Management tools

Various tools are needed to build and manage an IBM FileNet Content Manager environment. Some of the tools are supplied as part of the IBM FileNet Content Manager software, and other tools are provided by the vendors of the infrastructure components (database, application server, and LDAP) that are used with IBM FileNet Content Manager.

For example, prior to configuring the Content Platform Engine, use the tools supplied by the database vendor to create the databases and tables required for the GCD, at least one object store, and at least one workflow system. IBM Content Navigator also requires a database in which to store configuration information.

Recommendations: Although creating a workflow system is not required when setting up an IBM FileNet Content Manager environment, consider creating an initial workflow system and validating basic workflow functionality when configuring the Combined Platform Engine.

After the basic IBM FileNet Content Manager installation and configuration are complete, the database tools will also be needed for tasks, such as backing up the environment, and creating complex indexes to improve performance.

See the FileNet P8 Information Center for a full list of the tasks that need to be completed before starting an IBM FileNet Content Manager installation:

<http://publib.boulder.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8toc.doc%2Fplanning.htm>

The following administrative tools are supplied with IBM FileNet Content Manager:

► Configuration Manager

Running the Content Platform Engine installation program lays down the software but does not perform any configuration tasks. So, after you run the installation program, use the Configuration Manager to identify the following information:

- The application server to host the Content Platform Engine application and services.
- The LDAP server that will provide the authentication and authorization services required by IBM FileNet Content Manager.

Authentication defines who can access the system. *Authorization* identifies what clients are allowed to do after they have been authenticated.

- Data sources that identify the databases to be used for the GCD, object stores, and workflow systems.

The Configuration Manager is also used to deploy the Content Platform Engine application ear or war file.

- ▶ ACCE

A web-based tool for defining P8 domain and object store configuration data. ACCE replaces the FileNet Enterprise Manager tool that was provided with previous IBM FileNet Content Manager releases.
- ▶ Deployment Manager

A thick client tool used to move object store configuration data and content between object stores in the same or different P8 domains. This tool is also used to reassign object stores to different P8 domains.

Chapter 9, “Deployment” on page 271 in this Redbooks publication provides detailed information about using the Deployment Manager.
- ▶ Consistency Checker

Used to validate that pointers in the object store database that reference content in a storage area are correct.
- ▶ FileNet Enterprise Manager

A thick client tool used to administer P8 domains and object stores. This tool is being replaced by ACCE.
- ▶ Process Configuration Console, Process Administrator, Process Designer, and Tracker

These are Java applets that are started from FileNet Workplace XT and that are used to instantiate a process region, and to design and manage business processes.

3.1.7 Bulk Import Tool

The Bulk Import Tool is installed by the Content Platform Engine installer and is used to add large volumes of documents to an object store. This tool is used only if you have purchased the IBM Production Imaging Edition license.

For more information about using the Bulk Import Tool, see the P8 Information Center:

<http://publib.boulder.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.blkimpt.doc%2Fp8pit000.htm>

3.1.8 Hardware layout

IBM FileNet Content Manager can be configured for use with both small and large deployments. Each component can be installed on a single or on multiple servers; you can also collocate all the components on a single server. This flexibility allows the environment to be scaled for both performance needs and to support security requirements.

For example, the application tier, provided by products such as IBM Content Navigator, can be installed within a DMZ. However, the Content Platform Engine is located behind firewalls inside the corporate network. Figure 3-4 illustrates separating elements of the system in different infrastructure security layers.

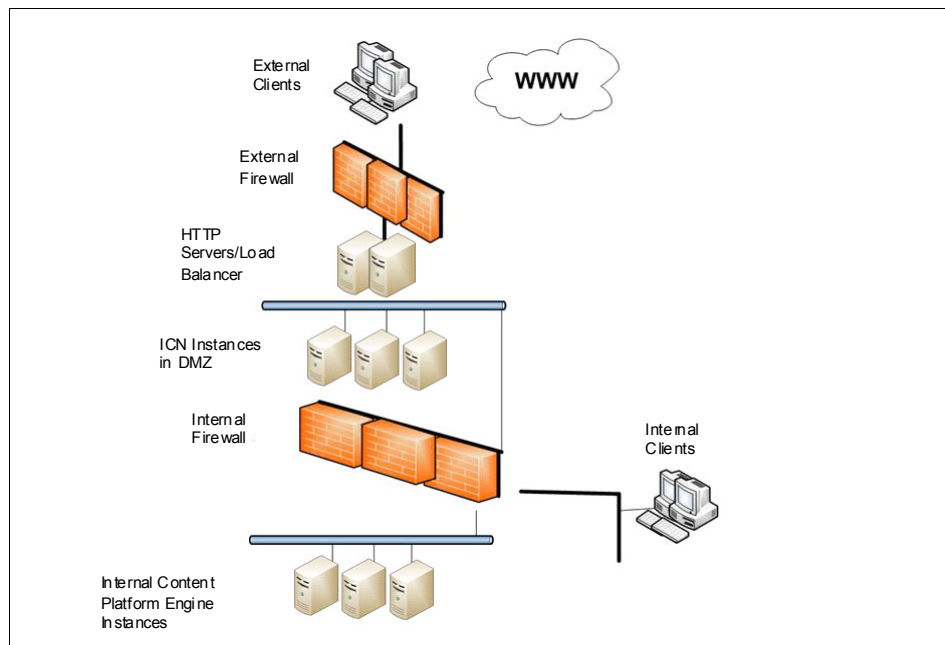


Figure 3-4 IBM FileNet Content Manager configuration based on security requirements

For more details about configuring the hardware layout based on security requirements, see *IBM FileNet P8 Platform and Architecture*, SG24-7667.

3.1.9 Setting up a sandbox or demo environment

Setting up enterprise applications and associated hardware can be both costly and time-consuming. For a quick start and to gain familiarity with a specific release, consider setting up a single-server configuration of IBM FileNet Content

Manager. Use the Composite Platform Installation Tool (CPIT) supplied with IBM FileNet Content Manager to install and configure this environment. CPIT completes the following tasks:

- ▶ Installs and configures:
 - Tivoli Directory Server as the LDAP
 - DB2 as the database server
 - IBM WebSphere® Application Server as the application server
 - Content Platform Engine
 - FileNet Workplace XT
 - IBM Content Navigator
- ▶ Creates:
 - The databases required for the GCD, one content store, one workflow system, and the IBM Content Navigator configuration information
 - Administrative user accounts and groups in the LDAP
 - The GCD
 - One content store
 - One workflow system

After the CPIT installation completes, use the single-server installation to perform these tasks:

- ▶ Gain familiarity with IBM FileNet Content Manager in general, or the latest new features.
- ▶ Define requirements and specifications for new applications or updates to existing applications.
- ▶ Start building a new solution or refining an existing solution.

3.1.10 Using Information Center and other product documentation

IBM FileNet Content Manager ships with documentation that can be deployed on a Java EE server, but to ensure you are using the latest documentation, use the information center published on the IBM website:

<http://publib.boulder.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=/com.ibm.p8toc.doc/ic-homepage.html>

The Information Center has a number of useful features:

- ▶ Setting the search scope to match the components used in your environment

- The ability to print selected topics

For example, use these two features together to generate a custom installation or upgrade guide.

One of the most important guides for anyone setting up or maintaining an IBM FileNet Content Manager environment is the IBM FileNet P8 Hardware and Software Requirements guide. This guide documents the supported types and levels of the infrastructure that can be used with IBM FileNet Content Manager. See the following link to download the guide:

<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27013654>

Figure 3-5 illustrates the Home page of the IBM FileNet P8 Information Center. The information center contains information about the IBM FileNet Content Manager and other components in the P8 Product Family that integrate with IBM FileNet Content Manager.

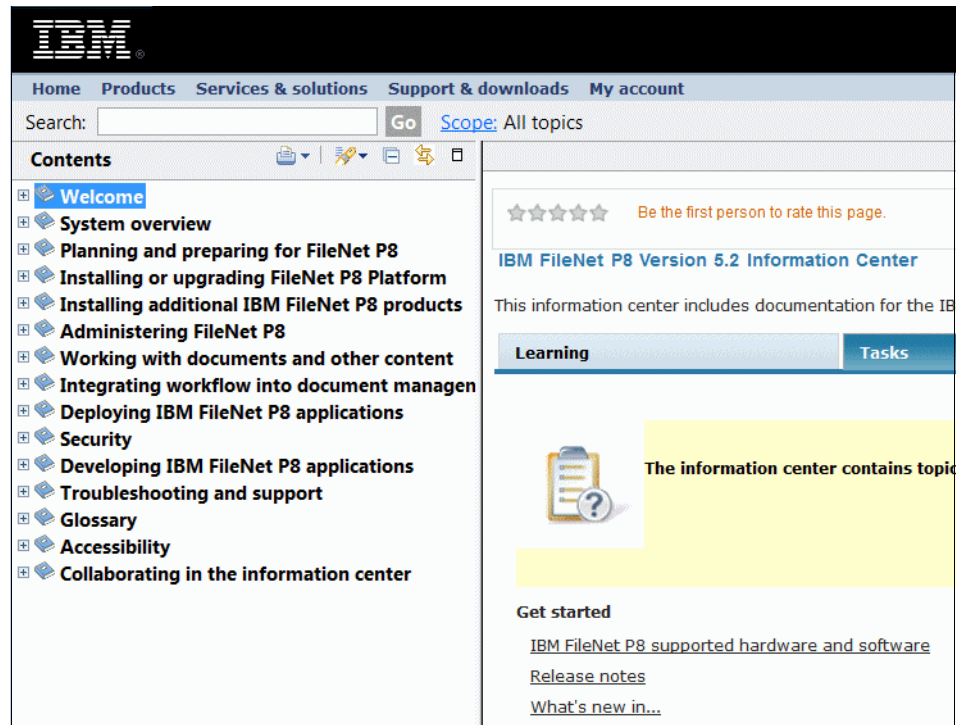


Figure 3-5 IBM FileNet P8 Information Center

3.2 Scalability

When planning for an enterprise-wide system, there are a number of environments to configure, and the workload and scaling requirements for each are likely to be different. Consider configuring the following IBM FileNet Content Manager environments:

- ▶ Sandbox

Typically, a single-server installation that can be used by system architects and project leads to learn about new software releases and for demonstration purposes.

- ▶ Development

The environment used by the development team to build custom applications for an IBM FileNet Content Manager deployment. The size and configuration of this environment is typically driven by the number of developers on the project.

- ▶ System test

The environment used by the test organization and representatives of the client teams to verify that the application being built meets their needs in terms of functionality, usability, and response times.

This environment is also used for integration testing to ensure that elements of a single solution or that multiple enterprise-wide solutions complement each other, can coexist, and interact as expected.

The size and configuration of this environment is usually a scaled-down version of the expected production configuration.

- ▶ Load and performance test

The environment used to ensure that the production system can cope with the expected load and provide clients with response times that enable them to complete their work. The system needs to be configured similarly to the production system in terms of hardware and data (both the type of data and the volume need to emulate what is available in the production environment).

- ▶ Production fix

When introducing new functionality into a production environment, a best practice is to introduce the changes into development, then promote the changes first to system test, then to load and performance test, and ultimately to the production environment. This path is recommended to help ensure that the new functionality meets both the functional and performance needs of a production application.

However, when following this methodology, occasionally an issue occurs in production that cannot be replicated in the lower environments because they are not running the same software as the production environment.

A production fix environment is a scaled-down version of the production environment that is always running the same level of software as that in production. When a production issue occurs, the fix is first introduced into the production fix environment to ensure that the fix resolves the reported issue and does not introduce any new issues. (If appropriate, the fix, although it might be in a slightly different form, needs to also be introduced to the lower environments.)

- ▶ Production

The environment hosting IBM FileNet Content Manager and other applications that provide the required business functionality.

This environment must be built to meet current business needs and be easy to both expand and contract as capacity needs and response time requirements change.

When designing the layout of each environment, consider many elements:

- ▶ Expected volume of activity from both automated applications and clients
- ▶ Types of activities that will be performed
- ▶ Volume of content that will be stored
- ▶ Required response times
- ▶ Geographic locations that must be supported

When planning for a geographically dispersed environment, the amount of hardware needed at each site can be different.

The IBM Capacity Planner is a tool that provides guidelines on “how much” is needed given your business requirements. For more information about this tool and capacity planning in general, refer to Chapter 8, “Capacity planning with IBM Content Capacity Planner” on page 253.

When scaling an IBM FileNet Content Manager environment, you can choose to scale:

- ▶ Hardware using both horizontal and vertical techniques.
- ▶ Java virtual machines (JVMs) using application cluster technology, such as WebSphere Network Deployment, or by adding independent JVMs and load balancers.

3.2.1 Horizontal scalability

Horizontal scaling, also known as *scale-out*, means to add additional computer systems to the existing environment. This approach is common in Microsoft Windows environments. Clients who prefer horizontal scaling distribute applications through a large number of inexpensive machines. Blade servers generally group multiple compact servers in a rack and allow a large number of servers in a small physical space.

3.2.2 Vertical scalability

Vertical scaling, also known as *scale-up*, means to use more powerful servers or extend the existing ones to be more powerful. This approach is frequently used in the mainframe world and is also available in UNIX and Windows environments. An example is adding more CPUs and RAM to the existing system. Vertical scalability is often used with virtualization (see 3.3, “Virtualization” on page 57). Servers are consolidated and multiple previously stand-alone servers are merged and run virtualized on a large machine.

The core IBM FileNet P8 components can be scaled vertically and horizontally. As an extension to vertical scalability, clients can scale up a server that is running an application server instance with multiple deployed applications. The benefit is better application segregation and more effective use of memory resources.

Instead of just scaling up the server with additional hardware, you can use multiple Java EE instances on a single physical server, with each application running independently in its own Java EE instance. By separating the applications, you achieve more efficient use of system resources.

3.2.3 Clustering

Since Content Platform Engine is a Java EE application deployed on an application server, the system can be scaled out further using the clustering technology provided by the application server. For example, using WebSphere Network Deployment, additional application servers can be configured. The Content Platform Engine, FileNet Workplace XT, IBM Content Navigator, and custom applications can be automatically deployed from the WebSphere Network Deployment management node to these additional servers. The advantage of this type of configuration is easier maintenance because software is updated manually in only one location and the deployment to the managed nodes is automatic.

Figure 3-6 on page 55 illustrates this type of clustering.

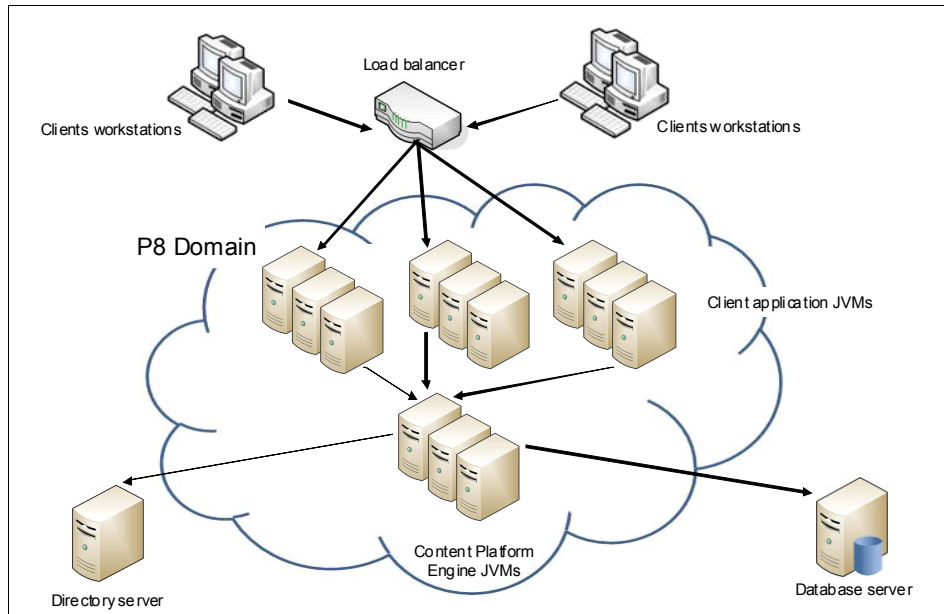


Figure 3-6 Scaling using application server clustering technology

Alternatively, the applications can be deployed independently and load balancers can be used to spread the load between the servers. When using this model, a best practice is to collocate each Content Platform Engine instance with the appropriate client application instance as shown in Figure 3-7 on page 56 to minimize the loss of sessions and resources when a server becomes unavailable. The disadvantage of scaling using this approach is that if either a client application or a Content Platform Engine instance goes down, the client's session is lost and work might need to be resubmitted. However, the stand-alone application server software used in this configuration is usually less expensive than cluster-aware versions of the application server software.

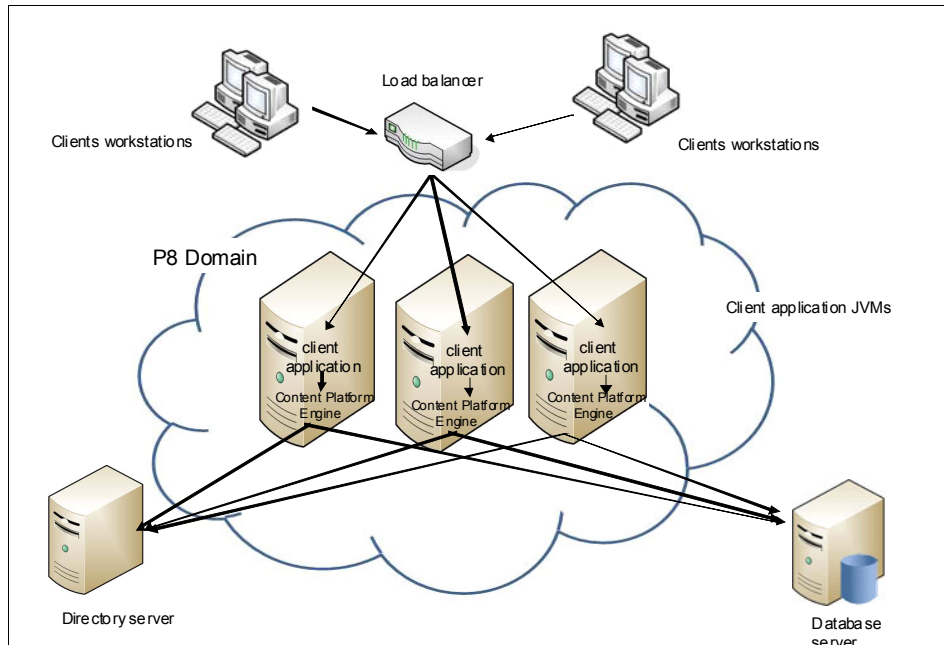


Figure 3-7 Scaling application servers using a stack approach

To distribute the load to the client applications, use an external HTTP server with a load balancer, even when the client applications are installed using an application server clustering technology.

3.2.4 P8 domain and object store scaling

Each P8 domain can contain many object stores, but a practical limit needs to be set based on considerations, such as how long it takes to upgrade to a new IBM FileNet Content Manager release or apply software updates, and the need to roll out applications in different time frames.

There are no specific limits on the number of artifacts in an object store, but from a practical perspective, a limit needs to be identified for each object store. The limit needs to be based on these factors:

- ▶ The time required to perform backups and restores.
- ▶ The efficiency with which indexes can be created and maintained.
- ▶ The performance as perceived by your clients; for example, the larger the data set, the longer it might take to run searches and to add documents.

Other reasons for segregating object stores:

- ▶ In global deployments, legal requirements regarding the physical location of any content and the location of the clients accessing the data
- ▶ Security requirements

In addition, requirements for workflow processing can influence the need for multiple isolated regions, workflow systems, and object stores.

The IBM FileNet Content Manager makes it easy to add new object stores to an existing P8 domain, and the FileNet Deployment Manager tool also makes it simple to move object stores to new P8 domains. However, to take advantage of these capabilities, custom applications must be designed to accommodate these features:

- ▶ When ingesting content, the destination object store must be configurable.
- ▶ When searching for content, the search needs to be capable of looking in multiple object stores.

3.2.5 Scaling Content Search Services

When scaling Content Search Services (CSS), you must address both the required indexing throughput and the client search requirements. Like the other components of IBM FileNet Content Manager, the CSS scaling is achieved using both horizontal and vertical scaling of the CSS servers.

Indexing is a CPU-intensive and memory-intensive process, so configurations that allow these items to be scaled are important. In addition, since the text extraction process for the indexing process occurs on the Content Platform Engine, using CSS places an increased load on the Content Platform Engine and therefore requires additional scaling of the Content Platform Engine.

For optimal performance, use separate, dedicated index and search servers.

3.3 Virtualization

Virtualization has become a major trend in the IT industry. The drivers for virtualization are cost reduction and providing better management of hardware resources. Virtualization can be applied over servers, storage, and applications. In this section, we focus on server virtualization.

In general, multiple servers are consolidated into fewer servers and operate inside of their own environment. An abstraction layer between the physical resources and the running application is created. Physical resources are encapsulated as logical resources, and the environment for the application is moved into a virtual machine (VM). The shared resources usually are CPUs, memory, network bandwidth, and hard disk storage.

The benefit of virtualization is better use of the current hardware, because the number of physical boxes decreases, and a physical box becomes a virtual machine. Instead of managing multiple systems, the resource optimization can be concentrated at one point. It also opens new pathways for high availability and disaster recovery, because you can copy entire systems to another location.

Depending on the virtualization technology that you use, the system administrator can assign each virtual machine an individual amount of resources, such as memory or a fraction of CPU resources at run time. This increases system agility and ensures scaling on demand. An administrator can react dynamically to changes in system utilization.

For example, if at the end of the month, usage of a certain virtualized application increases sharply, it can be scaled on demand and assigned more system resources. In that way, the system hardware is used more efficiently.

Another example is systems that are usually idle and have predictable peak times. Given the fact that the peak times occur at different points in time, you can benefit by moving applications from these systems onto one virtualized server.

A third example is systems that are used for training and support. Because virtualization technology provides the option to clone an existing system, you can clone a training system with preloaded data from another system. In the area of client support environments with different operating systems, application version and patch levels can be stored and started on demand. That increases flexibility and speeds up problem deduction, because no time-consuming installation tasks are necessary.

Virtualization approaches differ in the degree of abstraction. In this book, we only provide an overview. For more information about virtualization, see the information provided by the virtualization solution providers.

Virtual machines using virtual machine monitors

Virtual machines (VMs) run on top of a guest operating system. The virtual machine is not aware that it is not running on real hardware. Physical resources, such as a network card, are emulated.

When the VM wants to access resources that are managed in a system context, the access is performed by a virtual machine monitor (VMM). The VMM analyzes the code and provides a replacement function that safely accesses the resources. Figure 3-8 illustrates virtual machines using VMM.

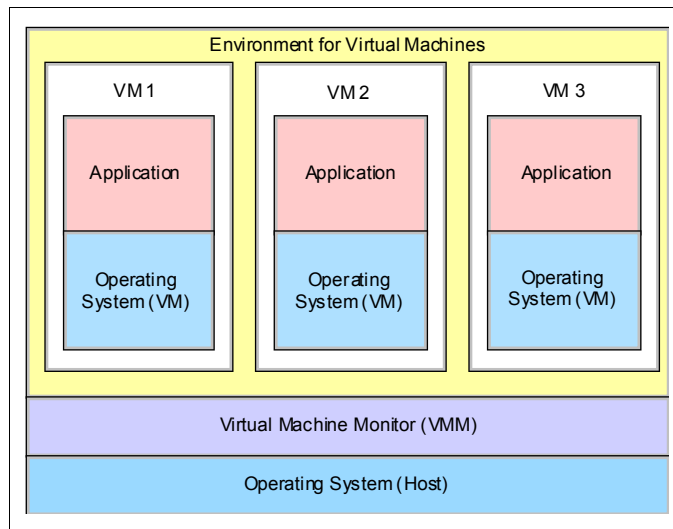


Figure 3-8 Virtual machines using VMM

In certain implementations, the host operating system and VMM are combined into a single layer. Examples of this approach are VMware products or Microsoft Virtual Server.

Virtualization on the operating system level

In this architecture, virtualization is done on the host operating system level. The solution uses a single kernel. Figure 3-9 on page 60 shows the architecture.

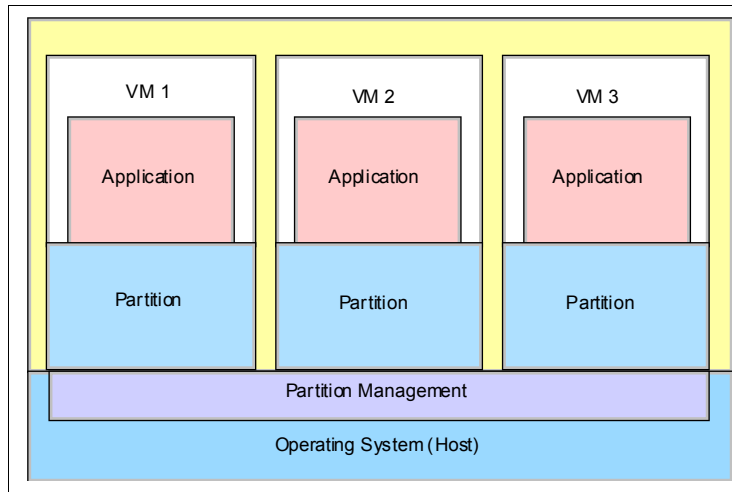


Figure 3-9 Virtualization on operating system level

In this scenario, the coupling between the host operating system and the VM is much tighter. Because only one kernel is used, the overhead incurred with this approach is small. However, the disadvantage of virtualization at the operating system level is that it does not allow you to run different operating systems.

The isolation of the single partition is key, because the system operates in one kernel. This is done in the partition management part of the operating system. The resource management, which is where the physical resources such as CPUs and memory are assigned, is also done in the partition management part of the operating system.

This level of virtualization is popular for service providers who offer Internet services or host special services. For this scenario, the low overhead and the automation for replicating and horizontal scaling of virtual servers are key.

3.3.1 A virtualized IBM FileNet Content Manager system

IBM FileNet Content Manager installations can be implemented using a variety of virtual technologies, including IBM logical partitions (LPARs), IBM workload partitions (WPARs), and VMware ESX. For more information about the supported virtual technologies, see the IBM FileNet P8 Hardware and Software Requirements guide.

Tip: When performing capacity planning, it is important to identify whether you will be using virtual technologies as this affects the required system resources.

Figure 3-10 provides one possibility for deploying an IBM FileNet Content Manager system in a virtualized environment. Multiple virtual machines are involved. Each component is deployed in its own virtual machine, in a separate partition. The figure includes the database server and the directory server, although typically these components are virtualized only in demo or sandbox environments.

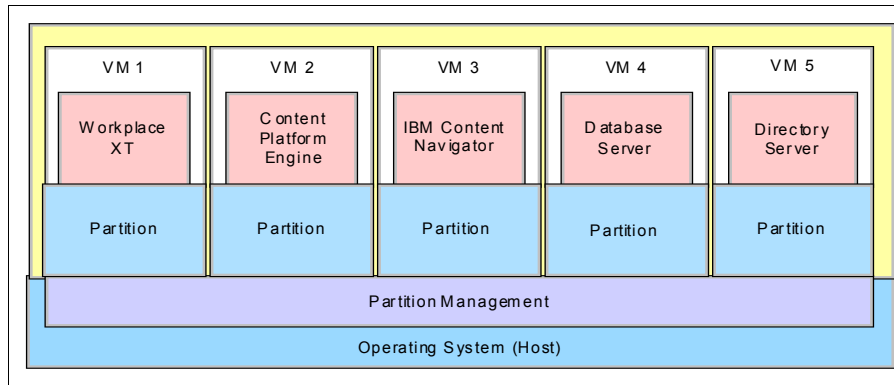


Figure 3-10 Deploying each engine in its own virtual machine

This architecture offers the highest flexibility and scalability because of the number of virtual machines that you can have in the configuration.

Collocating engines in a single virtual machine

The opposite approach to multiple single virtual machines is to collocate everything in one virtual machine. See Figure 3-11.

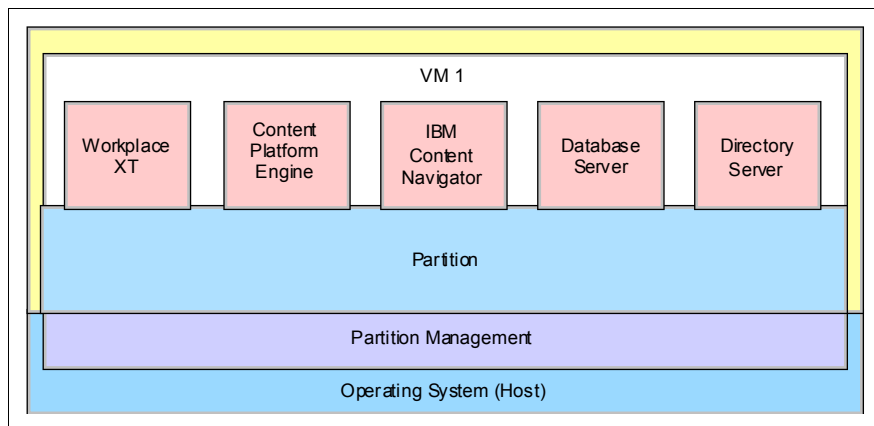


Figure 3-11 Collocating engines in one virtual machine

This approach reduces the complexity; however, scalability is limited. Collocating multiple components in a single virtual machine is suitable for sandbox environments, demo systems, and small development systems.

System duplication

Using virtual technologies can make it easier to duplicate systems, but consider uniqueness requirements when reusing images:

- ▶ Host names must be unique and some applications, including WebSphere, are sensitive to host name changes.
- ▶ If using a multitiered application solution in a WebSphere Application Server Network Deployment environment, ensure that the cell names are unique at each layer in the tier. See the following article for more details on this issue:
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fae%2Fuagt_rcell.html
- ▶ When using static IP addresses, ensure that duplicate images are updated with new values and that any load balancers are updated to include the new addresses.

3.4 Shared infrastructure

At times, you might want to set up a shared infrastructure. For example, clients roll out IBM FileNet Content Manager as an enterprise-wide solution and want to manage multiple projects on the same system by sharing the resources and infrastructure of the IBM FileNet Content Manager system among the projects.

Another use case is an internal or external application service provider. The application service providers can share the infrastructure with several independent customers or tenants.

In this section, we discuss options for clients using a shared infrastructure model and provide best practices with regard to the requirements.

3.4.1 Communication between the engines

To simplify the description, we use IBM Content Navigator, an out-of-the-box application that comes with IBM FileNet Content Manager, to explain how an application interacts with the Content Platform Engine, object stores, and isolated regions.

How IBM Content Navigator locates an object store

IBM Content Navigator is deployed to an application server. Connections to repositories are defined using the IBM Content Navigator user interface. In addition to being used with IBM FileNet Content Manager, IBM Content Navigator can access Content Manager OnDemand and Content Manager repositories. IBM Content Navigator can also be extended to access other types of repositories if they have a Content Management Interoperability Service (CMIS) interface.

Figure 3-12 on page 64 illustrates configuring IBM Content Navigator access to a specific object store using a specific Content Platform Engine server. In this instance, the supplied URL uses the Internet Inter-ORB Protocol (IIOP) port on a stand-alone WebSphere server. To use a Content Platform Engine deployed on a WebSphere Application Server Network Deployment cluster, the URL has the following form:

```
corbaloc::node1_hostname:BOOTSTRAP_ADDRESS,:node2_hostname:BOOTSTRAP_ADDRESS/cell/clusters/your_websphere_cluster_name/FileNet/Engine
```

Note: Content Platform Engine can also be accessed via the Web Services transport or Content Engine Web Services (CEWS). Load balancing in this type of configuration is provided by a load balancer and the URL is in the following form:

```
http://virtual server name:virtual port number/wsi/FNCEWS40MTOM/
```

Currently, IBM Content Navigator does not support accessing Content Platform Engine using the CEWS transport.

After you provide the server URL, and the object store symbolic and display names, click **Connect** to establish a connection to the Content Platform Engine server. After a connection is established, the other tabs (Configuration Parameters, System Properties, and Browse) are enabled.

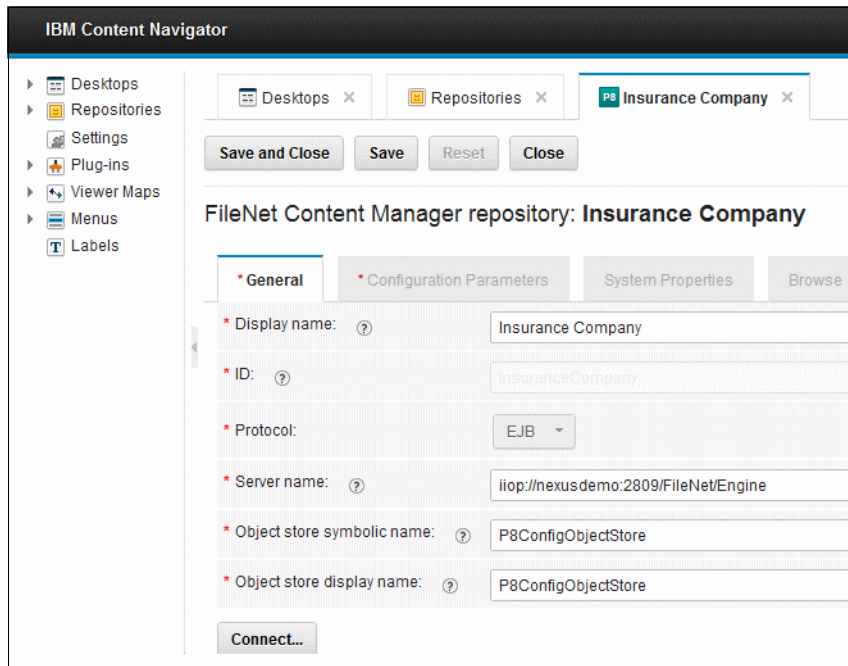


Figure 3-12 Configuring access to an object store using IBM Content Navigator

How IBM Content Navigator locates an isolated region

Each workflow system contains one or more isolated regions. An isolated region is a logical subdivision of the workflow system's database that contains queue, process, and status information. A *connection point* is used to identify a specific isolated region in a workflow system. Connection points are defined using the Content Platform Engine administration tools.

To identify the isolated region to be used with a particular repository, click the **Configuration Parameters** tab in the Repository configuration UI as shown in Figure 3-13 on page 65. Then, select a connection point from the Workflow connection point drop-down list box.

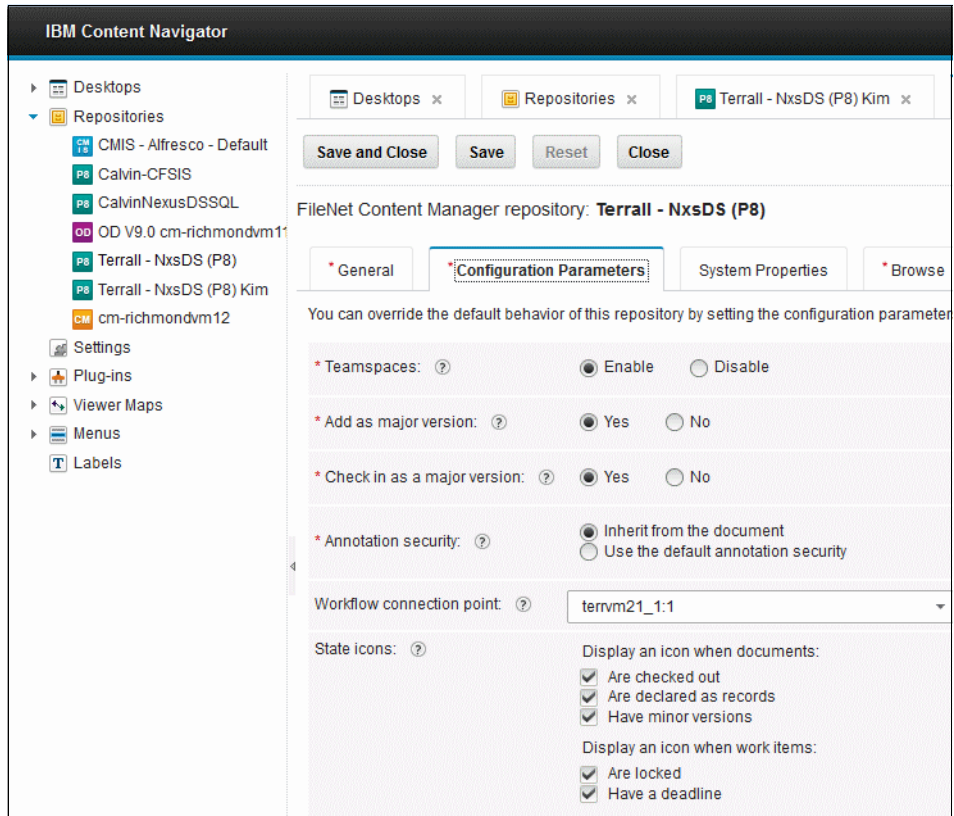


Figure 3-13 Defining the connection point in IBM Content Navigator

3.4.2 Data segregation

In the following section, we discuss the options for data segregation.

Note: There are multiple approaches to setting up a system, and you must select the methodology that best fits your business requirements.

The level of data segregation does not affect the ability to scale the system; instead, choose the level of segregation based on these factors:

- ▶ Application needs
 - Who needs access to the application and the associated content?
 - How often do application updates need to be rolled out?

- Are there conflicting performance requirements? For example, indexes needed to make one application perform well might have a negative impact on a different application.
- ▶ Client locations

You can improve performance by locating the databases used by specific object stores and workflow systems local to the clients.
- ▶ Legal requirements

When servicing the needs of global customers, be aware of legal requirements for business content to be kept in the country of origin.
- ▶ Volume of content

Even when building a single application, there might still be a need to plan for multiple object stores because of the volume of content being generated. Although there are no specific limits on the number of objects that can be kept in an object store, practical limits are driven by these issues:

 - Time required to back up and restore the object store database
 - Indexes required to optimize search performance
 - Required response times for ingestion and by clients performing day-to-day tasks
- ▶ Security requirements

The IBM FileNet Content Manager security model is flexible, but if, for example, there is a need for different administrators based on the content or application requirements, these needs might be best addressed by using separate object stores, workflow systems, and isolated regions.

For more information about the IBM FileNet Content Manager security model, refer to Chapter 5, “Security” on page 151.

3.4.3 Levels of data segregation

Data can be segregated in multiple levels.

Shared object store and isolated region

The form of data segregation with the least physical segregation is for all applications to use the same object store and isolated region. In this type of configuration, the way in which security is applied to the object store and isolated region maintains the required segregation between users and projects.

Separate object stores but shared workflow system

In this configuration, the business content is separated into different object stores, but all business processes are managed in the same workflow system.

This configuration is useful in the following situations:

- ▶ The workflow load is comparatively light compared to the volume of content being used.
- ▶ There is a need to physically locate content in different geographic regions for legal reasons or for better performance.

For example, if you are building solutions for departments that are located in different geographies, storing the content so that it is local to the departments can improve response times.

In this configuration, the workflow system is collocated with the object store that has the content that is used most frequently with a business process.

Separate object stores each with a workflow system

In this configuration, business processes are tightly integrated with the business content. Each object store is secured in a way that ensures that only the project users are allowed to access it.

Each project looks only at its own content and project data, but since the same hardware is used for all projects, the number of maintenance updates that need to be made to the infrastructure is still limited.

Separate P8 domains

To maintain complete independence between applications, use separate P8 domains. Each P8 domain has its own set of hardware (that can be virtualized), object stores, and workflow systems.

In deployments with a large amount of content, it is often necessary to “roll” to new object stores regularly, and eventually, it might also be necessary to “roll” to a new P8 domain. For instance, after 50 object stores have been created in a P8 domain, a new P8 domain can be created to house the next set of 50 object stores. When the new P8 domain is configured, consider moving the most “current” of the old object stores to the new P8 domain using the FileNet Deployment Manager. If the object stores are connected with active workflows, the associated workflow systems must also be moved.

A separate P8 domain can also be used to give a global view of corporate data for reporting purposes. In this type of scenario, Content Federation Services for IBM Content Integrator is used to federate content from various source P8 domains to a parent P8 domain.

LDAP servers, database servers, and storage devices

The use of common or disparate LDAP servers, database servers, and storage devices is also driven by environmental requirements. Typically, the corporate LDAP server is used for all applications. The number and location of database servers and storage devices are driven by capacity, performance, legal, and maintenance requirements.

3.4.4 Degree of sharing

When deciding on a suitable architecture, review the detailed requirements. A good starting point is to examine the security requirements. Does the system need to manage multiple projects using a common security base? If so, we suggest one domain. If the requirements mandate completely separated security, we suggest multiple domains.

Can the different projects share content with each other? If so, we suggest one domain. If not, consider using multiple domains.

If two projects that use different security structures must share data, use federation to make content visible to both projects. Alternatively, use one of these approaches:

- ▶ Put all users in a common directory service and secure the content via an access control list (ACL).
- ▶ Use federated repositories to create a single realm. For more information about using federated repositories in WebSphere, see the following link:

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=%2Fcom.ibm.websphere.zseries.doc%2Finfo%2Fzseries%2Fae%2Fcwim_fedrepos.html

Another factor that can affect the degree of sharing is maximizing system availability. This aspect of the data sharing design is influenced by the following factors:

- ▶ Location of application users
- ▶ Maintenance windows for applying infrastructure and custom application updates
- ▶ Service-level agreements (SLAs)

We advise that you use the Data Source sharing feature. For more information, see the “Sharing Data Sources” and “Creating a Database Connection” topics in Administering Content Platform Engine:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.admin.tasks.doc%2Fp8pcb027.htm>

3.4.5 Cloud deployments

IBM FileNet Content Manager can be deployed into a cloud environment as long as the cloud can meet your requirements for data segregation, location, and security.

The obvious advantage of a cloud deployment is the ability to access data without the cost of maintaining the servers or software. IBM provides a hosted archive cloud in Italy that uses IBM FileNet Content Manager. The Information Archive enables clients to archive content in a private cloud hosted in an IBM data center.

3.5 Geographically distributed systems

In previous sections, we discussed scalability, virtualization, and data segregation. IBM FileNet Content Manager contains a number of capabilities to extend the system geographically and use it as a distributed system with different locations.

Although centralized data centers can be simpler to administer and maintain, they do not necessarily perform these functions:

- ▶ Provide the responsiveness needed by the application users
- ▶ Meet the legal requirements for locating data in the country of origin

So, it is important to take these considerations, along with the required level of data segregation, into account when designing an IBM FileNet Content Manager system.

Tip: Geographically distributed environments are also built to support business continuity requirements for high availability and disaster recovery. Configuring IBM FileNet Content Manager to meet these types of requirements is covered in Chapter 7, “Business continuity” on page 217.

3.5.1 Site, virtual server, and server configuration

To address the needs for a distributed system, each P8 domain can be structured to have the hardware hosting the domain in multiple physical locations and configure the logical components to use hardware in specific locations.

A *P8 domain* is the environment in which all Content Platform Engine servers operate. Domain information is stored in the global configuration database (GCD). The GCD holds all topological information of the domain, such as servers

and assigned resources. It contains the descriptive and location information of the subcomponents.

When a Content Platform Engine server is brought online, its first task is to interrogate the GCD to get information about the P8 domain and to store a copy of the GCD information in the working directory of the application server.

If the GCD cannot be contacted, the Content Platform Engine server retrieves the domain information from the cached version of the GCD.

When the GCD cannot be contacted, IBM FileNet Content Manager applications still function. Clients can view, update, and add content to object stores, as well as participate in workflows, but tasks that cause an update to the GCD, such as creating new object stores or marking sets, fail.

Historical note: In earlier IBM FileNet Content Manager releases, the Content Engine servers cached a local copy of the GCD in memory so that issues with GCD connectivity did not cause the system to fail. However, if the GCD was offline, the Content Engine servers were unable to start.

Figure 3-14 on page 71 shows an IBM FileNet Content Manager system distributed over two locations at a domain level. The system is distributed over two locations: the main location and a satellite location.

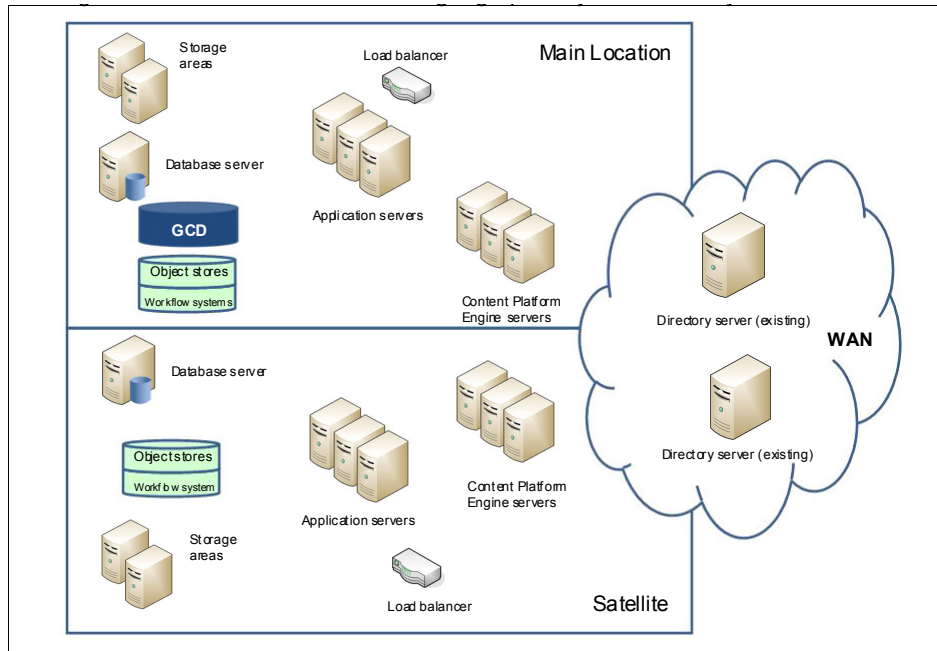


Figure 3-14 Domain level view of a geographically distributed system

A *site* represents a geographical location. All site resources are well-connected via fast, reliable LAN. There is no functional limit to the number of sites that a single IBM FileNet P8 domain can contain.

A *virtual server* is the logical service point with which Content Platform Engine clients interact. A virtual server can map to a single independent server instance or to a set of server instances. When a virtual server contains multiple server instances, client requests are load-balanced across the set of server instances through the Java EE application server's clustering capabilities or through the use of a load balancer that provides scalability and high availability. In either case, applications accessing the virtual server are unaware of the number or type of server instances that reside behind it. There is no functional limit to the number of virtual servers that a single P8 domain can contain. A virtual server can also be configured as an active/passive cluster, although we do not recommend this approach.

A *server instance* is an individual Java EE application server instance. Multiple server instances (each running in their own JVM) can be hosted on a single physical server. Content Platform Engine clients do not interact directly with a server instance. Logically, clients always go through a virtual server. There is no functional limit to the number of server instances that a single P8 domain can contain.

Figure 3-15 illustrates a hierarchical view of the domain, sites, virtual server, and servers as displayed in ACCE.

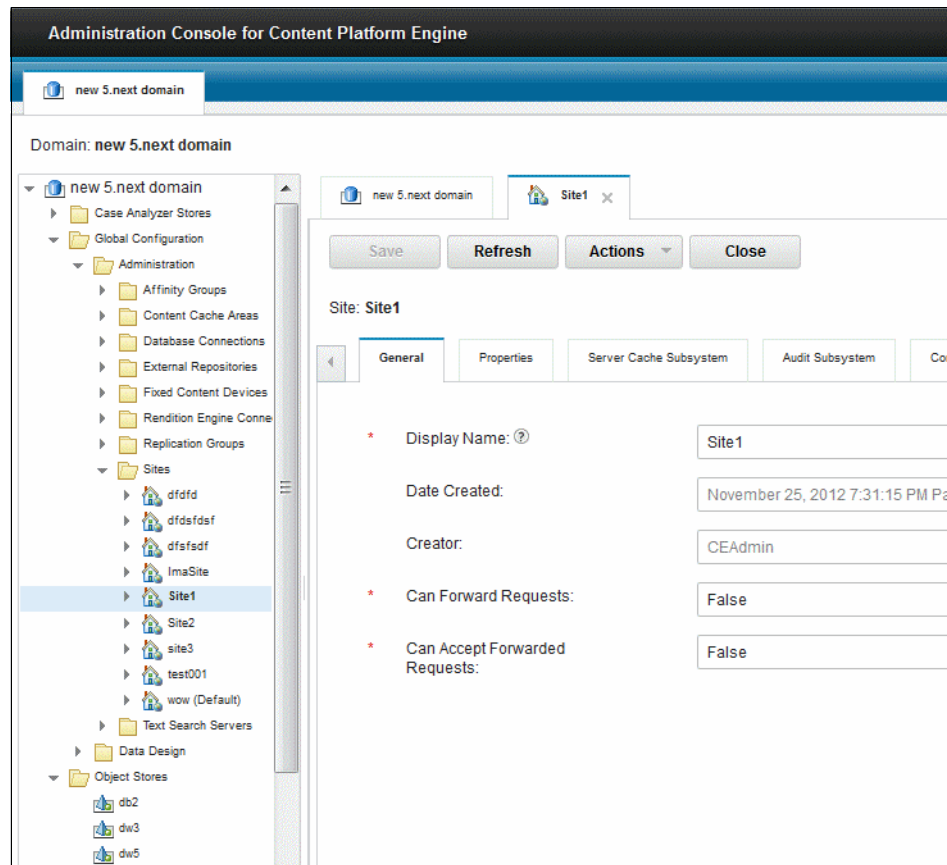


Figure 3-15 Hierarchical view of domain, sites, virtual servers, and servers

This hierarchy simplifies administration, because attributes are inherited from a parent component to its children. It minimizes duplicate configuration. One example is trace logging. If you want to analyze the entire system, you can activate trace logging at the domain level. If you want to activate trace logging only at a special site, virtual server, or server, you can configure it at the appropriate level.

To summarize, dividing a system into hierarchical components is useful for creating a distributed system and simplifies the administration due to the inheritance feature. For more information about domain, sites, virtual server, and server instances, refer to 4.5, “Repository organizational objects” on page 95.

3.5.2 Distributed content caching model

In this section, we discuss the caching mechanism and show the architecture of a geographically distributed system.

Caching is a building block for distributed systems. IBM FileNet Content Manager includes caching at the Content Platform Engine level. It is deeply integrated into the system. The benefits of caching are that it speeds up retrieval and it can be used by any client regardless of who authored the software. Caching addresses content objects and can be used for all types of storage. A document can reside in multiple caches. You can place each cache on the Content Platform Engine server or a network share. Cache servers can be installed at sites where you do not need to perform a full Content Platform Engine installation.

The Content Cache configuration can be customized in the following ways:

- ▶ **Threshold size**
Defines how much space the cache can use before content is removed from the cache.
- ▶ **Threshold elements**
Defines the number of elements that can be added to the cache before content is removed from the cache.
- ▶ **Amount to prune**
Defines the percentage of content that needs to be removed when a threshold is reached.
- ▶ **Preload content when created**
Loads content into the local cache as the content is added to an object store. This feature is especially useful when content is typically used at the same site that the ingestion occurs but the database associated with the object store is at a different location.

If the content is going to be frequently accessed from a site that is different than the one at which the content was ingested, consider developing a custom application to access the content during off-peak hours to load the content into the cache at the remote site.
- ▶ **Content lifespan**
Identifies how long content stays in the cache without being accessed.

When configuring the content cache, you need to be aware of the following information:

- ▶ The document content access patterns at each site. For example, will documents need to be viewed within 24 hours of being ingested? Or, are the documents being ingested primarily for archive purposes and are only accessed by clients sporadically?
- ▶ Who will be downloading or viewing the document content and where are they located?

If the content is being accessed by clients who are geographically close to the database and storage areas used to store the document information, configuring a specific content cache for those users might not be necessary.

- ▶ Any legal requirements regarding the location of content

For more information about using content cache areas, see the following topic in the P8 Information Center:

<http://publib.boulder.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.admin.tasks.doc%2Fp8pcc101.htm>

3.5.3 Request forwarding

When talking about distributed systems, the efficient use of the network bandwidth between the locations is essential.

Request forwarding is used to send requests from a “remote” Content Platform Engine server to a “local” server. In this case, “remote” implies a server that is not local to the object store database that must be accessed to complete the request.

A content cache improves the speed at which clients can view and download documents. However, the request forwarding improves the speed at which processing, for example, a search or an event action, executes because the processing is “pushed” to the Content Platform Engine server that is local to the needed resources. In addition, request forwarding reduces WAN traffic, which also helps overall system performance.

See the P8 Information Center for additional information about request forwarding:

http://publib.boulder.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.admin.tasks.doc%2Faboutem%2Frf_concepts.htm

Figure 3-16 on page 75 shows a system distributed over two locations, a main location and a satellite location. Request forwarding is disabled, which is the default setting.

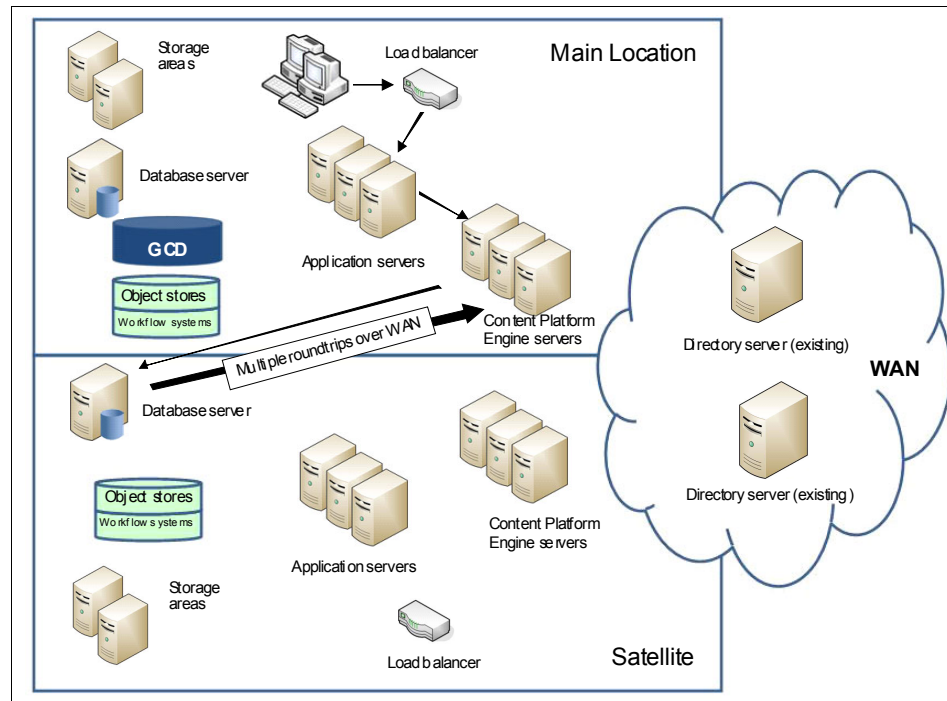


Figure 3-16 Retrieval without request forwarding

If a client on the main location (*main*) initiates a search request on content residing at the satellite location (*sat*), the communication goes from IBM Content Navigator (main) to Content Platform Engine (main). Content Platform Engine (main) then contacts the database (sat), and the database data is transferred over the network. Finally, Content Platform Engine (main) communicates the result list back to IBM Content Navigator (main) that presents it to the client.

When Content Platform Engine (main) talks to the database (sat) and searches for metadata, this can require a number of queries, and therefore network round-trips occur to complete the request. If the WAN link between the sites has high latency, delayed response times are the consequence.

Figure 3-17 on page 76 shows the mechanism for IBM FileNet Content Manager with request forwarding enabled.

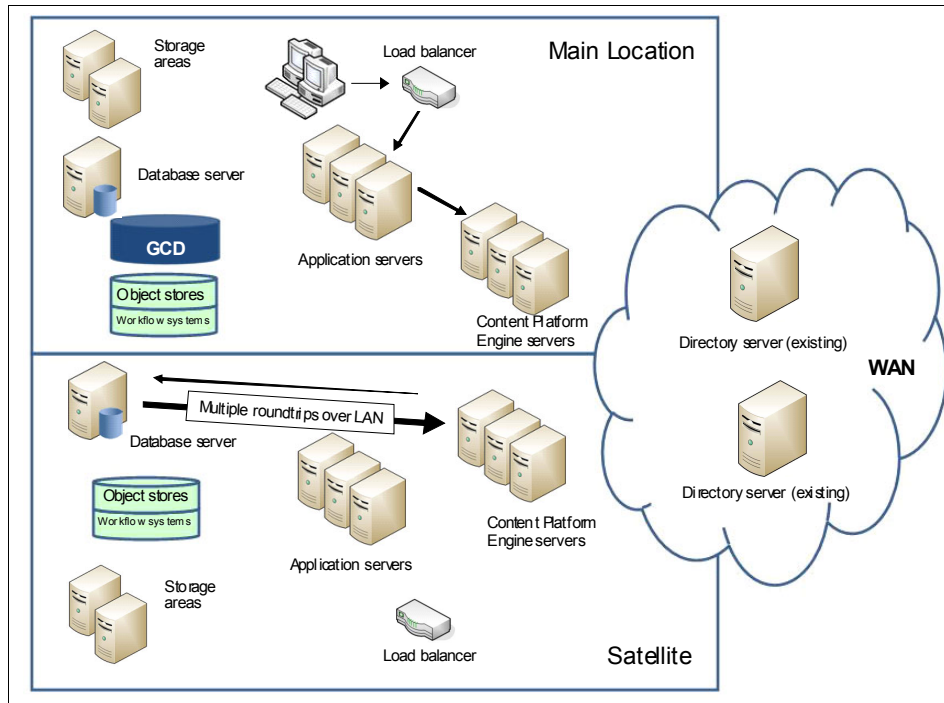


Figure 3-17 Retrieval with request forwarding

When enabling request forwarding, you declare that each defined object store has affinity with a specific site. There are two settings associated with request forwarding: the ability to forward requests and the ability to receive forwarded requests.

Again, the client (main) addresses IBM Content Navigator (main), which contacts Content Platform Engine (main). Instead of directly contacting the database (sat), Content Platform Engine (main) forwards the request to Content Platform Engine (sat), which contacts the database (sat). Content Platform Engine (sat) gathers all data and returns it to Content Platform Engine (main). Again, Content Platform Engine (main) passes the result back to IBM Content Navigator (main) where it is presented to the client.

In general, when request forwarding is configured, client requests to other sites are forwarded to one or more virtual servers at the site associated with the object store. This has the advantage of minimizing the impact of high network latency because the cost-intensive database access is performed locally via the LAN instead of through the WAN.

At the time that a Content Platform Engine server receives a request, it evaluates the request to decide whether to forward it or not. For metadata requests, if all actions in the client request are based on an object store at a different site, Content Platform Engine will attempt to forward it. At the destination site, the administrator enables one or more virtual servers to be able to receive the incoming requests.

Request forwarding is across the Enterprise JavaBeans (EJB) transport layer only and is only supported across homogeneous application servers.

3.5.4 Distributed workflow systems

In IBM FileNet Content Manager 5.1 and earlier releases, workflow processing was handled by a separate engine called the Process Engine and the best practice was to use a centrally located Process Engine. Now that workflow systems are collocated with an object store, as you design your environment, consider the following questions when deciding how to pair up workflow systems with object stores:

- ▶ Who will be participating in workflow processing and where are they located?
- ▶ Is there document activity as a result of the workflow process and are the documents primarily in a single object store?

Document activity includes using documents as attachments to the workflow or utilizing capabilities, such as the Content Extended operations, as part of the workflow.

3.5.5 Use cases for distributed systems

Let us discuss several use cases and the corresponding architecture. Assume that we have two locations: main location and satellite location.

The main location contains a full IBM FileNet Content Manager system (FileNet Workplace XT, IBM Content Navigator, Content Platform Engine, an object store with a workflow system, file store, database, and Directory Service). You can set up the following options at the satellite location:

- ▶ No IBM FileNet Content Manager components are deployed at the satellite location. Only third-party solutions are deployed.

The easiest way to enable the users at the satellite location to use the system is to provide them with the URL of the IBM Content Navigator application (or a custom application) at the main location. You can choose this approach if the satellite location has a similar infrastructure as the main location with high bandwidth and low latency.

An alternate approach is the use of third-party software, such as Microsoft Terminal Server or Citrix, in which the application runs at the main location and only the content displayed in the window is transferred to the remote location. This is a solution for clients who have already deployed this type of technology or for clients who have legal requirements regarding the location of the document content.

- ▶ Install IBM Content Navigator only.

Establishing an additional IBM Content Navigator installation at the satellite is a solution for WAN networks, where it is better to have the WAN cloud between IBM Content Navigator and the Content Platform Engine instead of between IBM Content Navigator and the clients. Although the footprint of this solution is small in relation to performance, much better results can be achieved when using caching.

- ▶ Install IBM Content Navigator and Content Platform Engine (not recommended).

We do not recommend this setup, because the Content Platform Engine needs to be local to the database for optimal performance.

- ▶ Install IBM Content Navigator and Content Platform Engine with a content cache area.

This is the classical scenario for a centralized system, where the satellite users are primarily downloading and viewing content, but they do not perform other types of work on the system. Preloading of content completes the solution, if appropriate.

- ▶ Install IBM Content Navigator and Content Platform Engine with a content cache area and request forwarding enabled.

This is the classical scenario for a centralized system, where the satellite uses caching for content retrieval and additional work needs to be done at the satellite in addition to downloading and viewing content.

- ▶ Install IBM Content Navigator, Content Platform Engine, and file store (not recommended).

We do not recommend this scenario, because the file store and object store databases need to both be local to the Content Platform Engine for optimal performance.

- ▶ Install IBM Content Navigator, Content Platform Engine, the object store database, and the file store.

In this scenario, the P8 domain has more than one object store. Each satellite location has a local object store but can still access data stored in other object stores.

This architecture is useful:

- If users at the satellite location primarily access content only in the local object store.
- An independent satellite must store its own data.

If retrieval from the satellite location to the main location is required, add a content cache area to the main location.

In all the listed configurations, since FileNet Workplace XT is being used primarily as an administrative tool, deploy it at one site only.

3.6 Conclusion

In this chapter, we described IBM FileNet P8 Content Manager architecture. In the next chapter, we provide best practices and recommendations when designing object stores.



Repository design

In this chapter, we introduce the basic concepts and elements that comprise a repository. Repositories encapsulate not only the content being managed but also the various metadata elements and infrastructure that support the IBM FileNet Content Manager functionality. In addition, we describe the repository design elements and guidelines for using these elements.

We discuss the following topics:

- ▶ Repository design goals
- ▶ Object-oriented design
- ▶ Repository naming standards
- ▶ Populating a repository
- ▶ Repository organizational objects
- ▶ Global configuration database (GCD)
- ▶ Repository design objects
- ▶ Repository content objects
- ▶ Storage media
- ▶ Considerations for multiple object stores
- ▶ Retention management and automatic disposal
- ▶ P8 Content Manager searches

4.1 Repository design goals

Repositories are the central components of IBM FileNet Content Manager solutions. In this chapter, when we talk about repositories, we do not mean just the low-level databases, file systems, and other technical components where pieces of data reside. We mean a more encompassing definition that includes not only those things, but higher level constructs, such as access control, relationships among objects, different types of business objects, and various ways that users and applications interact with them. Some solutions might have repositories composed of a single object store. Other solutions are composed of dozens of object stores, file systems, storage devices, and other resources.

Repositories store content, such as office documents, images, records, and other types of electronic content along with associated metadata. FileNet Content Manager repositories are capable of storing billions of documents and records, providing a centrally accessible, enterprise-wide library of information that can be centrally managed and used in many different ways.

The decomposition of FileNet Content Manager solutions into the various repository elements is designed to facilitate not only the separation of logical and functional purposes, but it also is designed to meet a number of additional goals. The architectural framework offers features that facilitate the specific goals of scalability, maintainability, securability, well-behaved enterprise citizenship, and flexibility for future function and growth. Each section in this chapter explains specific features of these elements.

Security features are present at almost every level and manifestation of the repository elements and, in many instances, in multiple ways. These features provide various security granularities from broad to specific and individualized levels. See Chapter 5, “Security” on page 151 for additional details of security features.

4.2 Object-oriented design

IBM FileNet Content Manager follows an object-oriented design (OOD) paradigm. Every element represented in the system exists as an object. For example, an element can be a content object that contains the metadata and reference for a specific document, or it can be the definition of a document class that defines what the metadata objects look like.

The following design aspects are complementary high-level approaches to repository design:

- ▶ *Content decomposition.* Examine the types of content across the organization, the purpose of the content, and then group the content based on shared properties and metadata. For example, policy documents can be grouped together, separate from checks, statements, and claims. The perspective of this approach focuses on the content and types of content contained in specific documents.
- ▶ *Grouping content by relationship.* Examine specific relationships between documents, such as checks related to a specific claim, which in turn is related to a specific policy. The perspective of this approach focuses on formalized relationships between documents.
- ▶ *How content is used and accessed.* This analysis can reveal content relationships that are not formally captured in metadata. For example, a spreadsheet listing appraisers in a specific geography is usually accessed along with claims. Customer service representatives typically look at all documents relating to a specific user or geography. The perspective of this approach focuses on how people access and use content to complete their tasks.
- ▶ *Business processes associated with the content.* The business processes that use the content, the document lifecycles, and the workflows all give a perspective that is based on the functionality of the documents. The perspective of this approach allows the grouping of content based on what it does. Content can also be combined with other content to create new content. For example, a report combines data from various sources and is generated by a report generation process.

4.2.1 Design approaches

Two basic directions from which to approach repository design are bottom-up and top-down. Both approaches offer specific benefits and advantages, and each approach carries with it certain limitations that can make it unusable in a specific situation. Because solution design is an iterative process, and because it can include a reasonably large scope, it is not uncommon for both approaches to be integrated and applied to different areas of the design as appropriate. It is better to employ both design approaches to the solution, focusing on both of their strengths, and reconciling the approach perspectives as they meet in the middle.

Regardless of which design approach is used, in what combinations, or with other design approaches, the ultimate design goals remain the same. There must always exist a specific set of clear business requirements that is driving the solution.

Bottom-up design approach

Approaching the design from the bottom up has the advantage of enabling the use of existing content, organization, business knowledge, and expertise that are either explicitly or implicitly present within the organization. Involving business users and subject matter experts (SMEs) greatly enhances the utility and usability of the resultant design.

Designing the repository from the bottom up means analyzing the existing content and processes in use by the organization and synthesizing the abstract entities from this information. Repeated applications of grouping the resultant entities based on a specific set of characteristics and then synthesizing the next layer up by abstracting these groupings yields the resultant design. Each level of organization of entities allows a different facet of design detail characteristics to be focused on and separated out from the others. During the requirements gathering period, collect the inputs from all groups that will be involved in using, building, testing, training, or operating the system.

Recommendations: Interview the business users and SMEs and record their comments as a set of initial requirements for the system. Use a proof-of-concept (POC) to validate the requirements and design in the early phases of the project.

The bottom-up approach has the advantage of allowing you to work with existing, well-understood content and with workers who have expert knowledge about that content. Often, as the design grows from the bottom up, it becomes more difficult, especially for the knowledge workers, to abstract further away from the concrete details with which they are used to working.

The bottom-up approach has the disadvantage of taking all of the implicit knowledge about how the existing problem space is approached, including any artificial constructs that were used for historic or other reasons. Some of those might be contrary to a good design. It is frequently difficult to get past those legacy design decisions in order to understand the true underlying requirements.

Top-down design approach

Approaching the design from the top down has the advantage of allowing the design to be formalized from a clean start. Any existing faults in the system, historical processes, procedures that no longer add business value, and any preconceived notions of what is expected can be avoided. This allows the enterprise viewpoint to be fully exercised and elevates the considerations for areas, such as future flexibility, growth, and overall integration structure, to be fully considered.

This process typically starts by utilizing not SMEs, but solution domain experts. They understand the technology and architecture of enterprise content management systems. The top-down approach works its way down to the level where SMEs must be consulted for the final and essential details.

Designing from the top down involves understanding the global picture and decomposing the various levels of the design through clear design goals or specific design choices. It is also an iterative process, driving from the most abstract toward the most concrete levels. By designing from the top down, the specific order of design characteristics can be approached in the manner that makes the most strategic sense for the organization.

The top-down approach has the advantage of developing a design that does not include any artificial non-technical barriers, for example, historical organizational structures. It produces a design that emphasizes the strategic requirements of the solution. This often results in the most flexible and adaptable design for the future.

The disadvantage of the top-down approach is the difficulty in mapping existing content and processes into the new design that is developed. As the design iterations approach the more concrete aspects and need to be mapped directly to concrete business entities, the process can become conceptually and politically difficult for knowledge workers, depending on historic organizations.

Recommendations: Even if you start with a top-down approach, keep SMEs informed and involved at an appropriate level. Let them know that they will be vitally involved as you get to the more concrete layers. An information vacuum can create genuine misunderstandings that can make everyone's job more difficult.

4.2.2 Design processes

Producing the best possible design requires coordination and cooperation from all of the major areas that the solution touches. In addition, all of the major areas that will be directly affected by the solution need to be involved and committed to the goals. However, that is not always possible to achieve, so designing as close to a perfect solution as possible is the next best goal. There are a number of design processes and concepts that have been shown to be extremely useful in producing an effective repository design.

The two key elements necessary are the team that undertakes the design and the specialized pieces of information that are needed to make the correct design decisions.

The design team itself can consist of one or more architects with the specific responsibility of producing the design. Regardless of the number of individuals in the design team, there is a clear set of roles and responsibilities that must be represented. These roles cover both the technical facets of the design as well as the business facets. The team is usually led by a technical architect who has the direct responsibility for the overall solution. The team is either populated with architects and representatives from the following areas or it includes contacts in the following areas who can provide feedback and direction as needed to the team without being full-time team members:

- ▶ P8 Content Manager architect technical role

This is the architect who has the ultimate responsibility for the overall repository and solution design. This role must be filled by a full-time member of the design staff who has expert knowledge of the P8 Content Manager product.

- ▶ Enterprise architect technical role

This is the architect who is responsible for overseeing the technical fit of the solution into the existing solution portfolio. This role must be filled by someone who has expert understanding of the current technology across the enterprise.

- ▶ Application architect technical role

This is the architect who has direct responsibility for the specific application or applications being addressed at this phase of the design and who is responsible for tracking the business requirements into the solution space.

- ▶ Enterprise security technical role

This is someone who has expert understanding of the security environments and models that are used in the enterprise infrastructure. The purpose of this role is to assure that all existing security policies are followed and to provide support as needed for security requirements outside of the P8 Content Manager solution itself.

- ▶ Legal business role

This role must be filled by someone who has expert knowledge of the legal requirements of the business sphere in which the solution exists. They provide guidance about requirements and restrictions on the system that are imposed for legal, as opposed to business value, reasons.

- ▶ Knowledge worker business roles

These roles represent the directly affected business workers whose content and processes are being integrated into P8 Content Manager and who have the inherent and implicit knowledge of the business that is not usually captured in any other manner.

4.3 Repository naming standards

Prior to designing the repository, initial thought must be given to the conventions that will be used to standardize naming across all of the types of objects that will be in the repository. At this stage, concern yourself with the standardization across the organization and design elements of the repository as opposed to the content objects themselves that will be placed into the repository by users. Through a well-thought-out naming scheme, you can avoid many potential problem areas at the beginning of the project as opposed to discovering them during the lifetime of the repository. All objects that are created as organization and design objects need to be named as descriptively as possible. When there are hierarchal relationships between objects, it makes sense to capture this relationship in the naming standard as well. For example, Company XYZ has a site that is called Upper Bay. One of the virtual servers in that site is named Upper Bay-Accounting.

There are a number of standard references to different labels and points that must be considered in every case. These are presented followed by the callout of several naming constructs for specific objects that have been shown to be useful.

Recommendations: Put the naming standards in place prior to the creation of any design or organizational objects in a repository and ensure that they are adhered to throughout the lifetime of the repository. Ensure that names are as descriptive as possible with consideration for the consumer of the label.

4.3.1 Display name

Display name is used to indicate that this label will be displayed on the user interface components. These display names will be utilized by the users of the system. These names are intended for human consumption and must have the proper white space and punctuation to make them the most meaningful to their intended audience. Display names are localizable, so consideration needs to be given to whether you will localize display names for your custom classes and properties.

4.3.2 Symbolic name

In contrast to display names, *symbolic names* are generally used programmatically to refer to particular objects. For that reason, symbolic names of various types must usually be unique within the type. For example, all symbolic names of classes within an Object Store must be unique. Symbolic names are not localizable.

Because of the uniqueness constraints for symbolic names, there is a convention for naming prefixes used by Content Manager and other ECM products. The purpose of this convention is to minimize the chances of a collision between names used in future product releases and your private symbolic names.

- ▶ Cm: Reserved for Content Platform Engine
- ▶ Dita: Reserved for Content Platform Engine for the FileNet P8 DITA Add-on
- ▶ RM: Reserved for IBM Enterprise Records
- ▶ EDM: Reserved for IBM eDiscovery Manager
- ▶ EDISC: Reserved for IBM eDiscovery Manager
- ▶ ICC: Reserved for IBM Content Collector
- ▶ Clb: Reserved for Social Collaboration
- ▶ CmAcm: Reserved for IBM Case Manager
- ▶ CmMcs: Reserved for Master Content Server

Check the product documentation for the latest list of reserved prefixes. You need to be aware that additional prefixes might be used by third parties who provide components for use with Content Platform Engine.

Recommendations: Use a unique prefix for symbolic names that you create. The choice of prefix is yours. It is typical to use something short but meaningful.

4.3.3 Uniqueness

Object names across the entire design generally have a requirement for uniqueness. Unique naming tracks with appropriate naming, that is, when proper consideration is given to naming objects, the uniqueness typically follows. Problems can arise when overly abstract names are given to an object where the same name more appropriately maps at a higher level in the hierarchy.

An example of naming an object Email implies that it is utilized high in the naming hierarchy whereas we expect a name, such as AgentCustomerEmail, is a good choice at a low level.

4.3.4 Taxonomy

Taxonomy is the establishment of categorization based on naming. Having a specific pattern that is applied to names with well-understood definitions for each name part facilitates an organized taxonomy. Giving initial thought to taxonomy and developing a taxonomy prior to the actual naming simplify the naming task and accent the self-descriptiveness of the names.

The best known example of a taxonomy is the scientific classification of living organisms, designating a name pattern that contains elements, such as species, family, phylum, and others.

The collection of metadata must achieve a balance between how it will benefit the user and the effort required to generate it. You need to consider the following information while collecting the metadata:

- ▶ What problems are you trying to solve?
- ▶ What kind of content and metadata do you need to solve the problem?
- ▶ How will you collect the data?

The primary purpose for standards and controls for taxonomies and metadata is to achieve consistency throughout the organization in the description of content objects to facilitate search and retrieval and overall object control.

4.3.5 Consistency

Consistency is important so that as the base of people who will be using the names is broadened, it leads to better understanding and less confusion as the system moves forward in scope and in age. Establishing good consistency standards is beneficial in the end. Consistency is facilitated by the complete application of the ideas that are already presented.

4.3.6 Object stores

Object stores are the highest point of naming for a certain repository. Make sure that you indicate the part of the solution that an object store represents when you name it.

For example, Company XYZ with a single object store can name its object store XYZ Enterprise. Another company ZYX has two object stores and it can name the two object stores, ZYX Operations and ZYX Support. The object stores represent repositories for all content pertaining directly to the business of ZYX: one repository for all of their internal administrative content and one repository for support organization content.

4.3.7 Storage areas

Storage areas are where the content is stored. There are various types of storage areas, including file system, database, and fixed content. Each type can represent a number of varieties, each with specific characteristics. Naming the storage areas in a manner that encapsulates the type and characteristics of the

storage area is useful because the storage areas are accessed and applied throughout the lifetime of the system.

For example, Company XYZ has three storage areas in use for the Company XYZ repository. The first storage area is a file store hosted on the network accessible protected storage segment of a storage area network (SAN) by a Network File System (NFS) mount. The second storage area is a fixed storage area that links to the company's image management system. The third storage area is a file store on a nearby set of inexpensive disk drives, also through an NFS mount. These three storage areas are named NFS-RAID, IMAGES, and NFS-CHEAP.

4.3.8 Document, custom object, and folder classes

When naming these objects, consider the inheritance hierarchy to both clarify the lineage of a specific object as well as to distinguish two leaf objects that might be the same type at first glance but have totally different lineages.

For example, memos from Engineering are classified under the `XyzOpsDevCommunicate` document class. Memos from Human Resources are classified under the `XyzSupHrCommunicate` document class.

Recommendations: Prefix the document class with the common prefix for the system, department name, and the purpose so that it is easy to find out the purpose of the class by looking at the name.

4.3.9 Property templates

There are special considerations for property templates because a certain property template can be widely used across many different objects. The names chosen for the property templates need to be self-descriptive of both the characteristics of the property template as well as the intended use of the template.

An example of three property templates is `XyzAgencyName`, `XyzFirstName`, and `XyzLastName`. These templates have multiple usages across different objects in a generic way.

Recommendations: Use the Category field in the property template for categorizing the properties.

Property template names need to follow the standard naming scheme and topology established at the enterprise level.

Property template names need to be generic enough that they can be used in a number of design classes but not so generic that they cannot be given a meaningful name.

4.3.10 Choice lists

Choice lists are ways to restrict the possible values of an integer or string property. They are used to limit the entries that the user will fill in for a property template. Choice lists need to be descriptive, informative names.

Recommendations: If you create applications that use IBM Content Navigator, considering using External Data Services (EDS) to control user interface elements instead of choice lists. An EDS is more flexible for application developers.

4.4 Populating a repository

In the solution domain, there are two major containers for data: the global configuration database (GCD) and the repositories, which are illustrated in Figure 4-1 on page 92. There is only a single GCD that encapsulates all of the configuration of the domain and at least one, but possibly many, repositories in the system.

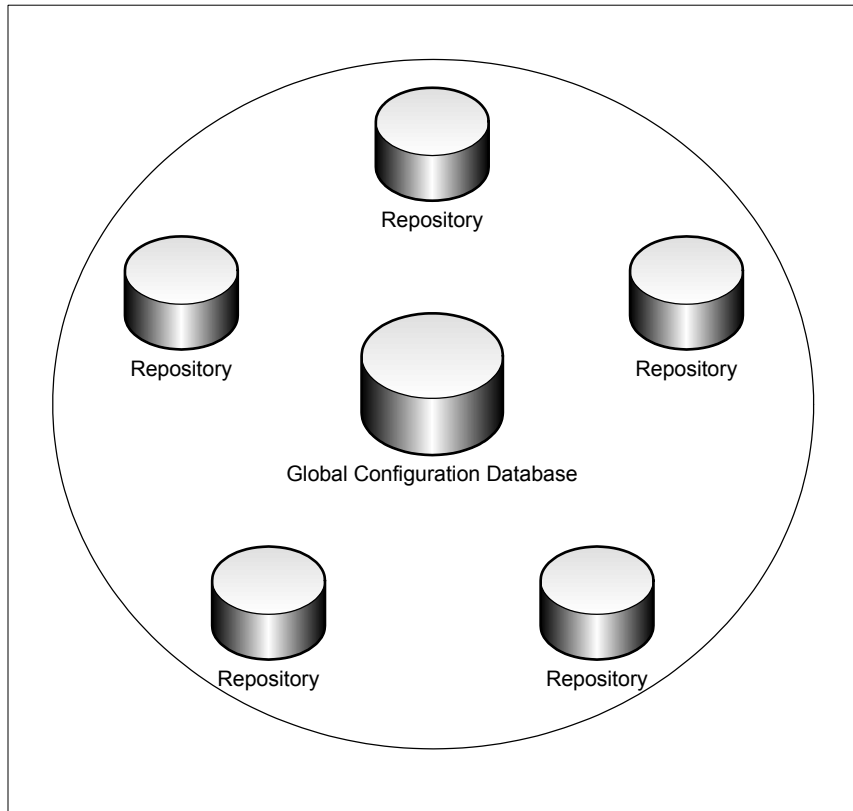


Figure 4-1 Storage objects in a domain

A repository contains a single object store and potentially one or more storage areas as shown in Figure 4-2 on page 93. An *object store* contains definitions, configuration information, and metadata for the content that is stored in the repository. The storage areas store the actual content.

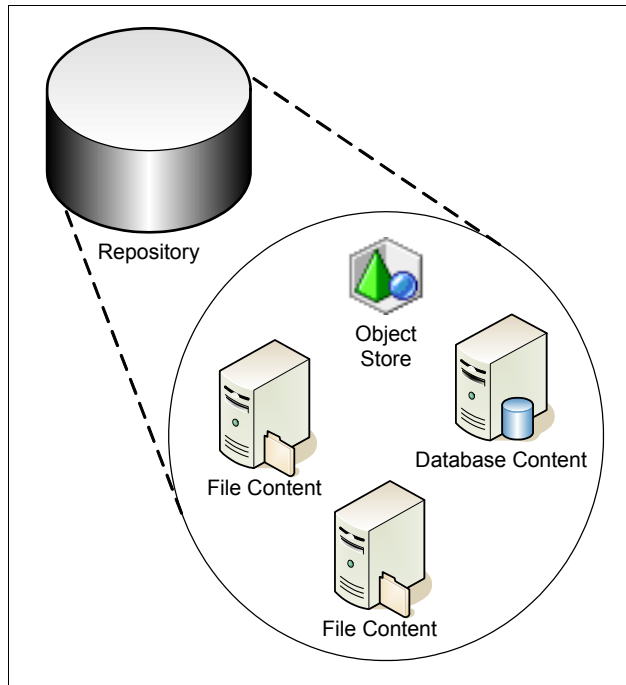


Figure 4-2 Repository contents

There are four major stages involved in the population of a repository: three design stages and one production stage. The three design stages include organizational design, described in 4.5, “Repository organizational objects” on page 95, repository design, described in 4.7, “Repository design objects” on page 98, and repository content design, described in 4.8, “Repository content objects” on page 117. The final stage in repository population is the actual test or production usage of the repository. The following sections describe the design stages and their relationships.

During all of these design phases, there are certain commonalities that are universally, or nearly universally, utilized in the objects of the design.

4.4.1 Generic object system properties

Generic object system properties are properties that are found in the lowest level of the object-oriented hierarchy from which all other objects are extended. All of the system properties are available to all the objects and do not need to be replicated in any custom properties. Therefore, you must understand what is available in order to leverage these properties where applicable.

Here, we list several of the system properties that have potential application in other places of the design:

- ▶ Class description
The class description contains the description of the class from which this object is instantiated.
- ▶ Display name
This label is intended for display to the user.
- ▶ Descriptive text
This text describes the purpose and meaning intended for this object.
- ▶ Is hidden
This is a Boolean value that indicates whether the object is hidden in its current context. This property can affect the user interface. Hidden objects or classes are generally of interest to application developers for special purposes but not of interest to users.
- ▶ Symbolic name
This label is used for internal, programmatic references to the object.
- ▶ ID
This immutable global¹ unique identifier (GUID) can be used to reference a specific object throughout its lifetime.
- ▶ Is content-based retrieval (CBR)-enabled
This is a Boolean value that indicates if content-based retrieval is enabled in the current context of the object.

In addition to the set of properties just covered that applies to all objects in the system, there is a set of properties that appears in many of the objects that is important to mention at this level. The following properties are present in most objects:

- ▶ Auditing enabled
This property indicates whether the object has its auditing enabled. This is a switch that enables and disables all audit logging for this specific object and its scope. Many events can be audited and controlled at a more granular level.

¹ In the P8 platform, GUIDs are only guaranteed to be unique in certain contexts. In other contexts, where uniqueness does not matter, GUIDs are created in a way that makes duplicates unlikely. For example, it is possible and harmless for a folder and a document to have the same ID value.

► Permissions

This property contains the access control list (ACL) for the object. An ACL consists of a number of access control entries (ACEs). A single ACE contains either an individual or group from the directory and the authorizations that entity has in relation to the object (See Chapter 5, “Security” on page 151 for more details).

4.4.2 Creating design elements

There are many design element types, such as document classes, custom object classes, folder classes, property templates, and GCD objects, that must be used in cooperation to achieve the best design. Each of these elements exists for a specific purpose and encapsulates a specific set of information. Because solutions are composed together from these elements, complex relationships can be created between them that must be maintained for system integrity and consistency. Most of the complexities of the relationships are handled by the underlying engine and removed from the concerns of administrators and application designers.

Modifying and removing design elements can be a tricky procedure, given the complex relationships that are possible. This is especially noticeable when attempting to remove a design element that might be used or referenced from a number of other design elements at differing levels of the design. It is always best to be as thorough as possible in the system design before actually creating the elements in the Content Manager. This thoroughness avoids most of these difficult situations.

Recommendations: Complete the design as much as possible prior to actually creating the design elements in the system.

4.5 Repository organizational objects

The solution space is divided into a number of logical divisions. Each division serves a specific purpose. The composition of all of these divisions provides a powerful solution that allows the requirements of any implementation to be clearly and succinctly decomposed.

Figure 4-3 on page 96 shows the logical relationships among the decomposition elements, domain, sites, virtual servers, server instances, and object stores.

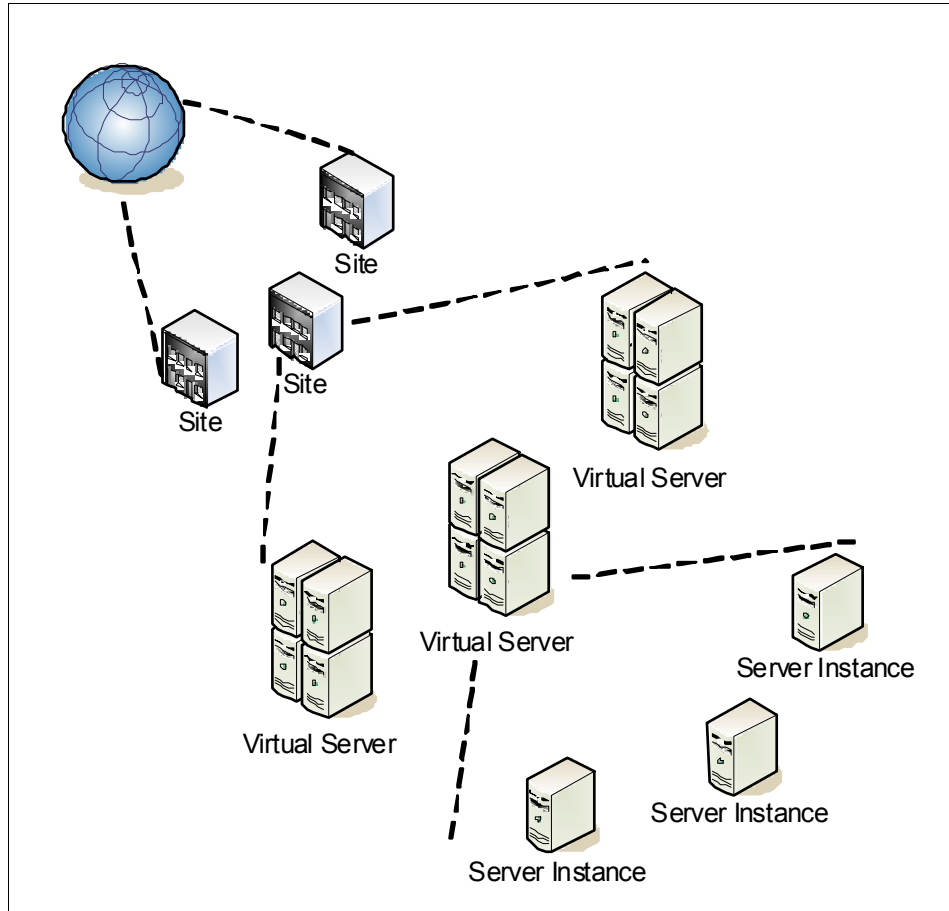


Figure 4-3 Repository organizational objects

All of the logical elements comprising a domain can be administered and managed through Content Platform Engine Administration tools. Naturally, all of the elements can also be manipulated using the APIs, but it is generally simpler to use the administration tools for most situations.

All these repository organizational objects were discussed in the Chapter 3, “System architecture” on page 37.

4.6 Global configuration database (GCD)

All of the repository organizational objects are contained in the GCD. The GCD is the single container that encapsulates all of the configuration information for a domain. The GCD is the logical representation of the domain, and it contains the subsystem configuration, which consists of the other organizational elements for sites, virtual servers, and server instances. In addition, it contains the specific configuration information for each object store's database space, directory configuration, text search server information, trace log configuration information, and other information that is accessible to all the object stores inside that domain.

Figure 4-4 gives a visual representation of the GCD layout.

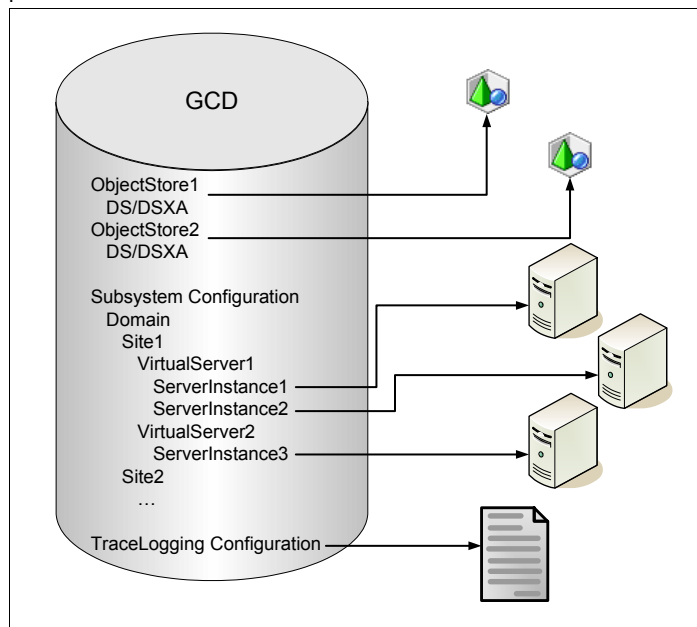


Figure 4-4 Global configuration database contents

4.7 Repository design objects

There are a number of elements that constitute a repository design. Each of these elements encapsulates a specific view, purpose, and role in the complete design. The division of responsibility between some of these elements is clear while others are highly dependent on the specific environment and application. There are a large number of design decisions that must be made to achieve a final design that is both efficient and scalable.

4.7.1 Object stores

In a similar manner that a domain encapsulates an entire repository solution, an *object store* is the basic component of a repository that contains not only all of the content that has been committed to P8 Content Manager, but all of the additional information and functional objects associated with that content. The number, type, and location of object stores that are needed for an organization are important design considerations (see 4.10, “Considerations for multiple object stores” on page 133 for additional details). Any object store is associated with a specific site and the storage areas associated with that site. The object store contains definitions for various classes that structure metadata, as well as actual metadata objects along with their connections to the content where applicable. An object store can contain all of the content for the entire enterprise or can be segmented from the overall enterprise design and assigned to a specific set of the overall problem. Regardless of the purpose, the object store contains the entirety of all of the definitions required for use by users and any applications that will access it. Figure 4-5 on page 99 shows a graphical representation of the scope of an object store.

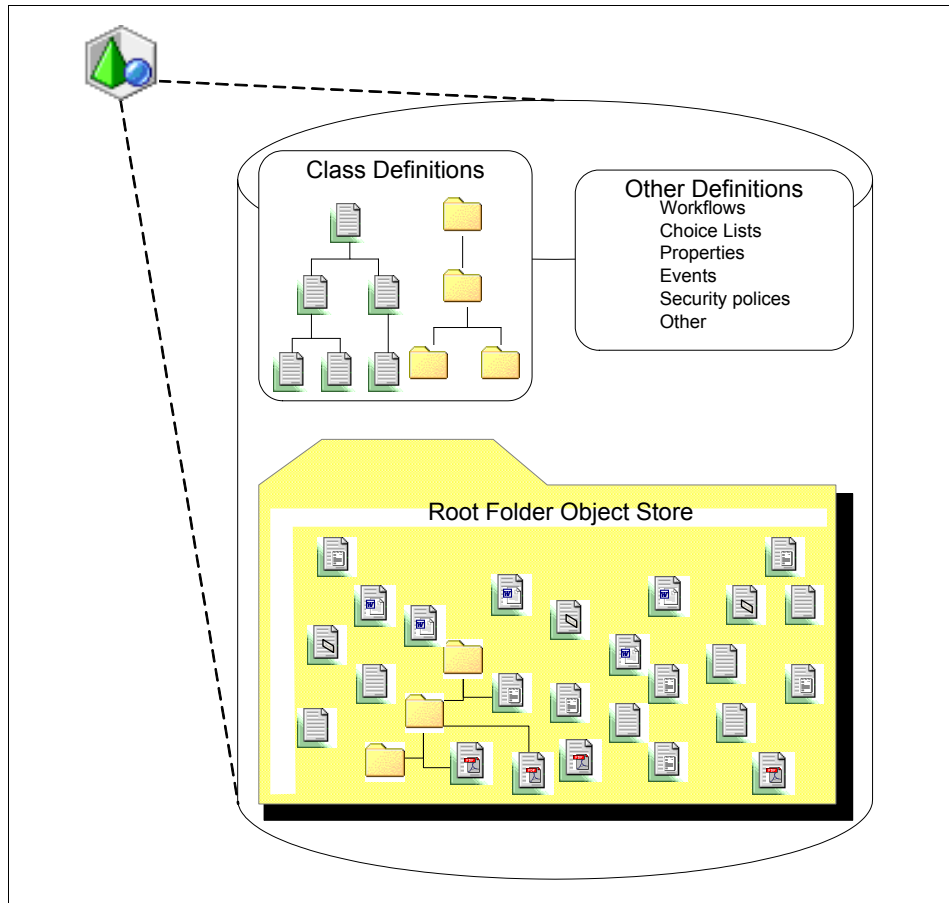


Figure 4-5 Object store contents

An object store is conceptually an object like all entities that make up a repository and that has specific characteristics. Object stores are created through the use of administration tools. The best practice is to utilize the wizard for object store creation, which simplifies the interface and ensures that all settings necessary at creation time are both set and synchronized where applicable.

Recommendations: If your design calls for more than a single object store, create a metastore that can contain all of the design objects that are common across all of the stores and replicate this as changes are made. If a meta object store is used, do not roll this store out into production, because it is strictly a development object store.

When creating an object store, *always* set the object store administrator to a valid administrator logon and grant the administrator all permissions.

4.7.2 Storage areas

Storage areas for repositories can be hosted on a wide range of storage devices and mediums, from SCSI drives, to fiber-attached SAN devices, to secure immutable storage units, as well as others.

In addition to storage media type, there are a number of logical storage types, such as database stores, file stores, fixed content stores, and cached content temporary stores. Each of these logical types has implications for performance and functionality that must be considered when determining specifically where to store content.

Storage areas have specific features to optimize the storage for space and other enterprise requirements.

Content compression

Content that is uploaded to the storage area is compressed if content compression is enabled for the storage area. Only the content that can be compressed beyond the compression threshold will be compressed. Content compression uses blocked-compression technology to divide the uploaded content into distinct blocks, which are compressed in memory before being written to disk. If encryption is also enabled, the block is first compressed and then it is encrypted in memory before being written to disk. Enabling content compression on a storage area will not encrypt the existing content.

Encryption of content

The content stored in the storage area can be encrypted using the storage area configuration. Content encryption helps protect the confidentiality of the content if it is accessed outside of Content Manager. A new key is generated every time that the encryption is enabled on the storage area. The most recent key is used to encrypt the new content. These encryption keys are stored in the object store database in a secured way.

Enabling content encryption on a storage area will not encrypt the existing content. In content replication, the external repository receives the non-encrypted content. Also, the decrypted content will be submitted for indexing purposes.

By moving content from one storage area to another, you can enforce the content encryption, re-encrypt with a latest key, or store non-encrypted content.

Suppression of duplicate content elements

The suppression of duplicate content can reduce the storage space that is required to store content. Content Platform Engine suppresses duplicate content by checking the existing content before adding new content to the storage area. If identical content exists, the new content is stored as a reference to the existing content. If no identical content exists, the new content is added in the normal manner.

Duplicate suppression is not available for fixed content devices, but those devices usually offer their own native duplicate suppression features.

Suppressing duplicate content might decrease storage space requirements but it also slightly increases processing time. To help you determine whether your space savings make the trade-off worthwhile, Content Platform Engine provides storage statistics for each storage area.

Database store

There is a single database store per object store, where the database store is part of the same database as the Object Store itself. The *database store* can be used to store content, but the content will be stored as a database binary large object (BLOB). Depending on the size of the content, this is not an effective use of the database and can have serious impacts on the database performance, especially with the large content.

File store

Recommendations: Using the database store can help with operational efficiency in some cases, but it almost always represents a performance cost over file storage.

There can be multiple file stores per object store with each one a separate directory structure on the server. The *file store* can be on local storage media or can be a mount point for remote, or networked, storage media. This is the typical location that is used for content with different file stores of different media types used for different content where appropriate.

Recommendations: The media type and cost must be clearly understood from the file store name to eliminate content storage area errors.

Fixed content store

There can be multiple fixed content stores per object store. This content type is designed to provide access to other content storage systems, such as an image repository, while leveraging the power of the Content Manager metadata management system. Fixed content stores can represent a physical storage appliance or can represent federated content.

Content cache store

A *content cache store* is a special store that allows local caching of content that is permanently stored in another storage area. A content cache is typically used for storage areas in a different geographic site. A content cache store allows local access to content that is frequently accessed, or in an active state of processing, to be available without degrading the network connection to the remote content and increasing the performance for these local operations. The cached content store provides a performance enhancement for remote content access but does not provide any type of high availability solution for the content. See Chapter 7, “Business continuity” on page 217 for more details about high availability solutions.

Although it is typically thought of as part of the configuration of a distributed environment, a content cache store can also give performance benefits locally if it is faster than the permanent storage for the content.

Recommendations: The Content Platform Engine guarantees that application access to cached content always follows the normal security controls, and it also guarantees that applications will not access stale content. Therefore, you can use a content cache area wherever it provides a benefit. You do not have to worry about compromising security or data integrity.

4.7.3 Document classes

Document classes are the design objects that, when instantiated, will contain most of the business content of the system. Most of the detailed design process is concerned with developing the correct set and hierarchy of document classes.

Document classes are inherited from a common top-level document class that contains all of the basic properties that the system needs. Although it is not technically necessary, it can be useful to create an immediate child subclass of document for enterprise-wide use. A top-level subclass for the enterprise can contain all of the metadata items that are the same across all document objects in the enterprise, either by requirement or policy.

The first level of document class design is concerned with the common enterprise objects, as opposed to specific application objects. The result of this first round of design is a hierarchical document class tree that contains all of the common enterprise document classes that can be leveraged by specific applications, because they are included in the Content Manager solution. A reasonable number of properties need to be defined in each class. It is easier to administer and expand a design where each document class is concerned with a specific aspect of the design. The resultant tree is typically neither extremely narrow, nor extremely wide. A narrow tree usually indicates that the class design has focused too specifically on an aspect and has been too exclusive. A wide tree usually indicates that there are too many aspects of the design encapsulated at a level.

Another test that can be applied to the resultant design is to see how various changes to the design can be made. If there are properties that have historically changed somewhat frequently or there are any properties that are projected to change, see what changes need to be made to the design to accommodate the changes. The ideal is to address a change with a change in a single class. This is a good indication that you have the proper level of design encapsulation. The types of changes to consider are property redefinitions, property additions, property deletions, class additions, class modifications, class deletions, security updates, functional changes, and organizational changes.

Recommendations: Avoid making many subclasses for a custom document class. Changes to a higher level custom document class will be propagated to all its subclasses.

Adopting an enterprise perspective allows the document class designs to facilitate greater information sharing and collaboration across the enterprise. In addition to assisting in breaking down information silos, this makes the overall design much more usable as well. You must always take usability into consideration during all the design phases. The use of SMEs at this phase can greatly assist you in meeting the unspoken requirements and usability goals of users.

As a key design object in the system, there are lots of additional components on which the document classes are dependent. Most of these dependencies are covered in the specific sections for the dependent elements. Probably the most important dependency is the usage of the property templates in the class designs. This dependency underscores the need to be clear and concise in the property template definitions and consistent with naming and topology across the entire design.

Finally, try to avoid designing for the current organization without being modular enough to accommodate change. Avoid carrying over limitations of the current system that might have been design flaws in the current system or limitations of the tools that are used to support it. Take into account any current or future processes in which the content is utilized. That is, always consider business process automation in the design. Remember that there will always be additional applications and functional areas that the system will need to support that are not currently identified or even identifiable.

There are three focus areas that the document class design typically follows: design based on organization, design based on content, and design based on function. Although these are the major design approaches that are used, variations on these themes as well as modifications and combinations of these approaches are also successfully used. The correct approach to use is highly dependent on the specific details of your corporation and the application that is supported by P8 Content Manager:

► *Design based on organization*

Design based on organization starts with the first level of decomposition after the enterprise root document class, which is groupings around how the corporation is organized. This can be reflected in line of business (LOB) objects, support and business value objects, or any other high-level structure that represents your organization. The subsequent layers of the hierarchy then follow the organization down into smaller and smaller groupings. Each level can also have classes that capture content-specific aspects where the document content that they represent has consistency across the entire organization from that root down the hierarchy. Eventually, the lowest level represents document content classes that correspond to specific functional areas or specific content.

This facilitates future changes that occur at the organizational level by capturing these aspects as high in the tree as feasible and letting these properties and attributes be inherited down the hierarchy.

► *Design based on content*

Design based on content starts with the first level of decomposition after the enterprise root document class. The first level includes high-level abstractions of the content types that will be stored in P8 Content Manager. This often follows record plans where they have been established. Lower levels of the hierarchy allow the capture of more and more concrete aspects of the content types until the resultant leaf nodes are declared.

This approach facilitates communication across the enterprise, because all of the properties of the document classes will be the same regardless of where in the organization they are used. You do not capture the organizational aspects of the corporation. This design approach can have significant political ramifications dependent on the culture of your corporation.

► *Design based on function*

Basing design on function starts with the first level of decomposition after the enterprise root document class, consisting of abstractions of the functions that are carried out in the corporation without regard to the organizational structure. As the document class hierarchy extends down, more and more concrete functional aspects are captured, as well as content-specific aspects for the content types that will be used.

This approach captures many of the functional aspects of the corporation, which typically mirror the organizational structure, but in a more abstract perspective of focusing solely on the function, business value, and processes for which the content is used. This approach is sometimes viewed as a blending of the purely organizational approach and the purely content approach.

Document classes are created through a wizard interface in Content Platform Engine Administration tools in the metastore. After the metadata of the metastore is finalized, all the metadata will be exported and imported to the other development, test, and production systems using the deployment tool as described in Chapter 9, “Deployment” on page 271.

Recommendations: There needs to be a single, top-level document class that extends the base document class and from which all other document classes will be derived.

All property templates, choice lists, storage policies, and storage areas need to be created prior to creating any document classes that utilize them.

Each document class encapsulates a single design aspect.

Create all the metadata in the meta object store and export and import the metadata from the metastore by using IBM FileNet P8 Deployment Manager as described in Chapter 9, “Deployment” on page 271.

Never skip the step of designing high-level abstract objects that are for aspect encapsulation and that most likely are never instantiated.

Document class characteristics

Document classes have the following characteristics:

- ▶ Have metadata
- ▶ Are containable
- ▶ Are versionable by both content and metadata
- ▶ Hold content

4.7.4 Folder classes

Folder class objects are the design objects that, when instantiated, provide aggregation or containment for other objects. The characteristics and usage of folder objects must not be mistaken for, or confused with, the foldering features and concepts provided by a file system. P8 Content Manager folder classes provide containment by reference, which allows any specific object to be contained in multiple folders at the same time. Most of the same considerations that are given to creating document object classes (4.7.3, “Document classes” on page 102) also apply to designing folder classes:

- ▶ Single top-level class that all others are derived from.
- ▶ Single design aspect captured per class.
- ▶ Design with changes in mind.
- ▶ Design in modularity.
- ▶ Do not repeat any mistakes that the current system or processes have.

A key design decision that needs to be made is whether the main access mechanism for content follows the search paradigm (represented in Figure 4-6) or follows the browse paradigm (represented in Figure 4-7 on page 108). Both of these paradigms offer their own strengths and weaknesses, and this decision directly affects how folder classes will be used and instantiated.

Search paradigm

The model for the search paradigm is represented in Figure 4-6 as a dialog box requesting some information and returning a set of content that meets the criteria specified in the dialog. The best analogy is accessing a database. Information is retrieved from a database by formulating a query, which returns a set of data elements that matches the criteria in the query.

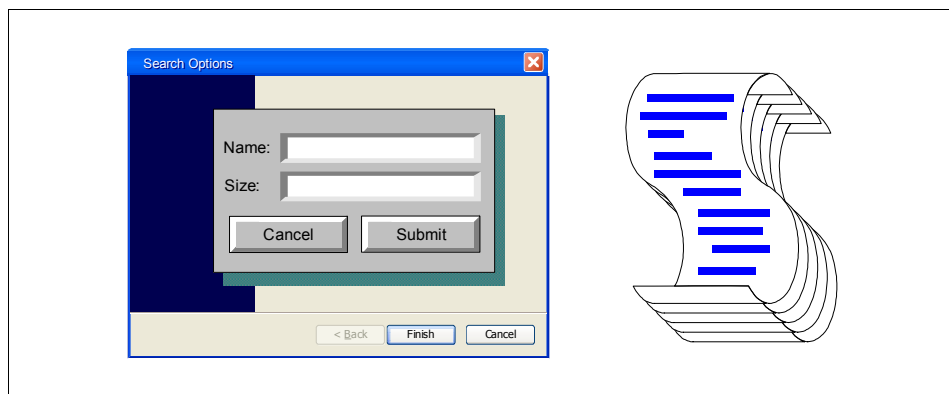


Figure 4-6 Searching for content

The search paradigm is powerful, because it does not rely on the user needing to know where the content is in the system or the name of the object that contains the content. Searching also returns a set of objects as an atomic operation; the maximum size of this set can be controlled as well. This can include objects that are in diverse places in the repository. Effective use of the search paradigm requires the selection of meaningful distinguishing properties for the objects that have meaning to users. It also requires meaningful document classes that are understood by users as well.

The search paradigm can be fronted with various methods of compiling the search criteria and usually is best served by designing searches or through custom interfaces. It is usually a faster and more reliable method of finding content than is offered by the browse paradigm.

Recommendations: Use a search paradigm to search for documents with as much metadata information as possible to get a small result set. As much as possible, avoid using wildcard searches, which give you a large result set.

Browse paradigm

The model for the browse paradigm is represented in Figure 4-7 as a typical file system structure. There is some meaningful relationship between the sets of folders that lead the user to sets of content in an understandable way. The best analogy is a file system tree structure. Although the analogy presented to help understand the browse paradigm is a file system structure, a file system folder is *not* the same as a P8 Content Manager folder, which supports multiple filed locations.

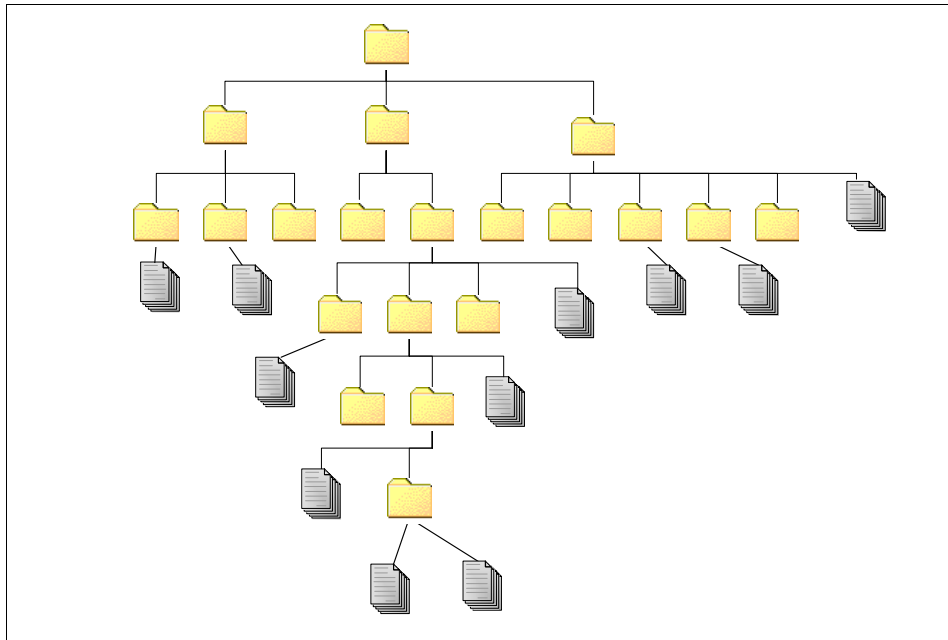


Figure 4-7 Browsing for content

The browse paradigm relies on the users who add the content to be thoughtful and knowledgeable in the manner in which the content is filed. This potentially includes filing the same content object in multiple folders. There is also a requirement that the name of the content object has a meaning in its context that is understood by users.

The browse paradigm can increase the time that it takes for the system to search for content, but it is well suited to users who all understand the basic concepts of foldering and are used to using foldering for file system access. The browse paradigm typically takes longer for users to find content than the search paradigm, and it requires users to have inherent knowledge to be able to reliably find content.

Recommendations: In most cases, the search paradigm offers a much better model for performance and maintenance. Avoid too many layers of too many folders (keep the total number to tens of folders, not hundreds), which can impact retrieval performance. There needs to be a single, top-level folder class that extends the base folder class and from which all other folder classes will be derived.

All property templates and choice lists must be created prior to creating any folder classes that utilize them. Create all the metadata in the metastore and export and import the metadata from the metastore using the deployment tool.

When using a browse paradigm, make sure to file the documents in a meaningful foldering hierarchy. Each folder class encapsulates a single design aspect.

Folder class characteristics

Folder classes have the following characteristics:

- ▶ Have metadata
- ▶ Are containable
- ▶ Are not versionable
- ▶ Are not content
- ▶ Are containers

4.7.5 Custom object classes

Custom object classes are design objects that contain metadata without content and provide no containment. They are designed to be versatile general-purpose objects that can be subclassed to perform a variety of functions, such as security proxies, or configuration objects for workflow processing. Custom objects are database objects that do not have any content.

Most of the same considerations that are given to creating document object classes (4.7.3, “Document classes” on page 102) also apply to designing custom object classes:

- ▶ Single top-level class from which all others are derived.
- ▶ Single design aspect captured per class.

- ▶ Design with changes in mind.
- ▶ Design in modularity.
- ▶ Do not repeat any mistakes that the current system or processes might have.

Recommendations: There must be a single, top-level custom object class that extends the base custom object class and from which all other custom object classes will be derived.

All property templates and choice lists need to be created prior to creating any custom object classes that use them. Each custom object class encapsulates a single design aspect.

Custom object class characteristics

Custom object classes have the following characteristics:

- ▶ Have metadata
- ▶ Are containable
- ▶ Are not versionable
- ▶ Hold no content

4.7.6 Custom root classes

Custom object subclasses are essentially a collection of properties. The disadvantage with instances of custom object class is that these will be stored in the *Generic* table in the object store database. This might cause a performance degradation if several instances of disjoint custom object classes exist in the *Generic* table. Querying for one instance of the subclass might be slowed because of the presence of a large number of instances of other custom object subclasses. Sometimes, it is not always practical to create an index on the *Generic* table since many of the columns might have null values because those columns relate to other classes.

To overcome the issues with custom objects, users can use the custom root classes. Every immediate subclass of the custom abstract root class has a separate table in the database. The table name will be generated from the symbolic name of the custom root class.

Three kinds of custom root classes can be created: abstract persistable, abstract queue entry, and abstract sequential. All these base custom root classes are abstract and cannot be instantiated directly. Also, Content Manager does not let users update the properties of these classes. Users create subclasses from these abstract classes in order to use them in the application. It is from the

immediate subclasses that the tables are created. Any instances of subclasses of the custom root classes will be saved in the same table as the custom root class table:

- ▶ Abstract persistable

It is similar to a custom object class, which is a collection of properties without any content associated with it. The instances of this class cannot be filed into any folder. The immediate subclasses of abstract persistable are each saved in a different table.

- ▶ Abstract queue entry

The subclasses of abstract queue entry are intended for the queues managed by the sweep framework. This class has additional properties that are required for the sweep-based queue operations. The abstract queue entry classes will follow the security model for queue item and replication. Access is defined by the default instance permissions. There is no owner or permission property on these instances.

- ▶ Abstract sequential

Abstract sequential is for the external applications' queue and log processing. It provides a single increasing sequence number property that can be used to process the entries in the order in which the transactions were created.

Important: The two subclasses created by any custom root class are disjoint because they are in separate table. Deleting the class definition for a custom root class will drop the associated tables.

4.7.7 Property templates

Property templates are used throughout the design as established containers for properties. A property template contains a name, a property data type, and a set of attributes. This enables the definition of common properties to occur once and be utilized throughout the design in a uniform manner. Properties, such as `FirstName`, `LastName`, and `PolicyNumber`, are typical generic property templates in a design.

There are two types of properties: the system properties that come preinstalled in P8 Content Manager and custom properties that you create for your specific installation. All of these properties can be utilized in any definitions as you think appropriate. Typically, there is a rich set of system properties associated with the base classes. The system properties that are, by default, associated with a class must be examined to both prevent duplication of information and to understand what is available to be leveraged by your class definitions.

Property templates must always have a data type associated with them. The data type can have a cardinality of either single value or multi-value for all data types.

Recommendations: Property templates need to follow a standardized naming scheme and topology established at the enterprise level. Property templates need to be generic enough that they can be used in a number of design classes, but not so generic that they cannot be given a meaningful name.

Avoid the creation of property templates that are named in such a manner that it might be confusing to know which template to use. Avoid the creation of property templates that encapsulate the same informational data but have distinct names.

4.7.8 Choice lists

Choice lists are defined to limit users from being able to enter free-form text or integer data into a property. Choice lists protect against typing mistakes and other human errors. It is not always appropriate to use a choice list, because there must be a well-understood, mostly static set of data that the property can take.

Choice lists can consist of levels of groupings of values to make it easier for the correct value to be selected. In multi-value properties, the user can select multiple entries from the choice list.

Recommendations: Group choice list elements logically with the user experience in mind. Limit the number of elements in each group to a small enough set that it can be easily displayed and scanned.

Avoid assigning the same value to more than one item in a choice list. Do not use choice lists for properties where the values are expected to change frequently.

4.7.9 Annotations

Annotations allow users to link additional information or comments to other objects, such as documents. These annotations can be in any format, such as text, audio, video, image, highlight, and sticky note. An annotation's content does not necessarily have to be the same format as its parent document and can be published separately. Document annotations are uniquely associated with a single document version; they are not versioned or carried forward when their document version is updated, and a new version is created.

You can modify and delete annotations independently of their annotated object. However, you cannot create versions of an annotation separately from the object with which it is associated. By design, the annotation will be deleted whenever its associated parent object is deleted. Annotations receive their default security from both the annotation's class and the parent object. You can apply security to annotations that is different from the security applied to the parent.

The content of annotations is stored in a storage area, as defined by the default area for the annotation class. The storage area used by the annotation class needs to be appropriate for the type of content associated with the annotations. That is, if a large content is being used for annotations, it must not be stored in the database storage area.

4.7.10 Document lifecycles

Document lifecycles allow for the fact that a certain document exists in a number of states throughout its lifetime. Figure 4-8 on page 114 shows a sample state diagram for the typical document lifecycle in the XYZ corporation.

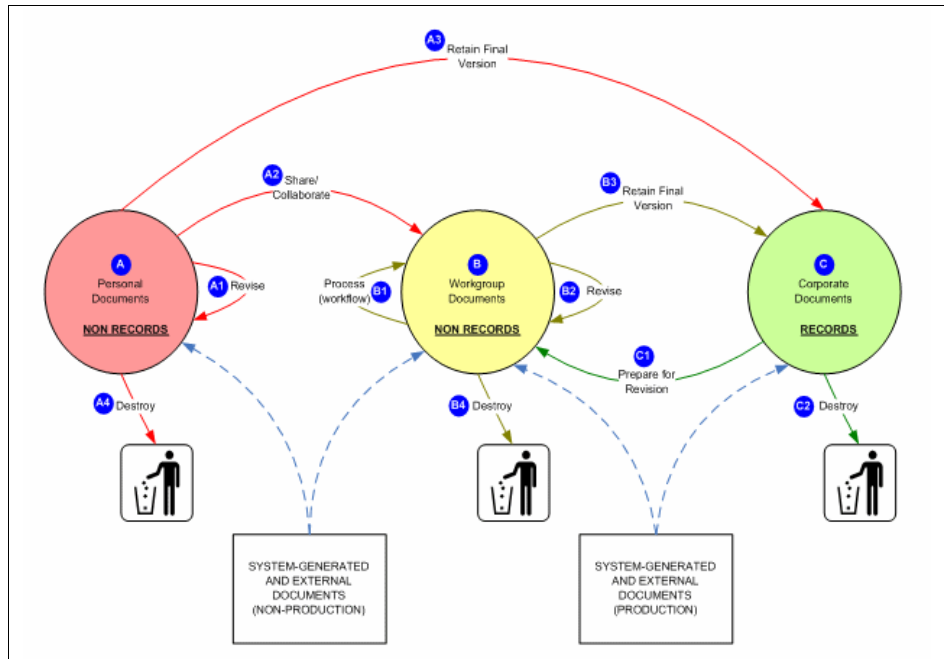


Figure 4-8 Sample document lifecycle model

In this example, documents are in one of three states:

- ▶ Personal documents not being shared or collaborated on
- ▶ Workgroup documents that have a limited scope of sharing and are intended for collaboration
- ▶ Corporate records that have meaningful business value to the company

In the first two states, a document can be revised and remain in its current state, reach its end of life and be destroyed, or be promoted to a higher state. In the workgroup collaboration state, documents can also be processed in some automated way, such as through IBM Case Foundation (previously known as IBM FileNet Business Process Manager). In the final corporate document state, a document can also be demoted back to the workgroup for revisions and updates.

While the figure captures the states and transitions between the states that a document can take, it also illustrates how IBM FileNet Content Manager document lifecycles can be extremely useful. A document lifecycle allows for the definition of the states in which a document can be and then can associate that document with a set of security templates that depend on the state that the document is in. This controls the access of the document as it progresses from being a personal document, to a workgroup document, to a corporate document.

Document lifecycles are contained in two design classes: the lifecycle policy class and the lifecycle action class:

- ▶ Lifecycle policy class
The definition of the document's states. The policy also identifies the lifecycle action that executes in response to the state changes.
- ▶ Lifecycle action class
Action that the system performs when a document moves from one state to another.

Document types in the Content Platform Engine have default lifecycle policies. You can also assign a default lifecycle policy to any new document class. When you create a document using a class with an associated lifecycle policy, the document uses it as a default lifecycle policy. This can be overridden at creation time by assigning a different lifecycle policy to the document.

Recommendations: Assign lifecycle policies to a document class whenever possible, instead of assigning them to individual documents. This practice helps the operator select the correct policy by choosing the document class associated with the desired lifecycle policy. This practice also prevents problems that can occur if you need to delete a lifecycle policy.

4.7.11 Events and subscriptions

Content Platform Engine Administration tools enable you to define events that extend the functionality of an object store, which enables you to configure objects to perform actions in response to specific activities that occur on each object defined on a Content Platform Engine server.

An *event* consists of an event action and a subscription. An *event action* describes the action to take place on an object. A *subscription* defines the object or class of objects to which the action applies, as well as which events trigger the action to occur. Subscriptions can be assigned to individual objects. However, it is more efficient if they are assigned to classes instead. Assigning subscriptions to classes ensures that a set of common objects is managed consistently. Assigning subscriptions to classes can also limit the number of subscriptions that run simultaneously, which can affect system performance.

Recommendations: Add properties to subclass event actions and subscriptions. Keep event actions short to ensure quick completion. This is especially true for synchronous subscriptions where the subscription processor waits for an event action to complete before moving on to subsequent processing.

Do not rely on priority to guarantee the order of execution for subscriptions. Ensure that you thoroughly test your events and subscriptions before implementing them.

Set up each event action with code stubs that specify each event trigger (Create, Update, Delete, CheckIn, CheckOut, File Event, Unfile Event, Mark for soft delete, and recover to restore from recycle bin), even if you do not define functions for every trigger. The subscription controls which of the triggers call an action. You need to prepare the action to handle all triggers gracefully.

4.7.12 Marking sets

Marking sets are intended for records management applications. They allow access to objects to be controlled based on the values of specific properties. The ACL for an object with a marking set is a combination of the settings of its original ACL and the settings of the markings constraint mask for each marking that is applied to it. The result of this combination is the effective security mask. It is important to note that marking sets are only subtractive in nature, that is access can only be denied or removed through marking sets. Refer to 5.3.6, “Markings” on page 167 for more information about marking sets.

The general mechanisms of marking sets include:

- ▶ A marking set is defined that contains several possible values called *markings*.
- ▶ Each marking value contains an ACL that defines who can assign that specific value to an object property, who can modify that value, who can remove that specific value, and who will have access to the object to which it is assigned.
- ▶ The marking set is assigned to a property definition that is assigned to a class. All instances of that class must have this marking property set to one of the markings defined in the marking set.
- ▶ The value for the marking properties can only be assigned by users authorized by the associated marking.

- ▶ Markings do not replace conventional access permissions on an object, but rather are coequal with them in determining access rights. If an object has one or more markings applied to it in addition to one or more permissions in its ACL, access to that object is only granted if it is granted by the permissions *and* by the markings.

The number or size of markings in a single marking set is limited by available system memory. To perform an access check on a marked object, the entire marking set and all its markings must be loaded into memory. This is not going to work if there are millions of markings. For this reason, limit the number of markings in a marking set to no more than 100.

Recommendations: Marking sets need to contain no more than 100 markings. Marking sets are domain objects. Having more marking sets affects the performance and the memory footprint of Content Platform Engine server.

4.8 Repository content objects

Part of the repository design process also involves how the content will be organized and laid out in the repository. You need to decide how to structure the objects that are instantiated from the design classes.

This section describes points for you to consider when laying out the content in the object store.

4.8.1 Folder objects

Folder objects can be participants in both sides of aggregation by reference. Because the foldering concept in IBM FileNet Content Manager is done by reference, and a containable object can be referenced in multiple locations at the same time, this can be an extremely powerful tool to meet sophisticated requirements. In general, if the search paradigm is followed, folder objects serve a purpose for actual reference aggregation and an additional layer of security for those aggregated objects. For many clients, a primary reason for installing a central repository is to bring scattered information into an organized structure.

Referential containment relationships

A *folder* is a container that can hold other objects. These objects can be custom objects, documents, and folders and their subclasses. Child folders are typically directly contained. That is, their containment model is a one-to-many relationship.

A containing folder can contain multiple child folders, but each child folder is directly contained within at most one parent folder. Custom objects and documents are always referentially contained. For referentially contained objects, their containment models a many-to-many relationship. A referentially contained object can be contained within multiple folders, and can also be contained multiple times in the same folder.

There are two types of referential containment relationships: dynamic and static. A *static referential containment relationship* is a relationship between a folder and a custom object, a specific document version in a version series, or a folder. A *dynamic referential containment relationship* is a relationship between a folder and the current version of a document. In this case, the current document version is the released version, or else the current version, otherwise, the reservation version.

Filed as opposed to unfiled

In an IBM FileNet P8 Content Manager (P8 Content Manager) repository, content objects can be added to a repository in two ways: without reference to a folder structure or into a particular folder (or set of folders). We refer to these options as *unfiled* and *filed* (see Figure 4-9). Unlike a file system, repository folders do not represent physical locations in the repository. Content is added, indexed, and is accessible whether or not it is filed in a folder.

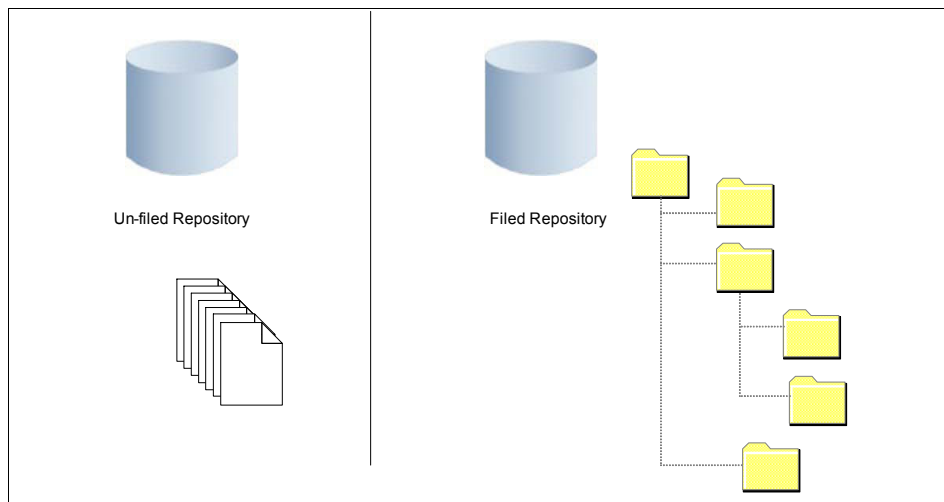


Figure 4-9 Repositories can be filed or unfiled

One of the primary benefits of filing into a folder is browsing. *Browsing* allows users to traverse a folder structure and locate content inside a folder. Hopefully, all the content in any specific folder relates to a particular activity or function. Another advantage is that in a P8 Content Manager repository, content can be filed in more than one folder at a time. There is one master copy of the content, and references filed in multiple folders point back to the single master.

Remember that with P8 Content Manager, users can always search for and view any content that meets search criteria whether or not the content is filed in a folder. Folders are simply a convenience for users who want to browse for repository content.

There are use cases where unfiled content makes sense. Table 4-1 is a decision table for the filed option as compared to the unfiled folder option.

Table 4-1 Folder options and their impact

| Folder option | Impact |
|---|---|
| Unfiled content (does not use folders) | <ul style="list-style-type: none"> ▶ Content is only accessible by search. ▶ There is no need to organize repository content using folders. ▶ Transactions that add content are slightly faster. ▶ Appropriate for high-volume image applications where access will be by search only. |
| Filed content (uses folders) | <ul style="list-style-type: none"> ▶ Users can browse or search for content. ▶ There is a need to organize the repository content. ▶ A single version of a content object can be filed in more than one folder. ▶ Appropriate for lower volume applications, or applications where users will be manually adding content. |

Organizing unfiled content

The P8 Content Manager repository can act as a receptacle for high-volume archive systems for image (scanned paper) or email messages. For these applications, folders and an organization scheme are not a priority. The “add content” transaction in P8 Content Manager is slightly faster when foldering is not required. In this type of solution, transaction rates and efficient searching are the most important criteria.

In solutions of this kind, searching becomes the primary mechanism for content retrieval. For this reason, the metadata that identifies the content when it is added to the repository is vital.

The metadata set that is collected for each content item must include all properties necessary to identify and retrieve the content. This set must include the usual properties, such as content title, content subject, and date collected, in addition to application-specific properties, such as customer name, customer ID, and account number.

Organizational metadata elements

You must also consider another set of metadata. You can add metadata properties that provide organization for the content. Organizational metadata identifies the type of content, the division or department to which it belongs, and potentially, the record series that controls its retention. The following list shows examples of organizational metadata properties:

- ▶ Division
- ▶ Department
- ▶ Function
- ▶ Activity
- ▶ Document type
- ▶ Record type

Adding organizational metadata tags to repository content is a valid method of providing a central structure to repository content without using folders. You can add the same elements that create an efficient folder structure to unfiled repositories as organizational metadata properties.

Repository folder structures

The design of a central repository is an opportunity to place scattered content into an organization-wide filing system. One of the primary functions of a repository is to offer ease of access; users must be able to quickly locate information with a minimum of effort.

Several parameters contribute to a well-designed repository folder structure:

- ▶ Is the structure self-explanatory? Is it easy to locate information?
- ▶ Does the structure work for all groups in your organization?
- ▶ What about groups that want to create their own folder structure?
- ▶ Does the structure avoid placing too many folders in a single subdirectory?

We will consider these questions as we move forward in this section.

An organization-wide folder structure

A central repository folder structure must make sense for all groups in your organization. During implementation, it is not necessary to build out the entire folder structure; the first three levels are sufficient. The goal for the first three folder hierarchical levels is a structure that is accessible at first glance to any member of your organization.

Recommendations: When designing a central repository folder structure, we suggest that you start with the first three levels of the structure. Build this out for your entire organization.

The first three levels of the folder hierarchy form the central organization scheme for your repository. Three levels are not an absolute rule; four or five levels might be necessary for large organizations. The idea is to create a structure that provides an organizational foundation. Depending on your organization, there are several approaches for organizational schemes. The best way to illustrate this concept is through examples.

Example: By organizational chart

The first example is a folder structure that follows a company organizational chart. In this organizational scheme, as shown in Figure 4-10, the folder levels represent:

(1) Department → (2) Activity → (3) Document type

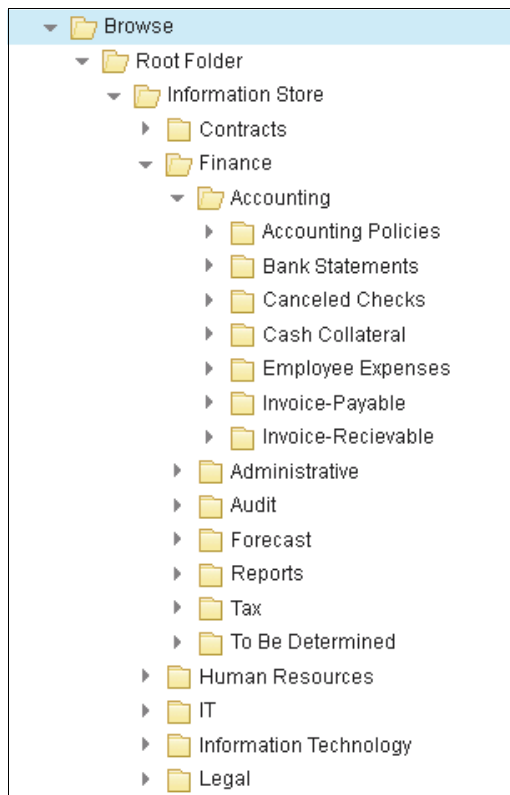


Figure 4-10 An organizational folder structure

Example: By geographical location

Another example is a repository that stores construction project records. For this organization, construction projects are organized by location (see Figure 4-11). In this scheme, the folder levels are:

(1) Region → (2) Construction project → (3) Document type

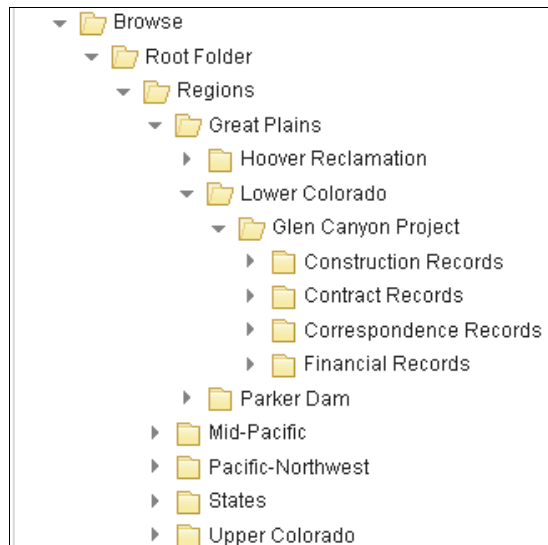


Figure 4-11 A geographical folder structure

Example: By function

The next folder structure is based on function. This structure is appropriate for records systems that are typically organized by the function of the document, the activity with which it belongs, and the record category under which it needs to be filed. In this scheme, as shown in Figure 4-12 on page 123, the folder levels are:

(1) Function → (2) Activity → (3) Document type

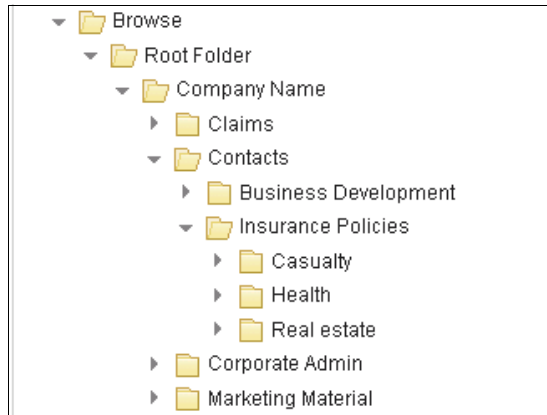


Figure 4-12 A functional folder structure

Recommendations: Create a folder structure that makes sense for your entire organization. Develop your folder structure to at least the third folder hierarchical level. This structure forms the framework for your repository organizational scheme.

Beyond the third level

The goal of the three-level folder hierarchy is to impose an organization-wide structure for repository content. But many times, individual groups have their own requirements for folder structures and want to organize their content without system-enforced rules. For these groups, simply release the organizational rules for any folders created under the third level.

Folder-inherited security allows repository administrators to restrict the creation of folders in the first, second, and third folder hierarchical levels and grant folder creation privileges to group owners below this level. This enforces the integrity of the organizational scheme, while still allowing individual departments to organize content to their own satisfaction.

In the example in Figure 4-13 on page 124, folder creation rights to levels 1 - 3 need to be reserved for system administrators only. Folder creation rights under the accounting folder need to be granted to the accounting group manager, and folder rights under projects need to be granted to the IT group manager.

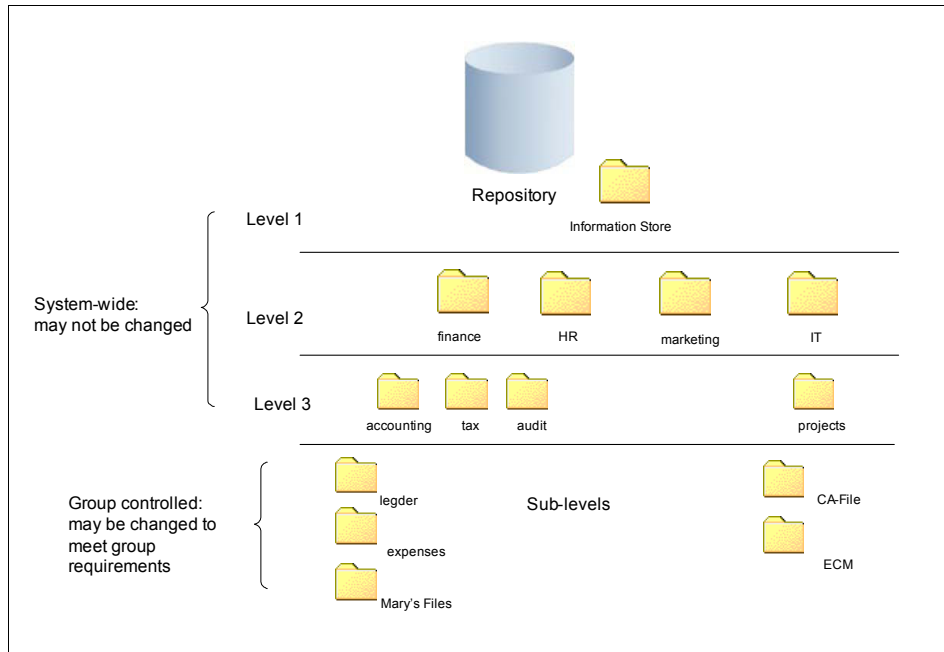


Figure 4-13 Folder creation rights in an organization-wide folder structure

Avoiding an excessive number of subfolders

It is possible to create too many subfolders under a parent folder. For all implementations, avoid creating more than 100 - 200 subfolders under any specific repository folder.

In any foldering application, large numbers of subfolders create performance problems. The system slows down when users open the parent folder and an excessive number of subfolder entries must be queried and returned from the database. The design goal is to create a deeper hierarchy rather than an overly shallow structure.

Recommendations: Limit the number of subfolders at every folder level. Create a hierarchy of subfolder levels rather than using many folders at the same level. Do not create a folder that contains more than 100 - 200 subfolders. Use a search paradigm wherever is appropriate, and limit the search result size.

4.8.2 Other objects

When you define other objects in the repository, consider how they are intended to be used and leverage their unique abilities. Another aspect of object repository layout is in the storage media that the content will use. Try to provide a range of storage media and use it appropriately.

4.9 Storage media

Recommendations: Try to give meaningful name properties to objects to assist users in navigating through collections of documents returned in a search or a browsing session.

Try to match content with storage media in a meaningful way. Internal memos and other short-lived pieces of content without lots of business value can be stored on a simple network-attached storage (NAS) device. Content that is critical to the business operation can utilize a high-speed, highly available storage subsystem that also has a higher cost associated with it.

A P8 Content Manager repository stores data in two areas: the object store and the storage area. The *object store* is a relational database that stores repository configuration: object references, properties, choice lists, and object relationships. The *storage area* holds actual content: electronic media files. Object stores can be configured to use three distinct types of storage (see Figure 4-14 on page 126):

- ▶ Database store
- ▶ File store
- ▶ Fixed store

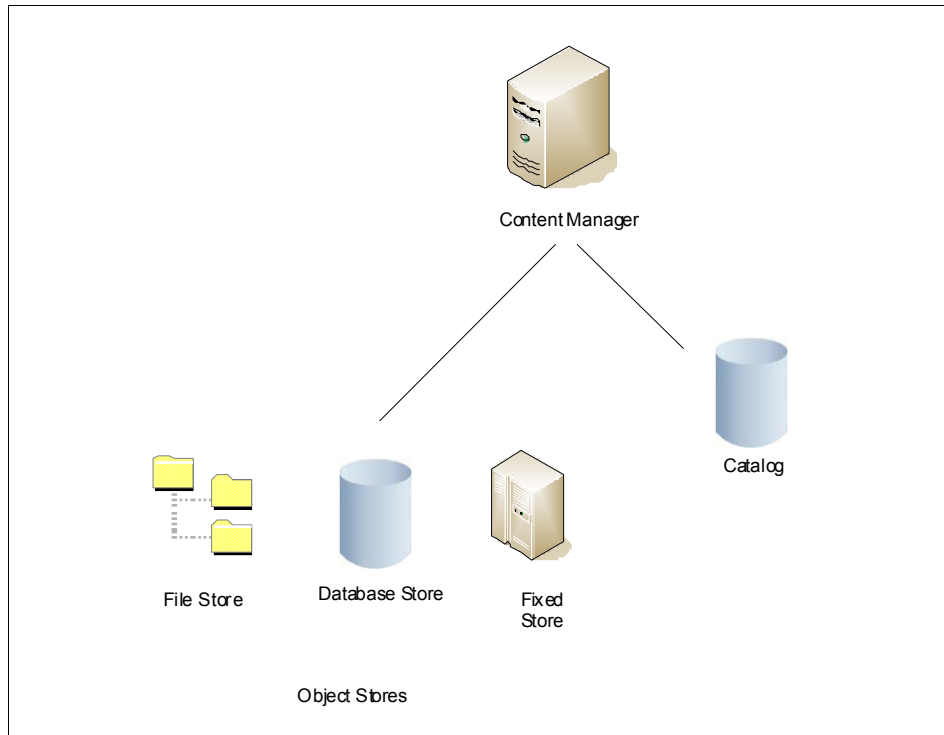


Figure 4-14 P8 Content Manager object store storage options

When choosing a storage method for your content, remember that each of these storage methods can be configured on a per document class basis.

4.9.1 Catalog

The *catalog* is a relational database that is specified at installation time. The catalog can be created on any supported relational database management system (RDMS). Refer to the product documentation for information about supported brands and versions.

The catalog database stores all of the P8 Content Manager configuration information. If you expand the object store view using Content Platform Engine Administration tools, the object tree that displays is pulled from the catalog database. The catalog stores the following information:

- ▶ Configuration information
- ▶ Object references
- ▶ Object properties
- ▶ Object security lists

- ▶ Choice lists
- ▶ Property values
- ▶ Document (content) links
- ▶ Search definitions

Database indexes for custom properties

P8 Content Manager does not support writing to the catalog database through direct SQL commands. Interaction with the catalog database must be handled by using Content Platform Engine Administration tools or through the application programming interface (API).

Important: P8 Content Manager does not support direct writes (updates) to the P8 Content Manager catalog database.

The exception to this rule is custom property indexes. You can create a database index for any class property except system-owned properties. These database indexes, also known as single indexes, are stored within the object store database. For properties that users search frequently, single indexes reduce processing time for queries on this property.

Note: When selecting a property to index, the object store search must be case-sensitive, or the index is not created correctly. You must create additional indexes in Oracle and DB2 to avoid full table scans.

4.9.2 Database stores

P8 Content Manager can be configured to store content inside a relational database. With this configuration, P8 Content Manager converts document content into binary large objects (BLOBs) for storage in the database.

Database storage areas are useful when the size of your object store is not large in terms of the number of documents and the sizes of those documents. Smaller documents of about 10 MB or less have performance advantages in a database storage area when compared to other storage area types. Do not store any document that is over 100 MB in a database storage area.

4.9.3 File stores

With a file store, P8 Content Manager stores content files on a shared network drive or a NAS device. A file store is the most common object store configuration. To organize the files on disk, P8 Content Manager sets up a managed hierarchy of directories on the specified drive.

Note: The file names of the content written to this directory will be based on Global Unique Identifiers (GUID) associated with the document.

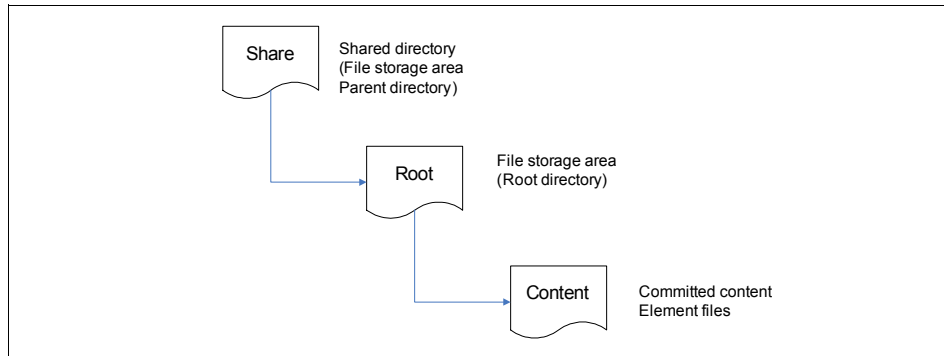


Figure 4-15 File store directory structure

A file storage area consists of a hierarchy of folders on a local or shared network location:

- | | |
|----------------|---|
| Share | The shared folder serves as the parent directory to one or more file storage areas. |
| Root | The root directory of the file storage area is the top-level directory for content storage. A single parent shared folder can contain one or many file storage area root folders. |
| Content | The directory where all committed content element files are stored in a large hierarchy of subfolders. |

During the creation of the storage area, you have the option of creating a small ($23 \times 23 = 529$ directories) or large ($23 \times 23 \times 23 = 12,167$ directories) file storage area. The choice of one or the other is typically determined by the anticipated growth and the need for physically grouping the documents for storage management, backup, or disaster recovery purposes. A large file storage area is more suited for storing a large number of small content elements that contain single-page scanned documents or small emails. A small file storage area is more suited for a smaller number of content elements with a larger average size, such as content element files with embedded images, spreadsheets, and graphics.

Documents are stored among the directories at the leaf level using a hashing algorithm. We suggest that the best practice is to limit the number of content element files in a leaf directory to fewer than 5,000. For a small directory

structure, the upper limit is around 2,500,000 content element files. For a large directory structure, the limit is about 60,000,000 content element files. The number of documents for the file store depends on several factors, such as the type of content being stored, the size of the content, and the type of the file store being used. After that, create multiple storage areas. With larger file stores, the following issues can arise:

- ▶ Larger file stores take more time to perform the weekly full backup. After that, consider differential and synthetic backups to avoid full backups or implement a SAN/NAS-based snapshot and replication.
- ▶ Consistency checker takes a long time to run.
- ▶ P8 Content Manager does not have a hard limit on the number of files in the file store, but the more documents you have, the larger the constraint placed on the file system.

P8 Content Platform Engine administration tools offer the capability to set a limit on the number of content elements and size for a certain file storage area. When either limit is reached, the file storage area is closed and the new content is directed to the next open file storage area in the storage policy. To help manage a large storage space across multiple storage areas, Content Platform Engine farmed or rolling storage areas can be implemented, as indicated earlier.

SAN versus NAS for file storage

Although NAS and SAN are used somewhat interchangeably in this book, there are some operational issues with SAN that typically make it inappropriate to use for a file storage area. Before describing these issues, we provide a better definition of SAN versus NAS as described by the IBM office of the CTO. From an operating system perspective SAN is seen the same as local disk or directly attached storage (DAS).

“Both local SCSI disks and SAN are accessed at the block level, so SAN typically looks like ordinary locally attached SCSI disk to the operating system. Disk I/O is done at the block or sector level. A driver translates those SCSI calls into calls through a host bus adapter over a Fibre Channel network to the SAN device on the other end of the Fibre Channel. The Fibre Channel network is a specialized optical network used just for connecting SAN devices to one or many servers through the host bus adapters in those servers. A host bus adapter is what you called a Fibre Channel card.”

“Network file shares, on the other hand, are accessed at the file level with operating system file system calls. So, network shares look like a local file system, with folders and files, not like local disk, to applications. Underneath the operating system, a network protocol is used to extend local file system calls over a standard LAN to the network file server. The act of binding a

remote network file system into the local file system folder hierarchy is called a mount. The network protocols most commonly used for extending file system calls over the network to the remote file server are CIFS for Windows based clients, and NFS for UNIX based clients. Network shares can be provided by a network file server, or by specialized storage devices called NAS devices. NAS devices plug directly to a LAN and are dedicated to providing network shares to other computers on that LAN.”

- IBM office of the CTO

A SAN Fibre Channel device, a logical unit number (LUN), can be physically connected to multiple server nodes at a time. However, you need to have a combined operating system/file system that supports concurrent access to a shared LUN to be able to share the physical device without using a network file system, such as NFS.

With standard (non-parallel) file systems (that is, 99% of them, such as UNIX file systems (UFS) and journaled file systems (JFS)), the Linux or IBM AIX® operating system is written to control the disk directly, and as efficiently as possible, assuming sole ownership of the device. It can do whatever it wants. Think of a SCSI disk or a directly attached device where there is normally only one SCSI master on a SCSI bus, and that is the computer/disk controller. Therefore, the OS uses cached copies of data structures that are on disk.

Typically, the file system information (inodes, files, and directories) is not necessarily constantly in synch with the state of the physical device as the caching occurs. However, the operating system tracks all the moving parts and ensures that everything stays consistent within the scope of what it controls. This assumes that no other system is attempting to access the same blocks from a different port. This leads to two issues:

- ▶ The OS might make several updates to a structure that is in memory without writing it out to disk, for efficiency purposes, for example, an inode. So, another computer reading the disk is unaware of the latest updates.
- ▶ Similarly, when the OS writes the updates out to disk, it simply writes out the whole block. If another computer has updated that block since it was first read in, the other computer's updates will be overwritten when this computer finally does its write. This problem, in particular, rapidly results in a corruption of the file system that most likely causes both computers to crash.

So, even though Content Platform Engine can handle and control concurrent access at the file level, it cannot control all I/Os going to a certain device. And, it has no access to a level lower than the file system, that is, to the physical disk block level. Content Platform Engine works by using the API provided by the file system. The fundamental problem is whether the file system underlying that API supports concurrent access to the disk. So, this is why Content Platform Engine needs to rely on a network file system, such as NFS, to go beyond the scope of any one machine's operating and file system to handle concurrent access to a single physical device.

Thus, a SAN or network protocols simulating a SAN, such as iSCSI, cannot be used for a file storage area if the Content Platform Engine servers run on different host machines. A SAN cannot control concurrent write access to the same directory if the requests come from different host machines. A SAN can *only* be used as a file storage area if all the Content Platform Engine servers that are writing to it are on the same host machine. In this case, the operating system on the host machine can control the concurrent write access to a common file store directory structure.

4.9.4 About storage policies

A *storage policy* provides mapping to a specific object storage area and is used to specify where content is stored for a specific content class. P8 Content Manager supports the mapping of storage policies to one or more storage objects; therefore, each storage policy can have one or multiple storage areas as its assigned content storage target (see Figure 4-16 on page 132). This concept is known as *farming*.

A storage policy can be used to distribute an I/O load through farming, and it can be used to provide continuous storage availability by pre-provisioning storage areas in a standby state. When a storage area referenced in a storage policy becomes full, a standby storage area, if available, is automatically opened to maintain the same number of open storage areas for the policy.

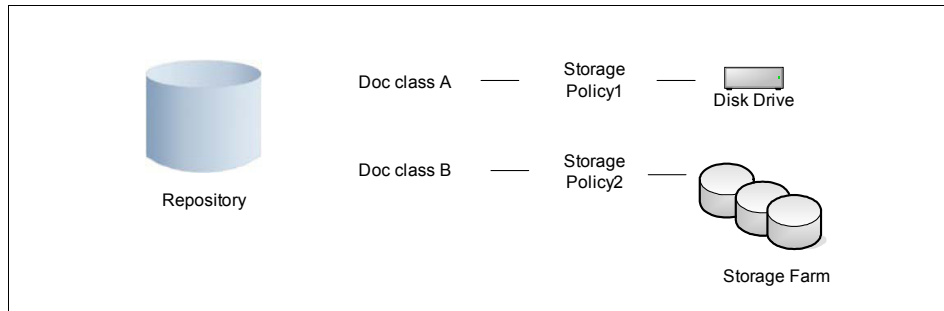


Figure 4-16 Storage policies

Recommendations: Use storage policies with the class definition to save the content.

Farming

A *storage area farm* is a group of storage areas that acts as a single logical target for content storage. Storage area farms increase the throughput by distributing the I/O load across all the open storage areas. With storage area activation, the Content Platform Engine maintains the same number of open standby storage areas for the policy, therefore, better distributing the I/O load across the standby storage areas. With farming, Content Platform Engine provides load-balancing capabilities for content storage by transparently spreading the content elements across multiple storage areas. Therefore, the storage policy functions as both the mechanism for defining the membership of a storage area farm and also the means for assigning documents to that farm.

Create separate file storage areas to ensure efficient document management. For example, you can create a file storage area to group documents with the same deletion or backup requirements. Map storage areas with documents by modifying the storage policy property on document classes.

Recommendations: Use Content Platform Engine administration tools to configure storage policies and storage area farms.

4.9.5 Using fixed storage devices

Fixed storage devices are large capacity third-party storage devices that feature hardware-level content protection. Examples of fixed storage devices are EMC Centera or NetApp Snaplock. Fixed content systems potentially provide extremely large storage capacity, as well as write-once hard drive technology.

Fixed content stores compared to file stores

Before deciding on a fixed content store, review the following considerations:

- ▶ Content stored in the fixed storage area is accessed via the Content Platform Engine using a third-party API rather than the file system API.
- ▶ Read/write access to the repository can be slower in a fixed content store than access to Content Platform Engine's file storage area.
- ▶ For fixed content store, the repository might be write-once, which does not allow any changes to the content. This is exactly the same as normal file storage areas in that the document content can never be changed after it is added to the repository. The document content can only be revised, and new versions can be added to the repository.
- ▶ The repository might not allow the deletion of content except through third-party device tools. The repository can support a retention period for content, which means that the deletion of the content is not allowed until the retention period has expired.
- ▶ A fixed storage area requires a small file storage area to be used as a staging area before it is stored into a fixed device.

The fixed content system can limit the number of concurrent connections to the server, which means that there are fewer connections allowed than current read/write requests normally supported by the Content Platform Engine. This might result in decreased performance, but not error conditions.

For more information about content storage management and storage farming, see this developerWorks article:

<http://www.ibm.com/developerworks/data/library/techarticle/dm-1003filestetstoragemanagement/index.html>

4.10 Considerations for multiple object stores

There are several valid use cases for deploying multiple object stores. A single object store can handle a catalog containing over a billion objects. Using multiple storage policies, there is a virtually unlimited amount of storage.

Except under extreme conditions, size is not a factor in the decision to add additional object stores.

Multiple object stores are warranted in the following situations:

- ▶ An object store is subject to high ingestion rates or frequent update procedures and needs to be segregated for performance reasons.
- ▶ Content must be separated for security reasons.
- ▶ User groups are separated by a large geographic distance.

For performance reasons

If an object store will be the target of high-volume ingestion rates, such as those produced by Capture, Datacap, ICC, or Email Manager, it makes sense to separate that object store from others that are dedicated to document lifecycle use. Users who search for and check out documents for editing will experience better performance if the object store they use is not busy handling high-volume automated processes. There are two common examples of this situation where multiple object stores are used: email archiving and IBM FileNet Records Manager solutions (see Figure 4-17 on page 135).

The IBM Records Manager object store that hosts record information is subject to processing intensive database activity during retention and disposition processing. In addition, record objects are small and best suited for database stores. For these reasons, records need to be stored in a separate object store.

Recommendations: Set up a separate, database object store for IBM FileNet Records Manager. This object store is commonly called the file plan object store (FPOS).

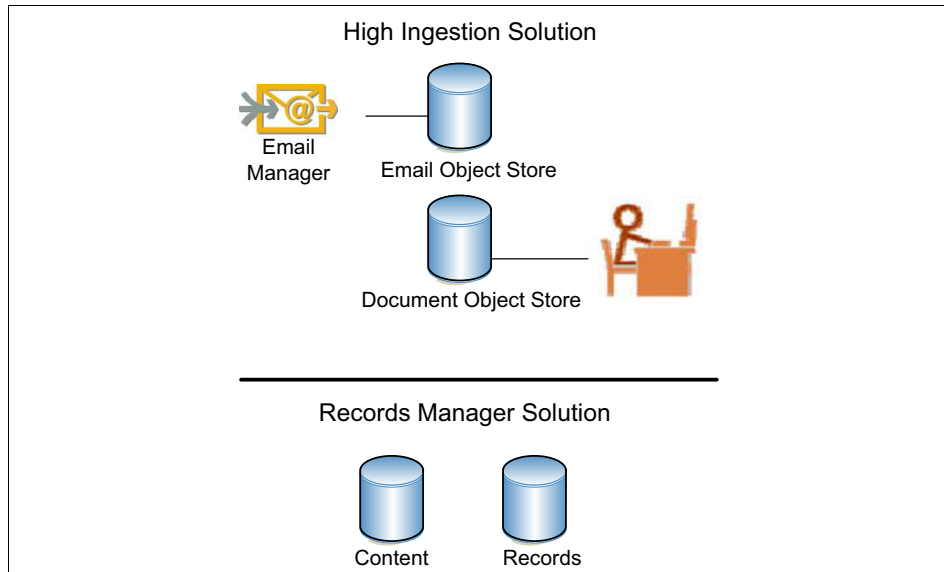


Figure 4-17 Two solutions with multiple object stores

Use the Data Source sharing feature. For more information, see the “Sharing Data Sources” and “Creating a Database Connection” topics in Administering Content Platform Engine:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.admin.tasks.doc%2Fp8pcb027.htm>

For security reasons

Another reason to implement multiple object stores is a requirement to strictly separate content for security reasons. Although it is possible to keep classified content secure by using marking sets and security policies, certain content must be kept absolutely separate. In these situations, install a second object store for classified content.

Here are a few situations where secure object stores are a solution:

- ▶ Board of director-level content
- ▶ Secret or top secret government content
- ▶ Public-facing Internet-accessible libraries
- ▶ Service companies that offer enterprise content management services to multiple customers

By geography

Many organizations have large offices in several countries. Wide area network (WAN) links are expensive over large distances and typically have low bandwidth and high latency. It is not always practical for offices in this situation to share the same P8 Content Manager system. One solution to this situation is two separate repositories managed by two separate P8 Content Manager systems as shown in Figure 4-18.

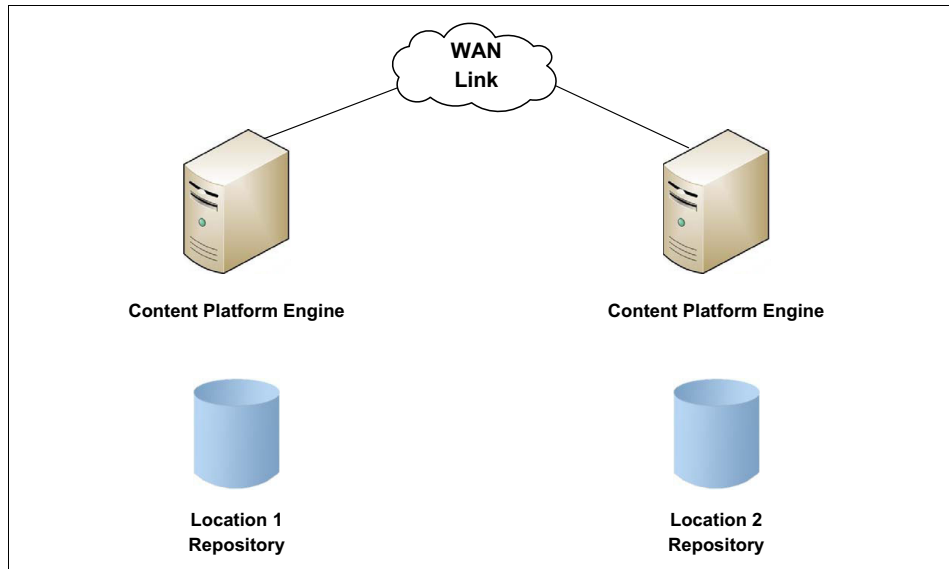


Figure 4-18 Two separate repositories

In this solution, an organization has installed two separate P8 Content Manager systems in two distant offices, usually under a single p8 domain. Users in each office have high-speed access to the local repository for document retrieval and editing. Users in remote offices can still search and retrieve content in the remote office repository, but because this activity is less frequent than local access, traffic over the WAN link is reduced.

By functional group

In the organizations with several functional units, each functional unit, such as Human Resources (HR), Legal, and Marketing, might want to have a separate object store. This separation of data offers the flexibility for the users to control the documents and implement the security and access rights for the users in the organization. This separation also allows the line of business (LOB) applications to use only the data pertaining to that business unit.

By the size of the object store

Content Platform Engine object stores can handle data from tens of millions to hundreds of millions of objects. The number of objects that an object store can handle depends on several factors, including the database used, storage areas, rate of ingestion of objects, the metadata defined, and the kinds and size of data stored. After the object store reaches its maximum capacity on any of the factors, it is advisable to create a new object store.

4.11 Retention management and automatic disposal

P8 Content Manager can be configured to set the retention on the instances of the annotations, custom objects, documents, and folders. The retention date on the object prevents the deletion of the object until the retention date is passed. Organizations might have to keep the documents for a certain period of time for legal, regulatory, and contractual reasons. Setting a retention value on those objects helps ensure that the objects are retained in the system until all the obligations are met. Also, having a retention date on the objects removes the possibility of the accidental deletion of objects until the retention date has passed.

Automatic disposal lets you run the disposition task to remove all the objects in the Content Manager based on the criteria specified. The following sections describe retention management, automatic disposal, and best practices.

4.11.1 Retention management

To prevent an object from being deleted, you can specify a retention setting for the object.

A retention setting can be applied either statically or based on events. Static retention for an object is set once. After a fixed amount of time, the static retention requirement is satisfied and the object can be deleted. Event-based retention is dynamic, allowing you to change the retention setting for an object after the occurrence of a business event. Event-based retention closely aligns the retention policy of the data with the business requirements for that data.

The default retention can be set at the class level. Instance retention can be set at the object level in the Content Manager. Events, such as Content Platform Engine events, can be used to set and alter the retention.

Class-level retention

The retention period you specify at the class level is applied to newly created object instances and to document instances when the document is checked in. Class-level retention can be set to the annotation, custom object, document, and folder classes. Users need to have the modify retention permissions at the object-store level to set the class-level retention.

Object-level retention

The objects of annotation, custom object, document, and folder can be set with the retention date at the time of object creation. By default, objects inherit the retention value from the parent class. To set or change the retention date for an object, users need a special modify retention permission. These special permissions are not required if the retention is defaulting from the class.

For static retention, users have to specify the retention date at the time of creating the object or the exact value has to be specified in the class as the retention date. Event-based retention is a scheme where you first set the retention to `Indefinite`, which means it cannot be deleted, but the expiration date is not defined. When a business event occurs, you initiate or trigger the retention by changing the retention value from `Indefinite` to a specific date. Another special value, `Permanent`, never lets the document delete from the object store. The retention value can only be set to a greater value than the current value. Content Platform Engine does not allow you to reduce the retention period.

Retention modes

There are two retention modes that are supported by storage areas in Content Platform Engine: aligned and unaligned. Only annotations and documents can have the content in Content Manager. In aligned mode, the retention value is reflected on the content stored on the fixed content device so that users cannot directly delete the content from the fixed store and from the Content Platform Engine. In unaligned mode, all the retention features are supported by the Content Platform Engine, and the content in the fixed store is not stored under retention. In unaligned mode, users can delete the content directly on the fixed store, because the enforcement of retention happens in Content Manager, not by the fixed store.

Note: Use aligned mode when you want strict enforcement of retention on the content object in the document. Use unaligned when all you need is enforcement by Content Platform Engine.

4.11.2 Automatic disposition

Automatic disposition is a process that runs in the background and deletes objects automatically after their retention dates have expired. Disposition uses the sweep framework in Content Manager. Many enterprise content management applications involve searching among a large set of objects for some subset that meets specific criteria, and then performing an operation on those objects that match the criteria. Content Manager calls that type of procedure a *sweep*.

The sweep framework is a policy-based framework. A sweep job is like a sweep policy, except it only executes a single sweep. A sweep policy executes repeated sweeps. A disposal policy has a target class that defines the set of objects that will be examined, a filter expression that defines the subset of the objects that will be disposed of, and a schedule that determines the time of day and the days of the week it will execute.

Automatic disposition is declared by using the disposition policy. The disposition policy requires a class, filter, and schedule to run. A disposal policy can run in one of three possible modes. *Normal* mode where the disposition action is taken on the resulting objects. *Preview* mode shows the preview of the objects where this disposition policy is going to execute. *Preview only counters* mode shows the count of target objects for this disposition policy. The disposition policy runs as it is scheduled.

Note: Initially, set the sweep to preview or preview only counters mode to determine the objects that are going to be disposed of. Use the FileNet System Monitor to monitor the sweep activity and sweep rate.

Note: A policy-based sweep combines all policies, including the retention update policy and disposal policy for a certain base class, into a single sweep.

4.11.3 Retention update

The retention period for instances of the four classes that support retention can be altered by using either a retention update policy or a retention update job. A retention policy is used for recurring retention updates, for example, changing retention from *Indefinite* to five years from the creation date on instances that have been in the system for more than 90 days. A retention update job is used for one-time only operations. For example, extend the retention date by one year for instances of a certain class.

Retention update job

Retention update jobs allow the retention period to be altered on retainable objects based on the class and property state of a candidate object. On a retention update job, the new retention date can be explicitly specified. Or, it can be computed by specifying the name of a date property on the Sweep Target Class, an offset, and the time units in which the offset is expressed.

Retention reduction job

Normally, a retention update job is used to extend retention on instances. In some limited use cases, a *retention reduction allow* job can be used to reduce retention. An example is to handle a change to a regulation. A retention job supports two modes of operation: retention reduction allow and retention reduction prevent. The default mode of operation is retention reduction prevent mode.

Retention update policy

Retention update policy is a policy-based implementation of retention update that allows the retention period to be altered on retainable objects based on the class and property state of a candidate object. Retention update policy can be specified with the specific retention date or the new retention date can be computed from the base retention date, offset, and offset units.

Note: You can create a covering index for better performance of policy-based and job sweeps on the base table being swept. The covering index needs to include all the columns that are included in the selection list and define the `object_id` as a unique value.

The selection list for sweep is in the P8 database trace logs with `Sweep.SQL`. A sweep that includes the overflow table has to include a covering index for the overflow table.

4.12 P8 Content Manager searches

There are several methods of searching for content in the P8 Content Manager repository. The methods can be divided by the purpose of the search:

- ▶ User-invoked searches
- ▶ Content-based searches
- ▶ Repository maintenance searches

P8 Content Manager offers a set of tools for each purpose.

4.12.1 User-invoked searches

Users can create and invoke P8 Content Manager searches through FileNet Workplace XT and IBM Content Navigator. FileNet Workplace XT and Content Navigator are web applications. FileNet Workplace XT and Content Navigator offer two types of searches: search and stored searches.

Search

Search can be customized in FileNet Workplace XT and Content Navigator by individual users. Search appears when users log in to P8 Content Manager. Using FileNet Workplace XT or Content Navigator search is an ideal tool for user-invoked ad hoc searches for repository content. Users can search any properties and can add any system or custom property to the criteria display.

Note: When users modify their search criteria, the system remembers the settings and will display them again on the next visit to the site.

Stored searches

FileNet Workplace XT and Content Navigator offer a tool for designing search templates for more sophisticated content searches. Search Designer offers the following enhanced features:

- ▶ Cross-object store searches
- ▶ Search criteria expressions (AND/OR options)
- ▶ Preset criteria for filtering search results
- ▶ Searches that appear as links on a browser favorites or bookmarks menu

Use Search Designer to create stored searches and cross-repository search. Stored searches can also be accessed as web links, which makes it easy to add the stored searches as favorites to the browser.

4.12.2 Content-based search

P8 Content Manager supports content-based retrieval (CBR) for documents, annotations, folders, custom objects, and their properties. With CBR, you can search an object store for objects that contain specific words or phrases embedded in document or annotation content. With CBR, you can also search an object store for objects that contain specific words or phrases embedded in string properties of objects that have been configured for full text indexing.

The Content Platform Engine uses the IBM Content Search Services (CSS) server for indexing and searching the documents. Content-based searches can be performed from all P8 Content Platform Engine client search tools. Content Platform Engine has the capability to fail over during indexing and search, and has supporting configurations with no single point of failure.

Indexing process

The indexing process begins at the Content Platform Engine when CBR-enabled objects, such as documents, are created or updated. The Content Platform Engine stores indexing data for the CBR-enabled objects in the indexes created and managed by the CSS servers. Each index is associated with a distinct index area in the object store. During an indexing process, the system can write to multiple indexes across the index areas. When an index's capacity is reached, the index is automatically closed and a new index is created.

The Content Platform Engine queries the items from the index request table to identify documents that are queued for indexing and then groups index requests pertaining to the same target index into an index batch. The binary documents in this batch are converted to text by the text extraction processes, then the entire batch is submitted to a CSS server for indexing.

Text extraction happens in an external process running outside the Content Platform Engine. Text extraction runs on the Content Platform Engine server by default. All the Content Platform Engine servers in a site can dispatch the requests for indexing. This allows the Content Platform Engine to share the text extraction load among all the available servers because the text extraction processes can be CPU-intensive and disk I/O-intensive. Text extraction throughput can be configured by using the Content Platform Engine administrative tools. The text extraction processes are also known as *text filters*. The number of text filters for Content Platform Engine can be increased or decreased based on the CPU utilization and the available system memory of the server. During text extraction, the text filter process writes the intermediate temporary data to the text filter's temporary directory. Having this temporary directory on a fast I/O device will increase the performance of the text filters.

Note: When IBM Content Collector is used with the Content Platform Engine, text extraction for the binary documents occurs on the CSS server via the IBM Content Collector plug-in.

After the batch of documents is processed by the text extraction processes, the text file batch is submitted to the CSS server. After the CSS index server receives the index batch, the preprocessing functions begin.

Preprocessing functions consist of these tasks:

- ▶ Document construction
- ▶ Language identification

Ensure accurate processing and optimal performance by specifying a default language for an object store. If one language cannot be definitively identified for the content of an object store, set it to the list of languages that is contained by the object store's documents.

- ▶ Tokenization

Tokenization creates the tokens from the extracted text. The language of the document plays a key role in identifying the tokens. After the tokens are created from the document, the index for the document is updated or created with these tokens in its respective index area.

The full-text indexes in the Content Platform Engine have a stickiness to the IBM Content search servers for the purposes of indexing. This stickiness is never changed until the lease expiry time has been reached. The lease expiry time is the time since the last server performed indexing on the full-text index has reached a time threshold or the content search server for which it is sticky became unavailable. The Content Platform Engine server changes the stickiness of an index to the least loaded index server to load balance between the available index servers.

Content Platform Engine can optionally be configured to group the CSS servers and index areas accessing the common shared index area root directory into a group called an *affinity group*. Each index area can be attached with an affinity group. The CSS server can optionally be assigned to a single affinity group. If all the servers in the affinity group are on the same machine accessing the local index area root directory, any failure to the machine causes all the servers to go down and the high availability for these indexes might be at risk. The load balancing for indexing happens between the servers in the affinity group. Also, the load balancing and failover occur between the CSS servers that are not associated with any affinity group. For indexing purposes, it is important to assign multiple servers for the affinity group. The Content Platform Engine servers perform the indexing load balancing based on the active index servers and the workload on each server.

Important: Content Platform Engine load balances and fails over within the affinity group and among the indexes that are not part of any affinity group. Consider the high availability feature when using affinity groups.

Content Platform Engine performs the index partitioning based on the partitioning property value configured on the object store. *Index partitioning* is the way to group the indexing information for the objects whose partitioning property value is the same. Indexing partitioning improves the performance of CBR searches when the search criteria contains the partitioning property by including only indexes that match the partitioning values specified. Without partitioning, performance might be worse because the Content Platform Engine has to search a larger number of collections. For a CBR search to work on the partitions, consider the searches that are going to be run when you are creating a partitioning property.

Important: Specify the complete list of languages used within the object store's content. Identifying the correct language for a document improves the tokenization and search.

Use CSS in a dedicated mode, such as index or search. Avoid using it in the dual mode of index and search. By default, each CSS server is configured to use four CPUs for indexing. Consider this configuration when deciding how many CSS server instances to create on the system.

Search process

Users can submit CBR queries against the full-text index with its criteria: CBR-enabled properties or terms that exist in the content. Search requests are initiated through the Content Platform Engine administration tools or other client applications using the Content Engine API and include a full-text expression that is submitted to the CSS search server. The content-based search expression is highlighted in the following query:

```
SELECT d.This FROM Document d INNER JOIN ContentSearch c ON d.This =  
c.QueriedObject WHERE CONTAINS(d.*, 'lion AND tiger')
```

The search server uses word stems, synonyms, and stop words to improve search efficiency and accuracy. It searches for and identifies the stem for all word terms included in a full-text search expression. A *stop word* is a word or phrase that is ignored by the search server to avoid irrelevant search results caused by common expressions. The search server uses these definitions on the index and runs the full-text search. The results are returned to the Content Platform Engine server, which then joins the results with other tables in the query and runs the query. The stop words do not affect the indexing, and they appear in the indexes created by the CSS server.

The Content Platform Engine server runs the searches concurrently. The search server configuration on the domain allows full-text indexes to be searched in parallel to satisfy user queries. With content-based searches, you can search the content based on the words and phrases, string properties of a CBR-enabled object, partitioned properties, and also by using the XML and XPath queries.

Recommendations: If query criteria includes partitionable properties, consider using index partitions to reduce the number of indexes to search by increasing the speed of the search. Index partitioning increases the number of indexes created. If there are no partitionable properties on the query criteria, we advise that you not use the partition. Search with order by rank reduces the performance, so use this search only when required. Ranking is determined by the CSS server.

Content-based searches tend to be slower when searches are run concurrently with indexing. Dedicate servers for content search because I/O and memory are the important factors in the search operation. We advise that you have up to 6 GB of memory for each CSS server. Content-based searches on property will perform better than the content and XML searches.

Index areas

An *index area* is a file system directory that contains CSS indexes. Each object store can have multiple index areas and each index area can have multiple indexes. The index contains the indexing information for the objects that belong to the same indexable base class or subclasses of the base class. Index areas can have different states: OPEN, CLOSED, STANDBY, or FULL. Unrelated to the status of the index area, all the indexes in the index areas might be searched.

Important: Index area root directories need to be unique among index areas even if the root directory path is on the local disk.

An *affinity group* is a group of CSS index servers and index areas. The servers in a group access only those index areas in the same group. The servers that are not in a group access only those index areas that are not in a group. Although the configuration of affinity groups is optional, it is a good practice to have multiple CSS servers assigned to an affinity group and to have the root directory local to the CSS servers indexing. All the servers in the affinity group must have read/write access to the root directory.

Configure the number of index areas to less than or equal to the number of CSS servers in the indexing mode. During content ingestion, you might want to have an equal number of indexing servers and index areas to keep all the indexing servers busy. You might want to consider having additional indexing servers for failover purposes.

Important: The location specified for the index areas needs to be accessible for read and write for all the CSS servers. If an index area is assigned to an affinity group, it needs to be accessible to all the servers in the affinity group.

4.12.3 Searches for repository maintenance

Content Platform Engine Administration tools feature a query tool that can be used for detailed report generation or for maintaining an object store repository. With the Query Builder tool, you can create a search query and apply bulk actions on the objects returned in the result set. With Query Builder, you can perform these functions:

- ▶ Find objects using property values as search criteria.
- ▶ Create, save, and run simple searches.
- ▶ Create and save search templates that will prompt for criteria when launched.
- ▶ Launch search templates that are provided with each Content Platform Engine and Content Platform Engine Administration tools installation. These templates are provided to assist with managing the size of your audit log and for managing entries in the QueueItem table.
- ▶ Create, save, and run SQL queries.
- ▶ Searches can be combined with bulk operations that include the following actions (available on the Query Builder Actions tab):
 - Delete objects.
 - Add objects to the export manifest.
 - Undo checkout (for documents).
 - Containment actions (for documents, custom objects, and folders): file in folder and unfile from folder.
 - Run VBScripts or JScripts (Query Builder Script tab).
 - Edit security by adding or removing users and groups.
 - Lifecycle actions: set exception, clear exception, promote, demote, and reset.

In Query builder, there are two ways to construct searches: Simple View and SQL View. Select **view** from the toolbar to select a view style:

- ▶ Simple View offers a point-and-click interface where you can select tables, classes, and criteria from drop-down lists.
- ▶ SQL View translates anything that you create in Simple View. This is a one-way translation only; you cannot translate an SQL View into a Simple View. SQL View presents the query in an SQL text window that you can then directly edit or load any *.qry files that you have saved on the network.

Both views construct a query that can be bundled with the other Query Builder features: bulk operations, scripts, and security changes. Both views support Search Mode and Template Designer Mode.

Tip: To aid administrators using SQL View, the P8 Content Manager help files contain P8 Content Manager database view schema.

Search templates and template designer mode

Search templates are like simple queries except when search templates are loaded from the Content Platform Engine Administration tools Saved Searches node. Then, they prompt you for search criteria and whether you want to include any defined bulk operations.

IBM FileNet provided search templates are installed with every Content Platform Engine or Content Platform Engine Administration tools-only installation into a folder on the local server named SearchTemplates. This folder is in the FileNet installation directory. Any queries placed in this folder appear in the Content Platform Engine Administration tools Saved Searches node as long as they have .sch as a file name extension.

Querying object-valued properties

One of Content Platform Engine's powerful search features is the ability to retrieve an object when provided another object that is a member of one of its object-valued properties. For example, you can find a document that has a particular security policy by using the identifying ID of the security policy in the search criteria.

Multiselect operations

Multiselect (or bulk) operations perform an operation on all objects returned in the search results from the query builder query. This feature is useful for object store maintenance activities.

With multiselect operations, you can perform the following actions on multiple files at the same time:

- ▶ Delete
- ▶ File to folder
- ▶ Unfile from folder
- ▶ Undo checkouts
- ▶ Change lifecycle states
- ▶ Add to security ACLs (you cannot delete existing entries)
- ▶ Run an event action script

For example, assume that several documents had been checked out by someone who left your company. Using multiselect operations, you can search for all documents that were left checked out by that person and undo these checkouts in one operation. To do this, you use the Query Builder to construct a search to find all documents currently checked out under the former employee's system login name.

4.12.4 CBR query optimization

CBR query optimization specifies how searches that contain both a content-based retrieval (CBR) search and a relational search on a database are executed. By default, the Content Platform Engine always performs the CBR search first and the database search second. The CBR-first approach is most efficient when there are few full-text hits. Efficiency decreases, however, when there are many full-text hits, and there are fewer database hits than full-text hits.

To provide control over how combined searches are executed, the `CBRQueryOptimization` property can be set on the object store. As an alternative to the default CBR-first option, you can set the property to the dynamic switching option. In dynamic switching mode, the Content Platform Engine dynamically determines whether to issue the CBR search first or the database search first, optimizing performance for these types of searches.

In dynamic switching mode, the Content Platform Engine switches from CBR first to database first based on an estimated number of CBR hits. The estimate is compared to a threshold value, set in the `CBRQueryDynamicThreshold` property. If the number of full-text estimated hits is less than or equal to the `CBRQueryDynamicThreshold` value, the CBR search is executed first (CBR-first search). If the number of full-text estimated hits is larger than the `CBRQueryDynamicThreshold` value, the database search is executed first. The dynamic switching operation is affected by various search options, including requests for rank ordering. The `CBRQueryRankOverride` property on the object store determines how the server responds to CBR search requests for rank order and can affect server performance.

Best Practice: To ensure that database-first searches execute efficiently, set database criteria on indexed properties. The database-first searches require database indexes on at least one property in the WHERE clause for good performance. Otherwise, queries perform inefficiently during the database-only portion of the search.

Specify limiting conditions on database criteria to achieve relatively small hit counts. Database-first searches are most effective when the database hit count is relatively small.

4.13 Conclusion

In this chapter, you learned about the basic concepts and elements that comprise a repository and repository design. While designing the system, ensure that you have someone to look after the design of the repository. Use the prefix for the symbolic names in the repository that uniquely identifies your solution and does not interfere with Content Platform Engine symbolic names and naming conventions. Create a meta object store and import the metadata from the meta object store to other object stores. Ensure that you create a prototype to validate your design before implementing it. Consider the best practices and performance considerations before finalizing the design.



Security

This chapter describes the security mechanisms provided by the Content Platform Engine to secure the resources under its management against unauthorized access and to ensure that authorized users are given only sufficient access to carry out the tasks assigned to them, referred to as *access control*. In addition, it provides a series of recommendations for how best to employ those mechanisms to achieve the desired access control goals.

We discuss the following topics in the chapter:

- ▶ Access control
- ▶ Authentication
- ▶ Authorization
- ▶ Security best practices

5.1 Access control

Access control can be thought of as being the answer to the question of *who* is allowed to do *what*. There are two parts to that question, the *who* and the *what*, and correspondingly two phases of the overall access control process. The *who* part concerns identifying the entity, usually a person but also potentially an autonomous process, that is making a request of the Content Platform Engine. This is the authentication phase. The *what* part concerns deciding what level of access that entity needs to be given to the content objects targeted by the request. This is the *authorization* phase.

The Content Platform Engine supports standard mechanisms for authentication and a rich set of *authorization* features, as described in the remaining sections of this chapter.

5.2 Authentication

All requests submitted to the Content Platform Engine are subject to authentication, meaning that they must carry within them a verifiable (authentic) identity of the entity making the request. The Content Platform Engine itself does not define or implement authentication; rather, it delegates that to a standard service of the application server environment in which it executes - the Java Authentication and Authorization Service (JAAS).

The outcome of JAAS authentication is required to be an identity that the Content Platform Engine can resolve to a user in a directory service containing configured users and groups, from which a security context can be built and delivered into the authorization phase of access control. These elements of authentication are described in detail.

5.2.1 Use of JAAS

The traditional authentication model of a user providing a user ID and password is tried-and-true and has been in widespread use for decades. It is easy to implement and conceptually simple, but it does have some drawbacks:

- ▶ Software systems built to rigidly expect user ID and password credentials are difficult to adapt in the face of other forms of credentials. Examples of other forms of credentials, which can be used with or without a password, are fingerprint scans and hardware security tokens. It is not possible for a software system built today to anticipate all of the forms of credentials that might be used in the future.

- ▶ In an environment where users must interact with several applications, either the user must repeatedly enter credentials when crossing application boundaries, or the credentials must be passed from one application to another. The first choice represents a usability annoyance, and the second choice represents an information security hazard, because it gives more opportunities for the credentials to be discovered or exploited by an attacker.

For the first of these problems, the software industry has evolved to a model of *pluggable authentication*. Components for verifying different credential types can be developed independently of the framework into which they fit. The output of a pluggable authentication framework is often a token affirming that valid credentials were presented and verified. That is typically enough information for most authentication consumers, although some systems also provide information about the types of credentials that were presented.

Pluggable authentication also works toward solving the second problem, because the token produced can be more securely passed between applications than the raw credentials. There are more factors involved in *single sign-on* (SSO) solutions than pluggable authentication. There must be additional conventions or APIs for the applications to communicate with each other or at least with the SSO framework. A full discussion of SSO frameworks is beyond the scope of this book. Most application server vendors provide at least some SSO capabilities, and there are many vendor solutions available. For example, see *Single Sign-on Solutions for IBM FileNet P8 Using IBM Tivoli and WebSphere Security Technology*, SG24-7675.

In the Java environment, the pluggable authentication framework is JAAS. The Content Platform Engine fully delegates authentication to JAAS (but does not use JAAS for authorization purposes). Virtually all modern SSO solutions also work in concert with JAAS, so the Content Platform Engine server will almost always automatically participate in any SSO solution that might be employed.

The net outcome of JAAS authentication is that the Content Platform Engine server receives a Java artifact called a *Subject*, containing unimpeachable information concerning the authenticated user. The Content Platform Engine requires that this Subject contain identifying information for a user in a configured directory service.

5.2.2 Directory service users and groups

It is a requirement that organizations using P8 define the users of its systems in a directory service. P8 supports the following directory service types:

- ▶ Microsoft Active Directory
- ▶ Microsoft Active Directory Lightweight Directory Services (AD LDS)

- ▶ Novell eDirectory
- ▶ Oracle Internet Directory
- ▶ Oracle Directory Server Enterprise Edition (formerly known as Sun Java System Directory Server)
- ▶ CA Directory
- ▶ IBM Tivoli Directory Server

In addition, support is provided for IBM WebSphere Virtual Member Manager (VMM), which is a directory service *aggregator*, capable of presenting a collection of directory services of heterogeneous types as a single unified virtual directory.

Alongside users, the directory service can (and usually will) define a number of groups. A *group* is a container of users and possibly of other groups, although not all directory services support nesting of groups. The contained users and groups are said to be *members* of the containing group. A group provides a convenient way to grant or deny access to the group members in a way that adapts automatically to changes to membership in the group.

Users and groups are referred to collectively as *security principals* and the authorization mechanisms described next are expressed exclusively in terms of security principals, mostly without regard for whether a particular principal is a user or group.

For each security principal, the directory service is required to provide an immutable identifier, which unambiguously resolves to that principal. This identifier is unique within that directory service and can be used to retrieve the directory service object having that ID. The form this unique identifier takes varies according to the directory service type and on configuration choices made by the owning organization. No matter what form it has in the underlying directory service, the Content Platform Engine transforms it to a universal format when it is presented in the API. This universal format is referred to as a security identifier (SID). SIDs are what the Content Platform Engine stores in its internal authorization data structures.

A security principal also has a *distinguished name*, a *principal name*, and a *short name*. The distinguished name (DN) is unique, and the principal name and short name can be also (particularly the former).

In addition to SIDs obtained from directory service principals, the Content Platform Engine acknowledges two special SID values, which have particular purposes described later. These SIDs have an associated predefined display name and do not resolve to a directory service object. They must be treated specially by API consumers, including administration tools.

There are two special SID values:

▶ #AUTHENTICATED-USERS

This SID (often abbreviated as #A-U) notionally identifies a group, to which all valid users belong (but for which there is no concrete representation). #A-U is automatically added to the security context for any authenticated user. #A-U can appear as the grantee of an access control entry (ACE) in order to grant or deny access to all users (that is, to everyone).

▶ #CREATOR-OWNER

This SID (#C-O), which is notionally of a user, plays no role in access checking (it does not appear in the security context). Its purpose is to act as an alias for the creator or current owner of an object during the application of default security, inheritable ACEs, and security templates. It is substituted by the actual creator or owner SID.

5.2.3 Security context

The security context (also sometimes referred to as the *user access token*) provides the identity information upon which authorization checks are based. It contains the SIDs of every security principal via which a calling user can be granted or denied access:

- ▶ The SID of the authenticated requesting user
- ▶ The SIDs of any groups of which the user is an immediate member
- ▶ The SIDs of any groups within which those groups are nested, and so on (an exhaustive flattening of group nesting)
- ▶ The SID for #AUTHENTICATED-USERS

The SID of the requesting user is determined by extracting the login name from the incoming JAAS Subject and searching the directory service to locate the corresponding user object. The group SID list (apart from #A-U) is obtained by further searches of the directory service.

Searching the directory service is a relatively expensive operation, so to avoid repeating it for each incoming request, the Content Platform Engine maintains a cache of fully constructed security contexts keyed by login name. Management of this cache, called the *user token cache*, is described in “User token cache” on page 189.

5.3 Authorization

Authorization is the second phase of access control, and is the phase in which a determination is made of the operations the caller is permitted to carry out on a particular object.

5.3.1 Access rights

Authorization determines a set of *effective access rights* that a caller is granted for a particular object. These in turn determine what operations can be performed on the object. Access rights are also the way grants and denials of access to a particular principal are expressed.

Concretely, access rights are represented by bit values, which are combined into an *access mask* - a 32-bit value in which distinct bit positions represent different access rights.

Some access rights have nearly universal meaning for all objects. For example, READ always implies permission to read an object and DELETE always implies permission to delete it. Others have meaning only for particular types of objects. Access rights that do not have meaning for a particular type of object can nevertheless appear in its effective access mask (this can occur for a variety of reasons) but play no role in determining what operations are permitted.

Table 5-1 gives a list of the access right names, a description of their meaning, and the types of objects to which they are applicable.

Table 5-1 *Access rights*

| Right | Applies to | Description |
|---------------|--------------------------------|--|
| READ | All types | Confers permission to retrieve the object. |
| WRITE | All types | Allows modification, and also (where relevant) the API Lock, Unlock, ChangeClass, and MoveContent actions. |
| LINK | Folder, Document, a few others | Grants permission to file into a folder or to annotate a document, and other similar things. |
| UNLINK | Folder | Grants permission to unfile from the folder by deleting a referential containment relationship. |
| MINOR_VERSION | Document | Allows checking out and checking in as a minor version. |

| Right | Applies to | Description |
|-----------------|--------------------------------------|---|
| MAJOR_VERSION | Document | Allows checking out and checking in as or promoting to be a major version. |
| CREATE_INSTANCE | Class Definition | Permission to create an instance of the defined class, specify the class as the target of a ChangeClass API action, and for custom event classes, allows the class to be used in a RaiseEvent API action. |
| CREATE_CHILD | Domain, Folder, and Class Definition | Allows creation of new domain level (global configuration database (GCD)) objects, or subfolders of a folder, or subclasses of a class definition. |
| CHANGE_STATE | Document, Task | Allows document lifecycle methods and task state changes. |
| PUBLISH | Document | Permits the document to be submitted for publication. |
| DELETE | All types | Allows the object to be deleted or marked for deletion. |
| READ_ACL | All types | Permits read access to the Permissions list. |
| WRITE_ACL | All types | Allows modifications to the Permissions list and the API Freeze, ApplySecurityTemplate, and TakeFederatedOwnership actions. |
| WRITE_OWNER | All types | Allows ownership to be taken of the object. |
| CONNECT | Object Store | Allows access to objects in the object store for read. |
| STORE_OBJECTS | Object Store | Permits new objects to be created in the object store. |
| MODIFY_OBJECTS | Object Store | Allows objects in the object store to be modified. |
| REMOVE_OBJECTS | Object Store | Allows objects in the object store to be deleted. |
| WRITE_ANY_OWNER | Object Store | Permits any object in the object store to be retrieved and the Owner property to be modified. |

| Right | Applies to | Description |
|--------------------------|--------------|--|
| ADD_MARKING | Marking | Allows the marking value to be applied to objects. |
| REMOVE_MARKING | Marking | Allows the marking value to be removed from objects. |
| USE_MARKING | Marking | Permits objects having the marking value to be accessed without constraint of the marking. |
| PRIVILEGED_WRITE | Object Store | Allows modification of certain properties that are normally treated as read-only for any object in the object store. |
| MODIFY_RETENTION | Object Store | Permits the retention date to be modified for objects in the object store. |
| VIEW_RECOVERABLE_OBJECTS | Object Store | Allows reading of objects that have been marked for deletion and thus rendered invisible to general users. |

5.3.2 Security descriptor

The *security descriptor* (SD) is the primary mechanism through which access to an object is controlled, although it can be augmented by other means. The majority of independently persistable objects have a security descriptor, although some do not and so are subject to a different access check, as described later.

The SD is an internal construct maintained by the server and not exposed directly in the API. Instead, the two elements of the SD - the Owner and the Permissions list - are exposed as independently editable properties, combined in storage as the SD.

For objects stored in an object store (repository objects), the server implements a single-instance mechanism for SDs, so that if two objects have the same owner and permissions and therefore identical SDs, only one copy of the SD is stored, referenced by both objects. This, along with other details of how SDs are managed, makes them highly amenable to caching. The server maintains a *security descriptor cache*, which ensures that it is frequently possible to evaluate access to an object without a separate database access to retrieve the security descriptor.

Owner

The Owner field stores the identity (SID) of the security principal that is considered to “own” the object and is granted certain rights that cannot be revoked by the permissions list. The owner is usually a user and in the majority of cases will be the user who created the object, but there is nothing (other than the rules) which requires that to be the case. In particular, it is allowed for the owner to be a group (in which case, any member of that group is considered to own the object).

Every SD has a slot for the owner but it is not always populated. There are a couple of reasons why this can be the case:

- ▶ For some objects, the concept of an owner is not considered meaningful, so no mechanism is provided to set or retrieve the Owner field and there is no default value initialized in it. This applies to all GCD objects and to a small subset of repository objects.
- ▶ The owner might have been explicitly set to null (or defaulted so) deliberately so that no one has the additional access rights conferred by ownership.

In cases where there is an exposed mechanism to modify the Owner field (which for the majority of repository objects there is, through the Owner property), doing so is subject to the following rules:

- ▶ If the user has WRITE_ANY_OWNER access to the object store in which the object resides, the Owner can be set to any valid real SID or to null. (It might not be set to one of the special SIDs).
- ▶ If the user has WRITE_OWNER access to the object in question (but not WRITE_ANY_OWNER access), the Owner might only be set to the user’s own SID (which is also known as “*take ownership*”) or to null.
- ▶ Otherwise, the Owner cannot be modified.

Permission list

The *permission list*, referred to as the *access control list (ACL)* is a collection of permissions or *access control entries (ACEs)*, each of which grants or denies a set of access rights to a security principal.

The order of the ACEs in the ACL has no particular significance, although when the ACL is exposed in the API as a collection of permission objects, it is ordered by source and type to match the order of precedence applied by the access check. The ACEs in the ACL are generally a mix of automatically assigned entries and entries applied “manually” through the API. See the Source field described next.

Each ACE has the following fields:

- ▶ **Grantee**

The SID of a security principal to which the ACE grants or denies access rights, which can be either a real SID or one of the special SIDs. In the API, the SID is exposed as the name of the security principal (GranteeName property).
- ▶ **Type**

Either Allow or Deny. An Allow ACE grants permissions, and a Deny ACE revokes them.
- ▶ **Access Mask**

A bitmask of access rights granted or denied by this ACE.
- ▶ **Inheritable Depth**

An integer value determining the extent to which this ACE can be inherited by security (grand)child objects.
- ▶ **Source**

An automatically populated (read-only) enumerated type indicating the origin of the ACE:

 - Direct - indicates an ACE that either was added manually through the API or is a modified Default-sourced ACE.
 - Default - an ACE that was added by default (described in 5.3.3, “Default security descriptor” on page 161) and has not yet been modified.
 - Template - an ACE that was added as the result of applying a security template (described in 5.3.4, “Security templates” on page 161).
 - Parent - indicates an ACE that was inherited from a security parent (described in “Inheritance proxies” on page 165).
 - Proxy - indicates an ACE that originates from a full security proxy (described in “Full proxies” on page 164).

ACEs with a source of Template, Parent, or Proxy can neither be removed from the ACL nor modified in any fashion through the API. ACEs with source Direct or Default can be removed through the API and allow modifications to the Type, Access Mask, and Inheritable Depth fields (Grantee cannot be changed). If an ACE with source Default is modified, the source is updated to Direct. (Default is purely presentational; functionally, it is treated no differently than Direct). ACEs with source Parent do not form part of the stored security descriptor but will appear in the API representation of the effective ACL of an object. ACEs with the source Proxy are similar.

5.3.3 Default security descriptor

For repository object classes, a mechanism is provided to allow the administrator to define the ACL and Owner that will be set by default on a new instance of the class. This is accomplished through an additional security descriptor, the default instance SD, attached to the Class Definition for the class, and edited via the Default Instance Permissions and Default Instance Owner properties.

Although they are stored together in the default instance SD, the default ACL and default Owner are applied separately, depending on which (if either) of the Permissions and Owner properties were modified before saving the new object. For both, the default value is copied into the new object (with modifications described next). If the defaults are edited, the changes take effect only for subsequently created instances. There is no retrospective effect on instances that already exist.

The #CREATOR-OWNER SID is given special treatment when forming the instance security from the default:

- ▶ If the default Owner is #C-O, the instance Owner is set to the SID of the user issuing the creation request (that is, the user SID from the security context).
- ▶ If an ACE in the default ACL has grantee #C-O:
 - An ACE is added to the instance ACL with grantee set to the requesting user SID and inheritable depth zero, but all other fields are copied from the default ACE.
 - If the inheritable depth of the default ACE is non-zero, in addition, an exact copy (no grantee substitution) of the default ACE is added to the instance ACL.

ACEs with a grantee other than #C-O are copied exactly, with no modifications except that the source is set to be Default.

When a new object store is created, the default security descriptors for system classes are initialized based on two lists of security principals given in the object store creation request, the *administrators* and *general users*. The administrators are given powerful default rights over all objects, although the exact set of rights varies by object type. General users are given read-only access to the majority of objects. In all cases, the default owner is set to #C-O.

5.3.4 Security templates

The *security template* mechanism provides a way to automate updating of an object's ACL as it transitions between versioning, document lifecycle, or application-defined states. The objective is to bring the object's effective access

rights to an appropriate state without “manual” intervention from the user initiating the state change.

Object model

A security template is essentially just an ACL tied to an identifier that determines the object state in which that template is applied.

Security templates come in three types with different packaging:

- ▶ *VersioningSecurityTemplates* and *ApplicationSecurityTemplates* are fully fledged dependent objects exposing a `TemplatePermissions` collection (accessing the underlying template ACL), a GUID property `ApplyStateID`, which identifies the state to which the template is applicable, and a boolean property `IsEnabled`, which allows application of this template to be disabled. For a versioning template, the `ApplyStateID` property is constrained to one of four well-known values representing the four versioning states: `Reservation`, `In Progress`, `Released`, and `Superseded`. For an application template, it might have any (application-determined) value except those four well-known values.
- ▶ A document lifecycle template is not a separate type of object. The template ACL is simply part of the `DocumentState` definition object (again, exposed as `TemplatePermissions` property), with the applicable state being implicitly that of the `DocumentState` object. A boolean property `ApplyTemplatePermissions` determines whether the template ACL is applicable during transition to this state.
- ▶ A group of Versioning and Application security templates representing all that are applicable to a particular object or set of objects are collected together in a *SecurityPolicy* object, which is associated to the relevant objects via their `SecurityPolicy` property (which is often defaulted). Folders, documents, and customer objects (collectively *Containables*) have the `SecurityPolicy` property and participate in the template mechanism, although Versioning templates are only relevant for Documents.
- ▶ `DocumentState` objects, with their attendant template ACLs, are collected together into a *DocumentLifecyclePolicy*, associated with relevant Document objects through their `DocumentLifecyclePolicy` property.
- ▶ Both *SecurityPolicy* and *DocumentLifecyclePolicy* have an additional Boolean property `PreserveDirectPermissions`, which affects the manner in which templates from that policy object are applied.

Template application

A template is applied to an object under the following circumstances and subject to the noted constraints:

- ▶ During a versioning state change (for example: `Checkin`, `Promote`, or `Demote`), if the document has a non-null value for its `SecurityPolicy` property

and if that policy contains an *enabled* (IsEnabled is set to true) VersioningSecurityTemplate with ApplyStateID corresponding to the new versioning state. If those conditions are not met, the object's ACL is not modified at all.

- ▶ During a lifecycle state change, if the DocumentState for the new state has ApplyTemplatePermissions set to true. (If ApplyTemplatePermissions is false, no ACL change is made.)
- ▶ During an ExecuteChanges operation that requests the ApplySecurityTemplate action, if the object has a non-null SecurityPolicy containing an enabled ApplicationSecurityTemplate with ApplyStateID equal to that given as the parameter to the ApplySecurityTemplate action. (An error is given if those conditions are not met.)

In all these cases, the template ACL is applied to the object's stored ACL in the same way:

- ▶ Any existing ACEs with source Template in the object's ACL are removed.
- ▶ In addition, if the policy object from which the template is drawn has PreserveDirectPermissions set to false, any ACE with source Default or Direct ACEs are also removed. The purpose is to allow the administrator to configure things so that the template completely controls access to the object without the possibility of being overridden by ACEs applied manually.
- ▶ ACEs from the template ACL are copied into the target ACL. This is done the same way that default ACEs are applied (including the special treatment for #C-O ACEs) except that the copies are given source Template rather than Default.

It is permissible for the template ACL to be empty, in which case the last step is skipped (but *only* the last step).

Also, as described above, template ACEs are copied into the ACLs of the objects to which they are applied. Therefore, changes made to the template ACL have no effect on objects to which that template has already been applied.

5.3.5 Proxies

Proxying is a mechanism by which the security descriptor of one object (the *proxied object*) can be replaced by or augmented by the security descriptor of another object (the *proxy object*). Replacement is referred to as *full proxying*. Augmentation is referred to as *inheritance* or *partial proxying*. In both cases, the relationship between the proxied object and a proxy is represented by a singleton object-valued *proxy-defining* property of the proxied object. Setting a value for the property establishes (activates) proxying and removing the value severs the

proxy relationship. The proxy-defining nature of a property is specified as its Property Definition, as either None (ordinary non-proxy defining property), Full, or Inheritance.

Proxying is transitive. That is, if object A has a proxy-defining property referring to object B, and B has a proxy-defining property referring to C, A's security (as well as B's) is influenced by that of C. The server prevents proxying loops (where a chain of proxy relationships leads back to the object from which one started), producing an error at the point the attempt is made to set the property that closes the loop.

Proxying is only available for repository classes; both the proxied and proxy objects must reside in an object store. Proxy-defining properties are often custom properties but there are a number of built-in system proxy-defining properties, in some cases with additional special behavior.

There is no restriction on proxying between different types of objects. For example, a Folder can act as a proxy for a Document or a Document for a Link. A consequence is it might be necessary to assign access rights to a proxy object that have no meaning for that type of object, in order for those rights to apply to proxied objects.

When a proxied object is retrieved or accessed for update, the server evaluates an *effective security descriptor* for the object by traversing each active proxy relationship, evaluating the effective security descriptor for each proxy object (a recursive process) and then applying those to replace or augment the stored security descriptor for the object. This process can involve fetching a number of additional objects, and to mitigate the cost, the server maintains a cache of recently fetched proxy objects. This *object-security cache* is particularly effective when retrieving a set of objects sharing the same proxy, as is typical for subfolders and relationship objects.

Full proxies

Assigning a value to a proxy-defining property of type Full results in the effective security descriptor (and active markings) of the proxied object being completely replaced by that of the proxy. In this case, the Permissions collection presented in the API for the proxied object will reflect exactly the ACEs in the effective security descriptor for the proxy, but with the special source type Proxy. The Permissions collection and the Owner property will both be read-only.

A class can define multiple proxy-defining properties of any mix of types, but among those of type Full only one can have a value at a time. (An attempt to set a value for a second full proxy-defining property will incur a "constraint-violated" error.) This places some limits on the usefulness of the feature, since if two applications want to full proxy a particular object in two ways, they can only do so

cooperatively. No restriction is placed on setting or modifying inheritance proxy-defining properties when a full proxy is in effect, but those will have no influence on the effective security for the object.

When a full proxy is in effect, the object's own stored security descriptor is rendered invisible and inaccessible for direct modification. However, if a security template is applied to a fully proxied object, the changes are made to the stored security descriptor in the normal way and will become visible and effective if the full proxy relationship is severed.

There are a few system properties that are full proxy-defining and in all cases these require a value. This reflects the fact that for the objects in question, there is no value in them having independently controllable security. As a consequence, none of these objects have the Permissions or Owner property. Table 5-2 lists the classes and properties to which this applies.

Table 5-2 Proxy classes and properties

| Proxied class | Property defining full proxy | Proxy object class |
|--------------------------------------|------------------------------|-------------------------|
| Referential Containment Relationship | Tail | Folder |
| Component Relationship | Parent Component | Document |
| Hold Relationship | Hold | Hold |
| Sweep Relationship | Sweep | Policy Controlled Sweep |
| Job Sweep Result | Controlling Object | Sweep Job |
| Policy Sweep Result | Controlling Object | Sweep Policy |
| Thumbnail | Input Document | Document |

Inheritance proxies

Security inheritance, like its genetic counterpart, is the passing of inheritable traits - in these cases, ACEs rather than genes - from a parent (the proxy) to a child (the proxied object). This analogy (and a certain amount of history) is the reason behind the use of the ACE source designation Parent. The analogy becomes a bit stretched when considering that a proxied object can have any number of "parents", represented by values in an arbitrary number of inheritance type proxy-defining properties.

The effective security descriptor for an object with active inheritance proxy relationships is constructed by combining the stored SD with copies of the *inheritable* ACEs from the effective security descriptor of each proxy. ACEs from

different proxies are combined on an equal footing and all are given source Parent, regardless of the source for the ACE from which the copy was made.

Inheritability of an ACE is determined by its Inheritable Depth setting, with the following rules:

- ▶ A value of 0 means that the ACE is not inheritable and will not appear in the ACL of the proxied object.
- ▶ A value of 1 indicates that the ACE will be inherited through one level of proxying only (the administrative UI refers to this as “Immediate children”). When such an ACE is copied to an inheriting object, the copy has Inheritable Depth 0 as a reflection of not inheriting further.
- ▶ Other positive values similarly specify inheritance through a maximum number of levels, with the value being decremented as a copy is made for each inheritance step.
- ▶ A value of -1 allows the ACE to be inherited to arbitrary depth, through any number of transitive proxy relationships.
- ▶ Other negative values have a subtle effect, because these ACEs (which are referred to as inherit-only) are not considered when evaluating access for the object on which they appear. They can be inherited, with their inheritable depth being transformed as follows on the first inheritance step:
 - -2 becomes -1, thus it means “inherit-only to any depth”
 - -3 becomes 0, thus meaning “inherit-only to immediate children”
 - -4 becomes 1, -5 becomes 2, and so on.

As well as ACEs that are directly applied, those that appear in a default instance ACL or template ACL can also be marked as inheritable. Thus, for example, a security template can apply inheritable ACEs. A number of system classes have built-in inheritance proxy-defining properties, as shown in Table 5-3.

Table 5-3 System classes and their properties

| Proxied class | Property defining inheritance proxy | Proxy object class |
|----------------------|--|---------------------------|
| Folder | <i>Parent</i> | Folder |
| Document | Security Folder | Folder |
| Custom Object | Security Folder | Folder |
| Annotation | <i>Annotated Object</i> | Containable |
| Task | <i>Coordinator</i> | Containable |
| Class Definition | <i>Superclass Definition</i> | Class Definition |

| Proxied class | Property defining inheritance proxy | Proxy object class |
|---------------|-------------------------------------|--------------------|
| Recovery Item | Recovery Bin | Recovery Bin |

Of particular note are the first three. A folder can inherit from its parent folder, and it from its parent, and so on, all the way to the root folder. Documents and custom objects can inherit from a folder, which is usually a folder in which the object is filed, although that is not required. Thus, by combining these two features, it is possible to manage security for the entire folder tree or for a subtree and all its contents from the root of that tree/subtree.

The properties with names in italics in Table 5-3 on page 166 all require a value. It appears impossible to have, for example, a folder that does not inherit from its parent. (In contrast, the security folder property does not require a value, so inheritance is always optional.) To overcome this deficiency, in three of the cases in Table 5-3 on page 166, a companion Boolean property is defined that disables this apparently required inheritance, shown in Table 5-4.

Table 5-4 Proxied class and inheritance

| Proxied class | Inheritance disabling property |
|---------------|----------------------------------|
| Folder | Inherit Parent Permissions |
| Task | Inherit Coordinator Permissions |
| Recovery Item | Inherit Recovery Bin Permissions |

The default value for each of these properties is True, which means that inheritance is enabled. Setting the value to False disables inheritance. The effect applies solely to inheritance through the system property from the preceding table; any additional custom proxy-defining properties are unaffected.

Note: To date, no use case disables inheritance for class definitions or annotations.

5.3.6 Markings

The origin of this mechanism is the military and intelligence notion of document classifications and clearance levels:

- ▶ Each person is assigned a clearance level indicating the maximum sensitivity of documents they are allowed to see.
- ▶ Each document is labeled with a classification indicating its sensitivity.

- ▶ A person is allowed to see only documents that are classified at or below their clearance level. For example, someone with Top Secret clearance level can see Top Secret, Secret, and Unclassified documents. Someone with only Secret clearance level cannot see Top Secret documents.
- ▶ Furthermore, a person who is responsible for classifying new documents can do so only up to their own clearance level. Therefore, a person with only Secret clearance cannot classify a new document as Top Secret.

This approach is sometimes referred to as *labeling* or as *mandatory access control* (in contrast to the features described earlier that can be categorized as *discretionary access control*).

The Content Platform Engine manifestation of this type of access control is via *marking-controlled properties*. A marking-controlled property is in every respect an ordinary string property (single-valued or multi-valued), except that its permitted values (called *markings*) are drawn from an administratively defined set called a *marking set*. Its current values influence the access granted to the object in a manner that is detailed next.

This approach generalizes upon the basic classification/clearance mechanism in several dimensions:

- ▶ An object can have multiple marking-controlled properties, drawn from multiple marking sets, combining in their influence on the access granted. For example, there might be a Classification property drawing from the Clearance Levels marking set, and a Project property, designating a project from the Projects marking set. The combined effect depends on the clearance levels assigned to different users and the projects on which they are permitted to work.
- ▶ A marking set can be defined as imposing a hierarchical order of precedence on the markings defined within it (for example, in the clearance levels example, where Top Secret > Secret > Unclassified. Or, it can define the markings as unrelated (likely be the case in the projects example).
- ▶ In the classification/clearance scheme, failing to have sufficiently high clearance with respect to a particular document acts as a blanket “off switch” - you are not allowed to see or in any way manipulate that document. In contrast, markings offer fine-grained control by way of an access mask that can revoke any or all of the rights granted to the object via its security descriptor.
- ▶ The right to apply and remove a particular marking can be assigned independently of the right to use objects to which that marking has been applied. (Compare with the statement regarding “classifying new documents”.) Therefore, someone might, in principle, be given the right to classify objects as Top Secret even though that person only has clearance to

see Secret documents. More likely, that person be allowed to designate objects as belonging to a particular project even though that person is not a member of that project team.

- ▶ Rights to use, apply, or remove markings can be assigned to both individuals (users) and groups.

This approach operates in the following manner:

- ▶ A `MarkingSet` object, defined at the domain level, has a name, a type of Hierarchical or List, and a list of dependent marking objects. In a hierarchical marking set, the marking objects must appear in order, highest precedence first. For a list type, the order has no significance.
- ▶ Each marking object specifies a value (string) for the marking and defines the usage and access control effect of that marking value via the following fields:
 - An ACL (exposed in the API as a `Permissions` collection). The ACEs in this ACL grant or deny some combination of `UseMarking`, `AddMarking`, and `RemoveMarking` access rights. (Other bits can be set, but have no effect.) The ACL is evaluated during the access check on an object that either has this marking value applied to it, or to which an attempt is being made to apply it. That evaluation depends on the type of the marking set, as described in 5.3.7, “The access check” on page 169.
 - A *constraint mask*, a bitmask of access rights representing the rights that are *revoked* if a user with insufficient rights to the marking attempts to access an object to which that marking has been applied. (Therefore, markings only subtract rights, they do not add them.)
- ▶ A custom `PropertyDefinitionString` can optionally reference a `MarkingSet`, turning the defined property into one that is controlled by that marking set. This establishes the marking values in that marking set as the permitted values for the property, and activates the access control effect of the marking value for instances of the class to which the defined property is added.

The access control effect of markings applied to an object acting as a full proxy also affect the proxied object, completely supplanting the effect of any markings on that object.

5.3.7 The access check

The *access check* is the process of evaluating the access control measures that apply to a particular object for a particular user in order to determine the set of access rights that are in effect for that object and user combination.

The following sections describe the circumstances and manner in which the access check is carried out.

When and how

All independent objects are subject to an access check during retrieval and update. In most cases, that access check is based on evaluating the effective access granted by the object's effective security descriptor and markings.

However, a number of object types diverge from this general rule:

- ▶ The overall effective access for an object store is determined by augmenting that resulting from evaluating the object store's own ACL with additional rights determined from the Domain's ACL (Table 5-5).

Table 5-5 Domain ACL

| Domain ACL grants | Added to object store effective access |
|-------------------|--|
| READ | READ + READ_ACL |
| WRITE | WRITE + READ_ACL + WRITE_ACL |
| DELETE | DELETE |

- ▶ The effective access to other independent GCD objects (besides object stores) is determined by evaluating the Domain's ACL. (No other GCD objects have their own ACL.)
- ▶ Access to class descriptions, for which READ is the only relevant right, is controlled by the scope from which the class description is drawn. To access a class description from an object store scope requires CONNECT rights to that object store. To access a class description from domain scope requires READ rights to that domain.
- ▶ For all repository objects, the result of the ACL check is augmented with the WRITE_OWNER right if the user has WRITE_ANY_OWNER access to the object store in which the object resides.
- ▶ Queue Items (SecurityPropagation/Event/DocumentClassification), Index Requests, Replication Journal Entries, and Sweep Queue Entries do not have instance security. Instead, access to these types of objects is determined by evaluating the default instance ACL for the object's class.
- ▶ There are no access checks for realm, user, and group objects. (It is sufficient for the user to be authenticated in order to be able to see these objects, which are immutable.)

Object retrieval

This form of access check takes place during a GetObjects operation (or equivalent) and during the refresh phase of an ExecuteChanges operation, with the purpose of determining whether the caller is allowed to see the object and whether it needs to form part of the response.

For GCD objects and ClassDescriptions, being allowed to see the object is simply based on the presence of the READ access right in the effective access for the object, evaluated from an ACL as described next.

For repository objects, the access check is in two parts, both of which must be satisfied:

- ▶ The effective access evaluated from the ACL of the object store in which the object resides must include CONNECT right.
- ▶ The effective access evaluated from the object's effective security descriptor and markings must include either READ or WRITE_OWNER permission (or both). The rationale for the inclusion of the latter is that it allows the owner of an object or someone with WRITE_ANY_OWNER access to the object store to retrieve the object regardless of the state of the ACL. Therefore, it provides an escape mechanism by which the ACL can be repaired if it has been put into a state that grants no access to anyone.

In the event that the access check fails, the intent is that the system behaves exactly as though the object did not exist. Therefore, the following actions occur:

- ▶ If the object is being retrieved directly, a “not found” error is returned.
- ▶ If the object forms part of a collection, it is simply dropped from that collection.
- ▶ If the object will be returned as the value of a singleton property, a null value is substituted.

If the access check succeeds, the object is returned with values for all the properties dictated by the property filter included, except for these properties:

- ▶ Any values that are independently persistable objects reached through recursion are subject to an independent access check and handling as described earlier.
- ▶ The Permissions collection and the Owner property are returned as empty/null unless the effective access mask that was used for the primary access check also includes either READ_ACL, WRITE_ACL, or WRITE_OWNER.
- ▶ If the property filter demands that content be returned, an additional check is made for VIEW_CONTENT access in the primary effective access mask.

ExecuteChanges

ExecuteChanges is the name of the underlying server request type through which creation, updates, and deletions are performed.

Access checking during ExecuteChanges is for determining whether the creation, update, or deletion operations in the batch are allowed. Each object in the batch is access-checked individually, but any failure causes the entire batch to be abandoned (rolled back).

For GCD objects, the access check operates on the effective access determined from the Domain's ACL, or in the case of an object store, the combination of the Domain and object store ACLs. The following checks are made:

- When creating an object, the effective access mask must include CREATE_CHILD right.
- When updating an object, WRITE right is required, and in addition, if the object has an ACL and that ACL is modified, WRITE_ACL right.
- Deleting an object requires DELETE right.

For repository objects, the checking is more complicated and differs depending on the basic operation performed - create, update, or delete:

► Create

The access check is in three parts:

- First, the effective access to the object store in which the object is to be created must include STORE_OBJECTS right.
- Second, the effective access evaluated from the ACL of the ClassDefinition for the class of object being created must grant CREATE_INSTANCE right.
- Finally, a provisional security descriptor is formed for the object from the default defined by the object's class, then the effective access granted by that SD is combined with WRITE, MINOR_VERSION, and MAJOR_VERSION to form an overall effective access mask that is fed into a property access check for any properties for which values are given in the creation request.

► Update

Again, the access check is in three parts:

- First, the effective access to the object store in which the object resides must include MODIFY_OBJECTS right.
- Second, the effective access evaluated from the object itself is checked for specific rights according to the actions requested, as described in Table 5-1 on page 156:

Special case: Checkout creates a reservation document and is subject to the normal creation checks for that object, in addition to the checks applied to the checked-out document.

Special case: InstallAddOn is an action (treated as an update) on an ObjectStore and is therefore subject to the previously described access checking rules for GCD objects. However, in addition, all the operations performed as part of installing the add-on (typically creating classes and properties) are subject to normal access checks, as though they were performed through the API directly.

- Last, the effective access of the object is fed into a property access check. A simple Update action involves no second phase access check, only the property access check.

► Delete

The access check is in two parts:

- First, the effective access to the object store in which the object is to be created must include REMOVE_OBJECTS right.
- Last, the effective access for the object itself must include DELETE right.

Special case: The deletion of the system class, ReferentialContainmentRelationship (RCR), is permitted if the effective access (which is evaluated from the tail folder) includes either DELETE or UNLINK right.

Special case: The deletion of a reservation Document (also known as a “cancel checkout”) is permitted if its effective access includes either DELETE or MAJOR_VERSION or MINOR_VERSION right.

Handling of compound actions

A change request can apply multiple actions to a single object (for example, Create + Checkin or Checkout + Lock), which is referred to as a *compound action*.

Each action in a compound action is subject to separate access checking according to the rules stated. However, the object’s individual effective access rights are evaluated only once, prior to the first action (that is, before any changes are made). Those effective access rights are used in the access check for all the subsequent actions. Thus, security changes made by one action do not affect the access check for subsequent actions (because the effective access is not reevaluated to take account of those changes).

Property access check

The third phase of a creation or update operation access check validates that the caller is allowed to set or modify properties presented in the request (the “dirty” properties). The input to this check is the effective access mask determined during the second phase, and that is compared with the metadata-defined or implicit *modification access required* (MAR) for the property. The rule is that the operation is permitted if all the rights present in the MAR are also present in the

effective access mask. This rule must be satisfied for every newly set or modified property; otherwise, the whole operation will fail.

For most system properties, there is no explicitly defined MAR, and for those system properties, an implicit mask consisting of just WRITE right is used. For most properties, setting or modifying them simply requires WRITE permission to the object to which the properties belong. A number of system properties do however have an explicit MAR (Table 5-6).

Table 5-6 Properties that explicitly define modification access

| Property | Modification access required |
|-----------------------------------|------------------------------|
| Permissions | WRITE_ACL |
| Owner | WRITE_OWNER |
| ReplicationGroup | WRITE_ACL |
| SecurityPolicy | WRITE_ACL |
| CmRetentionDate | WRITE_ACL |
| SecurityFolder | WRITE_ACL |
| DefaultRetentionPeriod | WRITE_ACL |
| Inherit Parent Permissions | WRITE_ACL |
| Inherit Coordinator Permissions | WRITE_ACL |
| Inherit Recovery Item Permissions | WRITE_ACL |

Special rules apply to setting Owner, depending on how the WRITE_OWNER effective access comes about, as described in “Owner” on page 159.

The MAR for custom properties can be defined by the property definition author, with again WRITE being the de facto value. Proxy-defining properties need to be given a MAR of WRITE_ACL because modifying the property value affects the effective ACL for the object.

In addition to the MAR check, certain property types are subject to further checks:

- ▶ Singleton object-valued properties (OVP)

Setting a value for a singleton OVP is subject to an additional access check on the object specified as the value. This check evaluates the caller’s effective access to the specified object (as determined by that object’s effective ACL and markings) and tests it against the metadata-defined or implicit *target access required* (TAR) for the property, allowing the operation to proceed if all

the rights present in the TAR are also present in the effective access. The default (implicit) TAR is READ, but many system properties have more stringent requirements.

For example, the TAR for the RCR.Tail is LINK. Therefore, it is necessary to have LINK permission to a folder in order to file things into it.

As with the MAR, for custom properties, the TAR is controlled by the property definition author.

There is no additional access check for unsetting (nullifying) a singleton OVP. Only the MAR check applies.

► **Marking-controlled properties**

Values placed into or removed from a marking-controlled property (MCP) are subject to access checks on the marking objects corresponding to those values:

- If setting a value of a singleton MCP where previously there was no value, the caller must have ADD_MARKING effective access to the corresponding marking object.
- If nullifying a singleton MCP, which previously had a value, the caller must have REMOVE_MARKING access to the marking object.
- Changing the value of a singleton MCP from one non-null value to another non-null value is treated as a remove followed by an add. Therefore, it requires REMOVE_MARKING access to the initial value's marking and ADD_MARKING access to the new value's marking.
- For updates to a list MCP, the server computes the net deltas - additions and removals - and demands ADD_MARKING or REMOVE_MARKING access to the corresponding marking objects.

The manner in which the effective access to a marking is determined is described in "Markings check" on page 177.

Search execution

Search applies an access check to any object contributing columns to a result row. (In a joined query, each From class is considered as contributing to the row, even if no properties of that class appear in the SELECT list.) This access check is the same as for object retrieval - READ or WRITE_OWNER rights are required - except that if the query includes a CBR clause, VIEW_CONTENT is also required. A failure is handled in the same way that it is handled for collection retrieval, that is, the row is simply dropped from the result set.

The access check is accomplished by adding columns to the SELECT list specified by the caller as needed to determine the security descriptor and the values of any proxy-defining and marking-controlled properties for each

contributing object, from which can be determined the effective access to the object.

Effective access

Key is the calculation of effective access to an object, determined from the effective security descriptor and markings applicable to the object. The overall effective access is, in general, the result of combining three bitmasks, produced by evaluation of the owner, ACL, and markings against the security context of the caller, in the following way:

$$\text{effective-mask} = (\text{owner-mask} \mid \text{ACL-mask}) \& \sim\text{markings-constraint-mask}$$

The owner mask and ACL mask are combined, and then any rights constrained by markings are subtracted. Note a couple of key points:

- ▶ Denial ACEs in the ACL do not (cannot) revoke rights granted through ownership.
- ▶ Markings do not grant additional rights, they only revoke rights, and that revocation is absolute (cannot be overcome via the ACL or ownership).

For an object without markings, the markings check is skipped and zero is used for markings-constraint-mask.

Owner check

Subject to the following caveat, the Owner check returns a bitmask of either all zeros or of READ_ACL+WRITE_ACL+WRITE_OWNER, depending on whether the caller is determined to be an owner of the object. That determination is made by comparing the Owner SID from the security descriptor with the SIDs in the security context. If there is a match with any of those SIDs (user or group), the caller is an owner and receives the non-zero owner mask. If there is no match (or if the Owner SID is null), the owner mask is returned as zero.

The caveat is that when evaluating access to a repository object, possession of the WRITE_ANY_OWNER right to the object store in which that object resides causes WRITE_OWNER right to be added to the owner mask. (This obviously only has impact if zero is otherwise returned.)

ACL check

The ACL evaluation phase combines the access masks from the ACEs in the ACL according to the following rules:

- ▶ Only ACEs whose grantee SID is present among the SIDs in the security context are considered, while all others are ignored. The only relevant ACEs are those whose grantee is the calling user or a group of which that user is a member.

- ▶ Inherit-only ACEs are ignored.
- ▶ ACEs with source Direct/Default take precedence over those with source Template, which in turn take precedence over those with source Parent.
- ▶ Within each source category (so as a second-level order of precedence), Deny ACEs (which remove rights) take priority over Allow ACEs (which add rights).

So, logically, we calculate six separate bitmasks, one for each (source, type) permutation (treating Default and Direct as one). Each of those bitmasks is the result of ORing together the access masks from the all relevant ACEs belonging to that permutation (or is zero if there are no such ACEs). The six bitmasks are then combined according to the following algorithm:

```

ACL-mask = parent-allow-mask
ACL-mask = ACL-mask & ~parent-deny-mask
ACL-mask = ACL-mask | template-allow-mask
ACL-mask = ACL-mask & ~template-deny-mask
ACL-mask = ACL-mask | direct-allow-mask
ACL-mask = ACL-mask & ~direct-deny-mask

```

An important point is that denials (Deny ACEs) override grants (Allow ACEs) with an equal or lower precedence source (for example: Template Deny overrides Template or Parent Allow) but are themselves overridden by grants with a higher precedence source (example: Direct).

Markings check

This phase returns a bitmask of rights that are unconditionally revoked from the caller - the constraint mask - as a result of insufficient access to markings applied to the object. This is determined by forming a list of the marking objects corresponding to all the values of all marking-controlled properties of the object, evaluating the caller's effective ask to each of those markings, and ORing together the constraint masks of any for which the effective access fails to include USE_MARKING right.

The key part is how effective access to a marking object is evaluated, which has relevance both here and to the additional property access check for marking-controlled properties described in "Property access check" on page 173. This evaluation varies according to the type of marking set to which the marking belongs.

List

In this type of marking set, the markings are independent of each other. The evaluation of effective access simply relies upon the ACL of the individual marking, which is evaluated according to the rules given in “ACL check” on page 176 and simplified because a marking ACL never includes anything but direct ACEs.

Hierarchical

In this case, the evaluation is more complex because it must include the order of precedence of the markings in the MarkingSet and the following rules:

- ▶ Granting rights to a higher precedence marking implicitly grants the same rights to all lower precedence markings. For example, someone given Top Secret clearance is inherently able to access Secret and Unclassified documents.
- ▶ Denying rights to a lower precedence marking must also deny those rights to all higher precedence markings. For example, if someone is not allowed to see Secret documents, clearly they also are not permitted to see Top Secret documents.

Therefore, evaluation of the effective access to a marking in a hierarchical MarkingSet needs to include not only all the ACEs of the ACL on that marking itself, but also the Allow ACEs from any higher precedence markings and the Deny ACEs from any lower precedence markings. Although it is not implemented this way, think of this logically as forming a composite ACL from those three components and then performing a normal evaluation of that composite ACL.

5.3.8 Auditing

In some environments, the assurance that the Content Platform Engine will only allow properly authorized operations might not be sufficient. It might be necessary for legal or other reasons to have an actual record of attempts to carry out unauthorized actions or even of permitted operations.

This is the purpose of the auditing feature, which allows an administrator to enable recording of either failed operations, successful operations, or both. This can be done with a considerable amount of selectivity:

- ▶ Auditing is enabled by class. For example, it can be turned on for some perhaps more sensitive classes but left disabled for others of lesser sensitivity.
- ▶ Within a class, auditing can be controlled by operation. For example, it can be enabled for deletions but not for anything else.

- ▶ For each enabled operation, the administrator can select whether a record is made of attempts to perform that operation that failed due to insufficient access, of successful operations, or both.
- ▶ Finally, a filter expression can optionally be specified that is applied to an object that is a candidate for auditing based on the preceding settings. Only if the object satisfies the filter condition will an audit record be written. This allows auditing to be narrowed to objects having only certain property values or to where certain properties are modified.

There is no mechanism for determining auditing based on the identity of the user.

Operations to be enabled for auditing are expressed in terms of events triggered by those operations, and audit records take the form of stored Event objects of various classes. These are the same objects that participate in the Content Platform Engine Events and Subscriptions feature. For a full description of the Event classes and the corresponding operations, see the following document:

http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.admin.tasks.doc%2Fevents_reference.htm

In all cases, the audit record includes the date and time of the operation and the identity of the user that initiated it. For everything apart from Query Event, the record also includes the identity of the object on which the operation was performed. (A query is of a class, not of any one object.) In addition, for all events other than Query Event, Get Object Event, and Get Content Event, the administrator can elect for the audit record to contain property values from the object at the time of the operation. The record can be either a complete snapshot of the object's state (either before or after the update or both) or in the form of selected properties copied ("audited as") from the source object into custom properties defined in the event class.

Audit records are independent of the object for which they record an operation and, in particular, they are not automatically deleted when the source object is deleted. An automatic disposal mechanism is however provided, through an Audit Disposition Policy, which periodically scans the audit records deleting those that satisfy a filter condition, often based on date.

5.4 Security best practices

The richness of the access control mechanisms described in the previous section can be both a blessing and a curse. The richness means that a wide variety of security requirements can be addressed but it is not always obvious how best to meet those requirements. This section will give some guidance on that issue.

5.4.1 Physical security measures

The Content Platform Engine rigorously applies the access controls that have been defined to any requests that are received via its APIs. But those controls are only effective to protect the resources managed by the Content Platform Engine if *the only way those resources can be accessed is through the APIs*. If users are able to bypass the API and access resources directly, the Content Platform Engine access controls are completely compromised. Therefore, it is essential that an organization have in place “physical” security measures to prevent such direct access.

A detailed examination of this topic is beyond the scope of this publication, but a brief list of areas for consideration follows. These are all common issues that arise in any distributed system:

- ▶ Databases hosting the GCD and object stores must be secured against unauthorized access. Generally, access is restricted to database administrators (who ideally are not P8 users) and the identities used by the Content Platform Engine itself to access the databases (as defined in the application server data sources configuration).
- ▶ The server host systems and the application server instances on those systems in which the Content Platform Engine is running need strictly controlled access, so that only appropriate people can stop and start the Content Platform Engine and be able to modify the application server configuration.
- ▶ File systems used for content storage need controls in place to prevent access by anyone but the operating system identities under which Content Platform Engine server instances execute and those with responsibility for maintenance operations, such as backup. File systems used for text index storage must be protected similarly, but also allow access by the CSS server processes.
- ▶ Fixed devices used for content storage need to give access only to the identity configured for use by the Content Platform Engine (in the Fixed Content Device object) and to device administrators.
- ▶ The network hosting the Content Platform Engine and CSS servers, databases, and storage devices need to be secured against snooping and attack.

Important: This area is critical for any network connections over which raw authentication credentials (such as passwords) are transmitted. We strongly advise that you configure these connections to use Transport Layer Security (TLS)/Secure Sockets Layer (SSL) or equivalent.

As a further measure to prevent direct access to content, the content encryption feature can be enabled for selected storage areas. This feature ensures that even if access controls on the storage device are bypassed, the content will be unreadable.

Compatible disk sector encryption technology can also be used to provide similar protection for the GCD and object store databases as well as for content.

5.4.2 Directory service configuration

Configuring the directory service is one of the most important first steps in setting up a P8 system. A particularly critical aspect is choosing which attribute of user and group objects serves as the unique identifier from which SIDs are constructed. This decision cannot easily be changed after SIDs are stored in ACLs (which happens as soon as the domain is fully initialized), so it is essential to get it right first time.

The default unique identifier attribute for each of the supported directory services is shown in Table 5-7.

Table 5-7 Default unique identifier attributes by directory service type

| Directory service | Unique attribute |
|--|------------------|
| Microsoft Active Directory and AD LDS | objectSid |
| Novell eDirectory | gulD |
| Oracle Internet Directory | orclguid |
| Oracle Directory Server Enterprise Edition | nsuniqueid |
| CA Directory | cn |
| IBM Tivoli Directory Server | ibm-entryuuid |

Whatever attribute is chosen needs to be either enforced as read-only by the directory service or treated as read-only by the administrators of the directory service. A change in the value for a particular user or group invalidates any stored SIDs referring to that principal and likely results in that principal losing any access they might have been granted.

With the exception of the `cn` attribute employed for CA Directory, all of the defaults listed in Table 5-7 have the required characteristics of uniqueness and immutability guaranteed by the hosting directory service. We strongly advise that you use them. Alternatives that are merely unique (such as distinguished name) should only be used if procedures are in place to prevent changes.

5.4.3 Defining the security approach

The key to defining an effective approach to access control is to make security considerations an integral part of defining the general data model. Identify the document, folder, and other classes that are to be used, and also in what ways they are to be used and therefore what access rights will need to be granted to them to allow their purposes to be served.

In developing this model, often different areas of responsibility are identified with correspondingly different access rights needed to act on those responsibilities. For example, being responsible for creating and editing a particular type of document requires one set of access rights but being responsible for approving it for publication requires a different set. It is important that the security model clearly identifies these distinctions, and in refining this aspect of the model, you need to remember a couple of general principles.

Avoid overlapping responsibilities. Overlapping responsibilities creates opportunities for accidental violation of intended security controls and for malfeasance. Continuing the previous example, it is likely undesirable for the same individuals with responsibility for authoring documents to also be responsible for approving them for publication.

Grant only the privileges (for example, access rights) that are necessary for a particular task and no more. For example, if a particular task requires only viewing the content of a document, do not give permission to modify or delete the document. Again, the objective is to prevent accidental or malicious violation of the intended security controls. Although this might seem obvious, it is not necessarily always straightforward to apply this principle successfully.

In addition to identifying the responsibilities and corresponding access rights for objects themselves, consider for the model:

- ▶ Property updates: Under what circumstances can it be possible to modify a particular property? This will determine the appropriate value for the modification access rights defined for the property. (For proxy-defining properties, always include WRITE_ACL.)
- ▶ Relationships between objects: When is it possible to establish a relationship between two objects by setting an object-valued property of one to reference the other? This will dictate the target access required for the property.
- ▶ Requirements for commonality of access between related objects: For example, object A is related to object B and the same individuals need to have similar access rights to both. These are potential opportunities for the use of proxying.

Having fully developed the security model, the next task is to determine how that can be expressed concretely in terms of security descriptors, templates, proxy relationships, and markings. Determine how to define the procedures for identifying the individuals to be given particular responsibilities and for assigning them the rights required to carry out those responsibilities. Design how the security model will be *implemented*.

Two elements of a successful implementation are important:

- ▶ Wherever possible grant rights to groups rather than to individual users.
To fully exploit this method means creating a group corresponding to each area of responsibility or distinct set of access rights identified in the model, making responsible individuals members of the group, and then assigning the group as the grantee in ACEs.

- ▶ Make the maximum use of defaults.

Some applications can set and update the security properties on individual objects, either based on their own logic or input from users. However, these are almost always adjustments to the defaults defined for the object class. By far, the most common case is that the class defaults are allowed to take effect without modification.

It is therefore essential that class defaults are defined appropriately. This does not mean only the default permissions and owner. If security templates are to be used through security policies or document lifecycles, defaults might need to be defined for the properties that reference those objects. Similarly, if markings are to be used, defining default values for marking-controlled properties can be desirable. It can even be appropriate to define default values for proxy-defining properties, although that is less likely.

In addition, it is advisable to keep the number of ACEs in any ACL reasonably small (fewer than ten) since this results in more economical storage and makes it easier for administrators to understand the overall effect of the ACL.

Supporting trusted applications

P8 is often used as a “back end” by “middle-tier” applications that offer a higher level service to their clients, making calls to P8 on behalf of those clients. Such an application will typically employ its own authentication and authorization techniques, but the calls it makes to P8 are nevertheless subject to P8’s own access controls. It is therefore preferred that the application’s authentication be aligned with that of P8 (typically by use of JAAS). It is typically preferred that the calls to P8 be made in the context of the end client, so that P8 access controls can be expressed in terms of those users.

However, that is not always possible. It can be necessary in some cases for the application to make calls to P8 under an identity that is distinct from that of any client it is serving and that has sufficient access to objects in P8 to enable it to carry out any operation on behalf of any of its users.

This is a somewhat tricky proposition, since it implies that the usual P8 access controls must be relaxed for the identity under which the application makes calls, and so further implies that the P8 administrator must trust the application to prevent its users from obtaining access that is not allowed if making requests directly to P8. This is not something to be taken lightly.

Assuming that such trust exists, the following approach for supporting such applications is preferred:

- ▶ A separate directory service user needs to be defined exclusively for use by the one application. It is completely inappropriate for the application to call P8 using an identity that can also be used by direct users or by another trusted application.
- ▶ The application needs to be configured to perform a JAAS login as that user. The configuration settings containing the credentials for the login need to be stored securely where only the application can get at them.
- ▶ The user (directly, not via any group) needs to be granted the required access in the default permissions for any class of which instances will be acted upon by the application.
- ▶ If need for the application ceases, the directory service user needs to be deleted, thus ensuring no possibility that the elevated rights granted to the application can be taken over by anyone or anything.

5.4.4 Planning for evolution

Access control requirements will inevitably change over time. Individuals can join the organization, others can leave, responsibilities can be reassigned, new applications can be introduced, and so on. Being able to evolve the security implementation to meet these new/changed requirements is therefore essential. If the wrong approach is taken in the initial implementation, accommodating such evolution can be impossible or at best extremely time-consuming.

The key to making such evolution possible in relatively straightforward fashion is to heed the advice given earlier to grant access to groups rather than to individual users. This is particularly critical for the lists of administrators and general users given at object store creation. It is almost as important for objects within an object store (although for those there is another technique described in 5.4.5, “Role-based access control using inheritance” on page 186 that can be used to some extent to overcome the same issues).

To see why not following this advice can cause immense problems, consider a couple of examples:

► Example 1

User X has sole responsibility for a particular task and is granted rights to perform that task directly (not via a group membership) in ACEs applied to various objects. Now, X leaves the organization and is replaced by Y.

In order to bring about this change, someone with sufficient privilege (specifically, an object store administrator) must locate all the objects with ACEs in which X appears and replace those ACEs with ones granting Y the same rights.

There is no mechanism for querying for objects based on grantees in the ACL. (The internal stored format of the security descriptor makes this impossible.) In limited circumstances, it can be possible to identify the relevant objects in some other way, based on queryable criteria. For example, it can be known that all instances of a particular class are affected. However, even if that is possible, the query can yield a large set of objects, making the process of updating the permissions of each lengthy. And more typically, there will be no such narrowing query and the administrator will have no option but to scan through all documents and folders, examining the permissions of each to locate those needing to be updated.

The most egregious case is where the task for which X is solely responsible is that of object store administrator, for example, X was the sole principal given in the administrators list when the object store was created. In this case, X was placed in the default instance ACEs for every class defined in the object store. Those ACEs likely have been copied into every instance created of all those classes. Fixing that is, practically speaking, impossible.

► Example 2

User A currently has sole responsibility for a particular task, but now user B joins the department and is to share responsibility with A for that task. As in the previous example, this requires locating and updating all the objects with ACEs in which A appears, adding ACEs granting the same access to B.

This is largely the same as example 1 except that there is a slight possibility of it being made easier by the participation of A, who can (a) be aware of all the objects involved, and (b) can be able to undertake the permissions updates without the intervention of an administrator, if granted WRITE_ACL permission by the ACEs on those objects.

However, it is likely equally as intractable as the first example.

Compare this with how the changes can be accommodated if groups had been used to assign rights to X and A. In each case, it is a matter of adding Y or B to the group (and in the first example, removing X).

5.4.5 Role-based access control using inheritance

The notion of roles in security access authorization is not new. The motivating idea is to be able to describe access policies in terms of generic functional roles rather than administering user access individually. Roles can be thought as a formalization of the idea of areas of responsibility discussed earlier. Therefore, in the example given previously, one might have an Author role and an Approver role.

A basic *role-based access control* (RBAC) model typically consists of three elements:

- ▶ A role definition, specifying a set of access rights associated with the role
- ▶ A member list for the role, identifying the security principals who are authorized to act in that role
- ▶ A set of objects to which that role applies

The overall effect of these three elements is that the members of the role are granted the access rights defined by the role for the objects to which the role applies.

Described in this fashion, it is apparent that roles have much in common with groups and offer similar evolutionary benefits. So why not just use groups? One can, in theory, define a directory group for every interesting role and assign users to the appropriate group or groups. In practice, most enterprises are not happy with that approach for one or more of the following reasons:

- ▶ In large enterprises, the directory is administered by a different group of people than those directly involved with applications.
- ▶ The dominant pattern for directory groups is one that largely reflects the hierarchical organizational structure of the enterprise. That does not align with roles concepts well since roles tend to cut across organizational boundaries.
- ▶ Many applications depend on directory information about individuals, and that tends to lead to a conservative policy about directory changes. Frequent updates, for example, to account for temporary role changes, are specifically avoided.

So, using roles defined “locally” for a particular application provides more flexibility than complete reliance on directory service groups.

P8 does not provide a built-in RBAC mechanism. However, inheritance proxies can be used to achieve much the same effect.

The following technote provides a full explanation of various ways this can be done:

<http://www.ibm.com/support/docview.wss?uid=swg21425080>

The basic concept of using inheritance proxy as described in the technote is as follows:

- ▶ An object serves as the role. Inheritable ACEs are placed in the ACL of the object, one per member of the role, with the same access mask in each granting the access rights defined by the role.
- ▶ For each object class to which the role is applicable, an inheritance proxy-defining property is added, with required class that of the role object. Applying the role to an instance of the class is then simply a matter of setting the proxy-defining property to refer to the role object.

With this approach, adding and removing members for the role are achieved by updating the ACL of the role object. Inheritance takes care of ensuring the effect extends to all the objects to which the role has been applied.

5.4.6 Using markings

The important thing to consider about markings is that they only restrict access, they do not extend it. Therefore, they must be used in conjunction with an effective security descriptor that grants the maximum access anyone needs to have to the marked objects. This is an effective technique when used in the manner of the Projects example given earlier and is an alternative to creating a group for each project or using a role-like scheme. (Changing the membership of the group or role is replaced by editing the ACL for the markings.)

The server holds marking sets in memory (including each marking with the associated security descriptor). A large number of marking sets or markings, or markings with complex ACLs, can require an increase in the amount of memory assigned to the Content Platform Engine server.

Marking sets of list type can contain a large number of markings without performance cost, so using them as in the projects example can scale up to many projects without problems (memory requirements notwithstanding).

However, hierarchical marking sets become increasingly expensive as the number of markings in the set grows, due to the need to traverse up and down the hierarchy when evaluating the effective access to any particular marking. Therefore, they are best suited to uses similar to the classification example given previously, where there are only relatively few markings in the set.

5.4.7 Effective use of auditing

Auditing is a useful tool for detecting security intrusion attempts (operations that are refused on access control grounds) or for recording a history of successfully performed sensitive operations. However, auditing is relatively expensive, both in terms of the cost of the additional processing and I/O to write the audit records and of the storage cost, especially when the option to record a complete snapshot of the object state is exercised. Therefore, it needs to be used judiciously. Consider these guidelines:

- ▶ Usually only certain objects will be of interest, for example, documents with confidential content, so the auditing configuration needs to be narrowed as much as possible to be enabled for just those objects. Often, this is a matter of selectively enabling only certain classes, but in some cases, it can be necessary to use the filtering mechanism to distinguish between interesting and uninteresting objects.
- ▶ For those objects, only particular operations are likely to be of interest, such as viewing the confidential content or attempting to delete critical folders. Therefore, the set of events for which auditing is enabled needs to be chosen carefully. API operations, such as Checkout and Checkin, generate a specific event *and* an Update Event. They will be audited if either Update Event or the specific event (in this example: Checkin Event) is enabled. In consequence, if Update Event is enabled, all API operations will be audited, which can be more than necessary.
- ▶ A complete snapshot of the audited object state consumes several kilobytes of database large object (LOB) storage, and double that if both before and after states are recorded for updates. This can quickly accumulate to consume a large proportion of the available LOB storage if care is not taken. A more economical approach is to use the “audit as” mechanism to capture a limited set of properties from the audited object.
- ▶ A disposition policy should be defined to remove audit records as quickly as possible after they are no longer needed. For example, if auditing is being used for intrusion detection, a potential approach is to regularly generate a report from the audit log and to define a policy that deletes records that are old enough that they are bound to have been included in a report.

5.4.8 Cache management

Several caches were described earlier that are used to improve access control performance. The default configuration of these caches is adequate for many situations, but in some cases, it can be necessary to adjust the configuration of one or more to obtain optimal performance or to tune server memory usage.

For each of these caches, the server maintains operational statistics that are extremely useful as aids to tuning. These statistics can be viewed in the IBM System Dashboard. Each Content Platform Engine server instance has separate caches, so in a multiple server configuration, the statistics for each need to be considered.

User token cache

The user token cache retains recently evaluated security contexts in memory. The size of a security context varies according to the number of groups each user is a member of. Therefore, the overall amount of memory consumed by the user token cache varies by that and by the maximum size configured for the cache. As the cache grows, the server discards the least recently used (LRU) entries to ensure that the maximum size is not exceeded. (This is referred to as an LRU policy and is common to all the caches discussed here.) So, the ideal size for the cache is around the number of users actively issuing requests at any time. More than this means wasted memory consumed by cache entries that are not being used. Less can mean reduced performance due to LRU discarding and reloading by querying the directory service. However, the ideal size can require a significant amount of memory, so there are trade-offs to be made.

Cache entries can become *stale* as a result of changes in group memberships for a user. To account for this, cache entries are retained for a maximum period of time referred to as the *time to live* (TTL). A cache entry whose TTL has expired is refreshed from the directory service to ensure it is up-to-date. However, depending on the length of the TTL, there can be a period of time after group membership changes are made during which the server will continue to use the stale entry. The consequence can be that the user cannot be granted all the access to which they are entitled (if added to a group) or can receive access to which they are no longer entitled (if removed from a group). Therefore, in configurations in which group membership changes are frequent, it can be necessary to reduce the TTL. Conversely, if group membership changes are infrequent, it can be possible to improve performance by increasing the TTL.

Security descriptor cache

There is one security descriptor cache for each object store, independently configured for maximum size with again an LRU policy applied to ensure the maximum size is not exceeded. The size of a security descriptor and therefore the amount of memory consumed by a cache entry is proportional to the number of ACEs in the ACL.

Again, the optimal size for the SD cache is around the number in active use - more leading to wasted memory and less to reduced performance because of the processing time and I/O cost of retrieving security descriptors from the database. However, sharing of security descriptors and other factors such as the frequency of proxy relationships among objects being accessed makes it difficult to estimate what the optimal size will be. So, using the performance statistics available in the Dashboard is the best approach to tuning. The size is independently configurable for each object store since the patterns of access to objects, and therefore to security descriptors, can differ between object stores.

There is no TTL for security descriptor cache entries since they are never modified and can never become stale.

Object-security cache

The object-security cache retains security information for proxy objects in memory, improving performance when the same proxy is accessed several times over a short period of time. An example is when retrieving the subfolders of a folder (each subfolder has the same proxy, namely its parent folder), so the security information for that folder is fetched just once and is reused to evaluate the access rights for every subfolder.

There is an independent object-security cache for each object store, again reflecting the possibility of different patterns of access, in this case, to proxy objects. For each object store, the maximum size and TTL for the cache are configurable.

The information retained in an object-security cache entry consists of a reference to the security descriptor for the proxy object (not the SD itself) plus identity information for the objects referenced by any proxy-defining properties of the proxy itself and the values of any marking-controlled properties of the proxy object. This typically requires a smaller amount of memory than for entries in the other caches, giving more scope for increasing the size without risk of running out of memory. Like the security descriptor cache, estimating the optimal size of the object-security cache is not straightforward and using the Dashboard statistics is the best approach for tuning.

An object-security cache entry can become stale as a result of updates to security-related properties of the proxy object - changes to Permissions or Owner or to proxy-defining or marking-controlled properties. The server that receives the request to make these updates automatically flushes the now stale entry from its cache, but other servers rely on a TTL as the means of overcoming staleness. The default TTL for the object-security cache is relatively short (30 seconds). In a multiple server configuration, it should only be increased if updates to any proxy objects are known to take place much less frequently than that.



Application design

In this chapter, we discuss useful principles for designing IBM FileNet Content Manager (P8 Content Manager) applications.

We discuss the following topics:

- ▶ IBM FileNet P8 applications
- ▶ Application technologies
- ▶ Principles for application design:
 - Available P8 Content Engine APIs
 - Transports available with the APIs
 - Minimizing round-trips
 - Creating a custom AddOn
 - Exploiting the active content event model
 - Logging

Note: Although the technical components that make up the server pieces are called the Content Platform Engine, there are still many separate aspects, including APIs, for content and process as of the writing of this book. For clarity, we use the earlier term “Content Engine” when talking specifically about content matters in this chapter.

6.1 IBM FileNet P8 applications

P8 Content Manager includes a number of standard applications, and many more applications are available as add-ons to the basic product. The applications are aimed at different audiences and use cases. In this section, we introduce a few of the applications to serve as examples for application development.

6.1.1 IBM Administration Console for Content Platform Engine

The IBM Administration Console for Content Platform Engine (ACCE) is a powerful tool for administrators to use in performing routine setup, maintenance, and specialized tasks. ACCE is implemented as a rich web application sharing much infrastructure scaffolding with IBM Content Navigator.

Because it is an administrator's tool that can be used for extraordinary and powerful low-level changes, ACCE strikes a balance. It exposes low-level details of the IBM FileNet Content Manager, yet it remains usable through extensive task wizards and other user interface help.

Although ACCE is an administrator's tool, it uses the normal Content Engine APIs, and you are subject to normal security access checks. The administrator running ACCE typically has a high level of security access, but ACCE does not and cannot provide any additional privileges. It therefore serves as a good example of the actions that can be done with custom applications that also use the Content Engine APIs.

Recommendations: ACCE is a replacement of an earlier thick client tool called IBM FileNet Enterprise Manager. IBM FileNet Enterprise Manager is still shipped because of the familiarity many administrators have with it. However, the primary administrator tool is ACCE, and you need to get familiar with it in preference to IBM FileNet Enterprise Manager. New features will only be added to IBM FileNet Enterprise Manager as exceptions.

6.1.2 IBM Content Navigator

In contrast to ACCE (see 6.1.1, "IBM Administration Console for Content Platform Engine" on page 192), IBM Content Navigator is intended for the wider audience of non-administrator users. Even though it is generic in nature, it still provides a comfortable and productive user interface for accomplishing a variety of everyday tasks. IBM Content Navigator is a rich web application. The user interface uses modern Web 2.0 and Ajax technologies to closely model a desktop application experience. It provides easy-to-use windows and wizards for

navigating and searching for documents and folders. IBM Content Navigator can connect to other IBM content repositories and is included as the standard client in several IBM ECM products.

In addition to being a ready-to-use client application, you can also easily customize IBM Content Navigator user interface elements or extend it with entirely new features. See “IBM Content Navigator extensions” on page 199.

Recommendations: You might have experience with using and extending earlier generations of IBM FileNet Content Manager web clients, including Application Engine, Workplace, or FileNet Workplace XT. For any new client application development work, you need to strongly consider basing it on IBM Content Navigator.

6.2 Application technologies

Content Manager comes with a set of applications that you can use as is. The operations and interfaces provided by these applications might not always satisfy your enterprise’s business requirements. In many circumstances, you have to create custom applications to fulfill your business needs. Your applications will be designed with specific business goals in mind, and those come in many varieties. We do not attempt to cover business goals here. Instead, we discuss more general technical application technologies.

6.2.1 Traditional Java thick clients

Content Manager’s Content Platform Engine consists of Java Platform, Enterprise Edition (Java EE) components, but your client application can be a Java thick client. By *thick client*, we mean an application running in its own Java virtual machine (JVM) launched from the desktop. It can be a simple command-line program or have a full-featured graphical user interface. Because it is launched from the local client machine, there are virtually no security restrictions on what a thick client application can do.

A thick client application normally consists of directories or Java archive (JAR) files of Java classes, both for the application and for supporting utility libraries. One of the biggest problems in using thick clients is the logistical hurdle of keeping all of the copies of the application up-to-date. This trait is not unique to Java applications; it is the same for any thick client technology. Because of this problem, however, thick clients are best suited for use by a small number of users or for mature and stable software.

Recommendations: Limit the use of thick client applications to exploratory code or utilities with a limited user population.

6.2.2 Java applets

A specialized form of thick client is a Java application that runs inside a security-constrained Java environment in a web browser. This application is called an *applet*. An applet can have most of the rich interactions of a traditional thick client, but it has advantages and disadvantages.

The most obvious disadvantage is that the users must run a Java capable web browser, and the use of Java applets must be enabled. All major web browsers are Java capable, but for security reasons, organizational policies sometimes forbid enabling the running of Java applets.

Recommendations: Avoid the use of Java applets. For most needs, modern JavaScript toolkits and Ajax technologies can serve just as well and have fewer deployment problems.

6.2.3 Java EE web applications and other components

The technology underlying much of the enterprise software development these days is Java Platform Enterprise Edition (Java EE). Java EE helps you make efficient use of resources by providing common services, such as security, high availability, transaction management, and scalability. Because the platform provides these services with mechanisms for configuring them when the applications are deployed, you are free to concentrate on business logic in your applications. The Content Platform Engine, which is implemented as Java EE components, uses many common features of Java EE. You can write Content Platform Engine applications with traditional thick client Java applications or even non Java client technologies, but the tightest integration will naturally be available when your application is integrated with a Java EE application server.

There are many standardized technologies available in Java EE, but a few technologies are particularly worth mentioning because they often show up in typical Java EE application development: servlets and Enterprise JavaBeans (EJBs):

- ▶ The Java EE servlet container is often thought of as the container for web applications because it represents the tier where Java EE presentation logic is generally placed. Web applications are perhaps the most popular use for servlets, but it is not necessary to have an actual web interface to use

servlets. For example, the IBM Content Management Interoperability Services (CMIS) provider is implemented using a servlet, and any user interface is provided by the CMIS client applications. The servlet container is appropriate for application components that receive and respond to outside requests and that optionally preserve some state on the server side between requests.

- ▶ The Java EE EJB¹ container provides what are often thought of as enterprise-level services. For example, EJBs can have declarative security and transactional properties, provide transparent load balancing across servers, and provide nearly transparent access to relational databases. EJBs are frequently used to encapsulate reusable business logic and seldom, if ever, contain any presentation logic.

In recent years, web services have expanded and matured. That has blurred the line between what needs to be implemented in the web tier and what needs to be implemented in the EJB tier.

6.2.4 .NET components

Just as the Java community has standardized on Java EE as a software component architecture, Microsoft has popularized the .NET environment. .NET shares many concepts with Java and Java EE, but, from the point of view of the Content Platform Engine, only clients can be written using .NET technology. .NET is fundamentally incompatible with Java and Java EE except when interacting via a common protocol. In Content Manager, the common protocol is the web services transport of the IBM FileNet P8 Content Engine APIs or the direct use of IBM FileNet P8 Content Engine Web Services.

6.3 Principles for application design

In this section, we present principles to consider when designing your own applications. Obviously, situations vary, and not all of these principles apply to every situation. Our intention is to give you a brief survey, which will have a bearing on your designs and that might even suggest new application designs to you.

¹ Although the Content Engine Java API uses EJBs internally to implement the EJB transport, those EJBs are not exposed or available to application developers. They are accessible only indirectly through the use of the Java API.

6.3.1 Available P8 Content Manager APIs

One of the goals of the Content Manager is to make all features available through robust APIs. Content Manager applications add their own utility layers, often with significant amounts of application logic, but interaction with the server always comes down to a set of calls to published and documented APIs. Those APIs are also available to you for custom application development. If you see a feature in a P8 Content Manager application, you can be confident that your custom application can do the same or similar things via the APIs.

This section describes the APIs available in the IBM FileNet P8 4.0 release and later. We do not discuss the *compatibility* APIs (for Java and COM) that exist to help in the transition of applications written for earlier P8 Content Manager releases. Both of those compatibility APIs are now deprecated, but no date for their complete removal has been announced as of this writing.

Recommendations: Use the current IBM FileNet P8 APIs for any new development and, where possible, for additions to existing applications. Avoid extending your use of the compatibility APIs any longer than necessary.

Many specific details are treated lightly in this section. That is intentional because there is a separate IBM Redbooks publication, *Developing Applications with IBM FileNet P8 APIs*, SG24-7743-00, that provides in-depth descriptions, details, and illustrative code samples.

Java API

Content Manager provides a full-featured Java API. Any feature that is available in the server is completely available to Java programmers. This access includes routine operations, such as retrieving and updating Document objects, and specialized operations, such as adding a custom class or property to an object store's metadata definitions.

In simplified terms, an API object can be thought of as containing the following information:

- ▶ Something that identifies the object residing on the server. Typically, this is an object store reference and an object ID or path.
- ▶ Some number of locally cached properties. These might have been fetched from the server, or they might have been set locally. A property value that has been set or changed in the API object and not yet sent to the server is said to be *dirty*, because its value does not match what is persisted on the server.

- ▶ Some number of pending actions. When you call a method that implies a change to the object (including simple property value changes), the change is not made immediately. Instead, a representation of that change is added to the API object's list of pending actions. For example, if you call the method `Document.checkin()`, a `Checkin` pending action is added to the API object.

Dirty property values and pending actions are not sent to the server until an explicit call is made to do so. If an API object is discarded without that call, the changes are never made on the server. The most common method of sending changes to the server is to call the `save()` method on an API object. There is also a batching mechanism for sending updates to multiple objects in a single round-trip over the network. Batching provides improved performance and provides transactional atomicity for all of the changes in the batch.

Recommendations: Use only exposed and supported classes and interfaces in the API. Do not use internal implementation classes; in particular, do not make calls into anything in the `com.filenet.apiimpl.*` packages.

.NET API

Content Manager provides a full-featured .NET API, which you can use to write programs in any .NET compatible language. With a couple of exceptions, any feature that is available in the server is completely available to .NET programmers. The exceptions are mainly custom code that must be executed within the server, for example, `EventHandler`. Because the Content Platform Engine server is a Java EE application, internally executed custom code is limited to Java compatible technologies.

The principles behind the .NET API are the same as those behind the Java API (see “Java API” on page 196), so we do not repeat that discussion here. One significant feature available only with the .NET API is the use of Kerberos to perform authentication via Microsoft Windows Integrated Login. This is only possible when the client application is running on Microsoft Windows and the Content Platform Engine is using Microsoft Active Directory. In practice, that latter constraint usually means that the Content Platform Engine is also running on Microsoft Windows.

Recommendations: Use only exposed and supported classes and interfaces in the API. Do not use internal implementation classes; in particular, do not make calls into anything in the `FileNet.Apiimpl.*` namespaces.

Web services

Modern, loosely coupled frameworks, such as a service-oriented architecture (SOA), favor web services protocols for connecting components. Content Manager provides Content Engine Web Services (CEWS) for accessing nearly all features available in the Content Engine server.

Typically, if you as a programmer want to use a web services interface, you obtain the interface description in the form of a Web Services Description Language (WSDL) file. You run the WSDL file through a toolkit to generate programming language objects for interacting with the web services interface. You then usually build up a library of utilities to provide abstraction layers, caching, security controls, and other conveniences. The Java and .NET APIs provided by Content Manager are already exactly equivalent to that, and both APIs can use web services as a transport (see 6.3.2, “Transports available with the APIs” on page 200). Consequently, there is not as much motivation to use CEWS directly, although there are still a few occasions where the direct use of CEWS might be useful:

- ▶ You have an application already using CEWS, and no plans exist for immediately porting it to the Java or .NET API.
- ▶ You are building an application component as part of a framework in which the use of web services is the model for communicating with external systems.
- ▶ Although a rare occurrence, you might be using a language or technology that can make use of web services but is not compatible with the use of a Java or .NET API.

For these occasions, the direct use of CEWS is a good choice and is supported.

In theory, you can take the WSDL file for CEWS and use any current web services toolkit to generate the interfaces that you will use on your end. In practice, however, toolkits are still individualistic in their handling of various WSDL features, and it is difficult to write a WSDL for a complex service that is usable by a wide cross-section of web services toolkits. Check the latest hardware and software support documentation corresponding to the product version you are using, and use only a supported toolkit.

Recommendations: To decide which API to use to implement your application, follow this approach:

- ▶ If you are writing handler code that runs inside the Content Platform Engine server, it must be written in Java or a supported scripting language.
- ▶ If you have a case where you must use CEWS, use it. However, if you can possibly avoid using CEWS, avoid it.
- ▶ If your development organization has more experience in .NET or Java, choose the corresponding Content Engine API.
- ▶ If it is still a toss-up, choose Java (because of greater flexibility in API transports that you might find handy later).

We intentionally do not list performance as a way to choose the API since it really is dominated by the other factors.

Content Management Interoperability Services

Content Management Interoperability Services (CMIS) is an industry standard for accessing content repositories and performing mainstream document management tasks. It defines a set of REST and web services interfaces and has fundamental constructs for documents, folders, and properties. IBM has created a CMIS provider for IBM FileNet Content Manager that is included in the Content Manager license (although it is a separately installed item).

Many third-party components include the ability to use CMIS to connect to content repositories. In such cases, your integration work can be relatively straightforward and simple.

CMIS is also well-suited for use in scenarios where you might already be considering using a REST or web services interface for mainstream document management functions. Instead of having to write those services yourself and interfacing to Content Manager repositories by using Content Engine APIs, you can instead use the CMIS provider as the service layer. There are readily available vendor and open source toolkits to help you construct the client side to communicate with a CMIS service layer.

IBM Content Navigator extensions

IBM Content Navigator is the current generation user client for all IBM content repositories. It builds on the extensive experience gained from previous generations of user clients across several product lines.

A key strength of Content Navigator is that it was built with extensibility in mind. We expect that a large percentage of application developers will find it useful to

start with Content Navigator and then customize and extend it to meet their custom application needs.

By customization, we mean altering the visual appearance or behavior of an existing Content Navigator component. By extension, we mean adding new features, large or small, to an Content Navigator environment.

Content Navigator is a browser-based web application. Its layered architecture consists of these components:

- ▶ A collection of visual widgets written using the Dojo JavaScript toolkit and the dijit component libraries.
- ▶ A layout framework for arranging visual components into logical desktops and pages.
- ▶ A browser-resident JavaScript model view controller (MVC) layer for orchestrating the flow of information.
- ▶ Mid-tier server-based components for interfacing to repositories and providing other services.

Each of those layers has available customization and extension points for application developer use.

Recommendations: Use Content Navigator as your application framework for content-centric applications that need a rich and modern user interface. Extend Content Navigator with features you need that are not already part of Content Navigator.

In addition to being a highly customizable and extensible application framework, most of the visual widget components used in the user interface layer can also be easily adapted for use outside of the Content Navigator environment. The reuse of those widgets can represent considerable development time savings even if you do not choose to use Content Navigator itself.

All of the customization and extension topics in this subsection are covered in extensive detail in *Customizing and Extending IBM Content Navigator*, SG24-8055.

6.3.2 Transports available with the APIs

When designing any multi-tiered application, you must carefully consider how information will be conveyed back and forth between the client side and the server side of the network connection. Different frameworks for remote calls typically come with different advantages and constraints.

In the Content Engine APIs, the framework mechanisms are called *transports*. The APIs were designed so that all API operations are completely independent of the transport used. (The few exceptions deal with the propagation of security and transaction contexts.) A benefit of this independence is that applications can be written without considering the transport. The selection of a transport is a configuration decision when the application is deployed (the API finds out about it through the URI used for the Connection object).

There are two available transports: Content Engine Web Services (CEWS) and Enterprise JavaBeans (EJB). EJB transport is available only for the Java API. CEWS transport is available for both APIs. For most situations, the EJB transport has slightly better performance, but the CEWS transport can be used in more environments. In all cases, the transport is considered *stateless*, which means that the APIs operate on the basis of a single request and response for each interaction. No client state is maintained by the server after a request has been serviced. There is one exception to the statelessness, which is that recent releases of the Content Engine Java API can be configured to use a stateful session bean when uploading multiple chunks of content over EJB transport.

EJB transport

The EJB transport internally uses EJB method calls. The method calls are made on the client side and transported by the application server to the server side of the network connection. Although many people think of EJBs using Java Remote Method Invocation (RMI) as the remote communications mechanism, that is not necessarily the case. Application server vendors are free to provide whatever implementation they like as long as they meet the EJB requirements, and many vendors use something other than RMI. In any case, the details of the application server's implementation are transparent to the API, and the API does not need to have facilities for controlling things, such as clustering or server affinity of the EJB, because those things are configured within the application server.

CEWS transport

As its name implies, the CEWS transport uses web services protocols. In fact, the WS transport uses the same Content Engine Web Services (CEWS) protocol that we mentioned in “Web services” on page 198. You probably already know that means XML over HTTP or HTTPS. Because HTTP and HTTPS use only a single port for the entire conversation and use a strict client/server interaction model, it is generally easier to configure a firewall or reverse proxy through which to allow CEWS transport requests to pass.

Web services attachments are used for carrying pieces of content between the client and server sides. Attachment handling has undergone many changes over the years, and different environments and tools support different standards:

- ▶ When using either API, you must select the CEWS endpoint that supports Message Transmission Optimization Mechanism (MTOM) attachments (recognizable because it has MTOM in the endpoint name: FNCEWS40MTOM).
- ▶ There is another attachment format called SOAP that is less efficient in a couple of ways than MTOM. Nonetheless, it is sometimes useful to temporarily use the SOAP endpoint (FNCEWS40SOAP) as a troubleshooting step if you suspect problems at the transport layer. That is seldom actually the case, but it does not hurt to rule it out.

Comparing the transports

Consider the following information when deciding which transport to use:

- ▶ Because it usually employs a binary protocol likely to have been engineered for high performance, the EJB transport typically has better performance than the CEWS transport in the same environment. In particular, processor utilization is likely to be a bit higher with CEWS transport due to XML parsing activity. The actual performance difference is extremely dependent upon the specific mix of API calls your application makes.
- ▶ The EJB used by the EJB transport automatically propagates any active transactional context to the server. In contrast, transaction propagation is not possible when using CEWS transport. Whether transaction propagation is desirable depends upon the application. The Content Platform Engine always treats incoming client requests transactionally, so most applications do not need to worry about it at all.
- ▶ The EJB used by the EJB transport automatically propagates any ambient JAAS authentication context to the server. If you are already using a JAAS-based authentication scheme, either in isolation or as part of a single sign-on (SSO) framework, Content Manager is likely to participate in that scheme with few or no configuration changes if you use EJB transport.
- ▶ In contrast, there is no general framework for propagating an authentication context when using WS transport. Although a standard called *WS-Security* provides a high-level framework for adding authentication schemes, CEWS transport can only support schemes backed by specific implementation programming in the Content Platform Engine server. Content Manager directly supports WS-Security Username token and Kerberos token authentication schemes. The latter can be used to facilitate integration with Microsoft Windows applications. Custom authentication schemes can also be implemented by using the IBM FileNet Web Services Extensible Authentication Framework (WS-EAF).

Specific details of using Kerberos and WS-EAF are provided in the *Web Service Extensible Authentication Framework Developer's Guide* section of the online help files, *IBM FileNet P8 Documentation*.

- ▶ CEWS transport, which is based on HTTP or HTTPS, uses just one or two TCP/IP ports for all interactions. There are also commercially available products for examining and validating web services traffic. Therefore, many administrators find it easier and more secure to open their firewalls to CEWS transport requests. In contrast, EJB transport might use a vendor-specific binary protocol. Such protocols often employ a range of TCP/IP ports. These factors typically lead to a greater willingness to allow CEWS transport to pass through firewalls and a reluctance to do the same for EJB transport.
- ▶ In cases where WS transport is using Username token authentication, the credentials will appear on the wire unprotected unless you use Transport Layer Security or Secure Sockets Layer (TLS/SSL), which we strongly advise.
- ▶ With EJB transport, content is uploaded or downloaded in chunks. With CEWS transport, the entire content is uploaded as part of a single HTTP request. For download, however, CEWS transport also generally chunks content.

Note: It used to be recommended to use CEWS transport for upload of large content. However, recent releases have included some optimization work using a stateful EJB call when uploading content chunks. That translates directly to less work needed on the Content Engine server side once the chunks have been uploaded. Although EJB transport still chunks content on both upload and download, the performance overhead of the chunking itself is typically quite small. Do not use the presence of large content as your sole reason for selecting a particular transport.

Recommendations: To decide which transport to use, follow this approach:

- ▶ If you are using the .NET API, you must use the CEWS transport.
- ▶ If you are using the Java API and need one of the features that is only provided by EJB transport (security or transaction context propagation), use EJB transport.
- ▶ If you are writing a Java application that is hosted in a Java EE application server, it is generally easier to configure EJB transport.
- ▶ However, EJB transport is only supported between homogeneous types of Java EE application servers on the client and server. So, if you have heterogeneous types of application servers, you must use CEWS transport.
- ▶ If you are writing a Java thick application, it might be easier to configure CEWS transport.

These considerations are mainly about the simplicity of runtime configuration and deployment. For almost everything, your application coding is exactly the same regardless of transport.

6.3.3 Minimizing round-trips

The number and nature of network *round-trips*, that is, requests from the client to get a response from the server, usually dominate the performance picture of the application. There are simple and powerful tools available in the APIs to reduce your round-trips, and API logging can be used to assess how well you are doing.

Recommendations: When developing your application, allow some time in the schedule to examine the working application for opportunities to eliminate round-trips. That can sometimes be done with simple code tweaks, but it will sometimes require a bit of refactoring of your logic.

Get or fetch

When many people think about interacting with an object from the server, they first think about doing a round-trip to fetch the object. That is a necessity for many things, but there are several cases where you do not need that initial fetch. For example, if you are only going to use an object so you can set the value of an object-valued property on another object, you really only need a reference. If you somehow know that the object already exists, you can skip the round-trip to fetch it.

(If it turned out that you were wrong and it did not already exist, the referential integrity mechanisms in Content Engine will throw an exception when you try to save the referencing object.) The APIs have a mechanism called *fetchless instantiation*. There are three types of Factory methods for creating programming language objects that reference Content Engine objects, and you can tell them apart by the word used as the *beginning* of the method name:

- ▶ `create` indicates that a new Content Engine object is to be created. No round-trip is done as the result of this Factory method call. A `save()` call must eventually be done.
- ▶ `fetch` indicates that a round-trip is immediately made to the Content Platform Engine to verify that the object exists and to return an initial set of properties. Fine-tuning of the properties returned can be controlled via an optional `PropertyFilter`. See “Property filters” on page 205.
- ▶ `get` indicates that no round-trip will be made. This is a fetchless instantiation. The API assumes that the object exists. There is no initial set of property values available, so you need to request any property values that you need. If you know that you always need some property values immediately, there is no advantage to fetchless instantiation.

Property filters

Property filters are optional parameters to a number of methods that fetch objects or properties from the Content Platform Engine. They allow highly granular control of the objects or properties being returned.

It is easy to understand how returning fewer properties can improve performance, but, less obviously, you can also improve performance by returning more properties and objects. The savings comes if you can return multiple objects in a single round-trip instead of making multiple round-trips to perform the same work. A property filter can do just that. Over time, most application developers know what properties and objects they need, so this can be an efficient way to perform most or all of your retrievals in just a few round-trips.

Most of the Content Engine API calls that can take a property filter also accept a null value. In these cases, the API still works correctly, but it might make additional round-trips in the background as your application progresses. It is designed that way so that you can get your application working quickly and optimize the performance later.

Batching

The Content Engine APIs contain two separate but similar batching mechanisms:

- ▶ A `RetrievingBatch` is used to fetch multiple, possibly unrelated, objects from the Content Platform Engine in a single round-trip. Object references and property filters are added to the batch, and `retrieveBatch()` is called to trigger the round-trip.
- ▶ An `UpdatingBatch` is used to group multiple updates in a single round-trip to the Content Platform Engine. Instead of calling `save()` on individual objects, the objects are added to the batch, and `updateBatch()` is called to trigger the round-trip. Updates are performed as an atomic transaction.

Recommendations: Unless it leads to tortured application logic, it is a good idea to accumulate multiple changes to objects before calling `save()`, and it is also a good idea to batch updates to multiple objects in an `UpdatingBatch`.

As a general rule, plan to carry no more than 50 - 100 items in a batch. Somewhere in that range, the overhead associated with batching itself tends to neutralize any performance benefits. Since specifics of application workload can change for various reasons, consider making the batch sizes configurable so that a code change is not needed for that adjustment.

6.3.4 Parallel processing

With any client/server application arrangement, there is likely to be a significant amount of time where the client is simply waiting for a response from the server. If your application is handling a large workload, it might benefit from being split up into a number of parallel work items. Not every application activity is amenable to that sort of splitting, but many are or can easily be adapted.

The usual way of splitting up work is to use multiple threads inside a single process, but it is sometimes adequate to simultaneously execute multiple single-threaded processes. Depending on the nature of the work being done, it might be necessary to have a single overall coordinator thread or process that dispatches specific work items to worker threads or processes. In other cases, it is possible to assign each thread or process a specific piece of work in its start-up parameters.

Recommendations: When splitting the application into multiple threads or processes, make the number of these threads or processes configurable. That eliminates the need for a code change if you discover that the optimal number of threads or processes changes over time.

There is a recurring application pattern that involves issuing a query for objects matching a particular criteria and then performing an action on each result object. The issuance of the query and accumulation of results are good jobs for the coordinator thread or process. Disjoint sets of result objects can be handed to worker threads or processes for action.

Alternatively, you might have an application that must process a large number of objects, but your performance constraints are to operate as a background task. That is, you want the processing to move forward, but you do not want to interfere with foreground work by placing an undue load on the server machines. In that case, single-threaded processing might be a better match. In some cases, you can easily distinguish the already processed objects from those still needing processing. For example, your criteria might include some property value being null, and your action might include setting that property to a non-null value. In such cases, you can use a non-continuable query instead of a continuable, paged query. Non-continuable queries have lower server overhead than continuable queries. Just be sure to include a TOP qualifier to the SELECT clause, for example, "TOP 50". The number that you use can be convenient for the batch sizes that you plan to use for the actions.

Recommendations: When the semantics of iterative processing allow it, use a non-continuable query for best performance. This approach generally does not work when multiple threads or processes perform the update actions in parallel.

6.3.5 Client-side transactions

All work performed by the Content Engine in a database or other storage is done transactionally, which means that you never get partially successful calls to Content Engine. The call either completely succeeds or completely fails. This is important for maintaining the consistency of the data in the repositories. You do not need to do anything to get that sort of transactional behavior inside Content Engine. Actually, there is no way to avoid it, because it is hardcoded into Content Engine logic.

There is another type of transaction that you can control in your application. If you use the Java API with EJB transport, you can include Content Engine activity within a client-side transaction. This feature is unavailable when using CEWS transport. (See "CEWS transport" on page 201.) The client-side transaction can be started implicitly by the Java EE container or started explicitly through your use of a `javax.transaction.UserTransaction` object.

Content Manager follows the Java EE model for transactions, and Java EE in turn follows industry standards for distributed transactions. In this context, the relevant facts are that a transaction is started, operations performed by a transactional resource (in this case, Content Engine) are tagged with the transaction identifier, and the transaction is either committed or rolled back. All changes tagged with a certain transaction identifier are committed or rolled back as an atomic unit.

Now that we have described the use of client-side transactions, here are a few reasons to avoid them:

- ▶ Client-side transactions tend to create or magnify performance problems. The overall transaction times are longer simply due to network latency and other factors inherent in the interaction between client and server. Longer transaction times mean that resources all the way into the database are being held for longer periods of time. This greatly increases the chances for resource contention and slows overall system throughput.
- ▶ Most of the tasks that applications want to do in a client-side transaction can be done more efficiently with the API batching mechanism using an `UpdatingBatch` object. A batch is performed as an atomic transaction, but the transactional control is on the Content Platform Engine side.
- ▶ API batches can be used with all APIs and transports, so it is a more flexible mechanism than client-side transactions.

After some analysis, it almost always turns out to be the case that applications using client-side transactions can be rewritten to use API batching. For the few cases where client-side transactions are genuinely needed, they are supported as described. The case where you might be forced into a client-side transaction is when your application must include transactional resources outside of Content Manager. For example, if you must include P8 Content Manager updates atomically with updates to a stand-alone database, that is a motive for using a client-side transaction. If you find yourself using a client-side transaction that you cannot avoid, do your best to minimize the amount of time that the transaction is active.

Recommendations: Avoid using client-side transactions. Instead, rely on the inherent transactional behavior of the Content Engine server.

6.3.6 Creating a custom AddOn

If you plan to use your application in multiple environments, either in your own organization or by distributing it to others, you need to be able to re-create the classes, properties, and perhaps some instance data from your repository.

We discuss the process of moving from development to production environments in Chapter 9, “Deployment” on page 271. For situations where you want to deliver your application as a package, you can consider developing an *AddOn*. An AddOn is a bundle of exported data with optional pre-installation and post-installation scripts. The scripts are run automatically before or after the AddOn is installed. The scripts can be used for any programmatic activity that you need to customize the data in the target environment. An AddOn also has information about other AddOns that must be installed as prerequisites.

An AddOn is *created* by creating an instance of the Content Engine AddOn class. When saved, the AddOn is stored within the global configuration database (GCD). Available AddOns are accessible via the Domain object’s AddOns property. An available AddOn can then be *installed* into an object store, which means that the data is imported and the post-installation script is run. IBM FileNet ACCE has menu actions and wizards for manipulating AddOns, including selecting which AddOns to install when an object store is created.

6.3.7 Using the JDBC interface for reporting

In addition to programming language APIs, P8 Content Manager also presents a read-only Java Database Connectivity (JDBC) interface. This interface is not an interface directly to the relational database tables used in the repository. Rather, it is a view into the object model represented by the Content Engine metadata. In the JDBC interface, queries follow a model analogous to that of the native APIs, where each metadata class looks like a database table and each property looks like a database column.

The JDBC interface follows the JDBC specifications and programming models, but the motivation for its development was primarily for use by reporting tools. The JDBC interface is also purely read-only. Therefore, the JDBC interface is not a good choice for use in application development. For general application programming, the native APIs provide a richer interface.

Recommendations: Avoid the use of the Java API JDBC provider except for integrating with off-the-shelf report generation packages and similar products that require a JDBC interface. For developer-written code, use the facilities of the Content Engine .NET or Java API. If you have an administrative need to perform reporting and counts that cannot be done in a performant way using the APIs, you might need to query directly against the underlying database.

6.3.8 Exploiting the active content event model

Content Manager provides a unique *active content* capability that proactively moves content and content-related business tasks through a business process without requiring human initiation. You probably have several objects that are mostly directly controlled by your application, but you also want to be aware of it if another application tries to make a change to these objects. When that happens, you might want to either prevent the change or perform follow-up actions to ensure data consistency in an application-specific way. One well-known follow-up action is to launch a workflow activity so that an affiliated IBM Case Foundation system can coordinate a complex chain of events.

As a programmer or an administrator, your exposure to active content is via the Content Manager's event subscriptions model. You create and register a subscription for various events. The subscriptions can be created for individual object instances or for an entire class of objects. The subscribed events represent updates (or at least attempted updates) to an object.

When an event occurs in Content Engine, any active subscriptions link the event to an `EventAction` and ultimately to your code. Your code receives parameters that describe the event that occurred as well as the state of the object when the event occurred. For some events, you get both before and after snapshots of the object.

Event subscriptions come in three types: *change preprocessor*, *synchronous*, and *asynchronous*. It is up to you as the creator of the subscription to decide which type to use:

- ▶ For a change preprocessor, which happens synchronously as an update request arrives at the Content Platform Engine server, your handler is allowed to make simple changes to the incoming object before it is passed along to the main part of the server. A change preprocessor runs under the security context of the original calling user.
- ▶ For a synchronous event subscription, your event action handler is called after the change has been made to the object, but before it is committed (in the transactional sense). You are not allowed to make changes to the object, but you have the opportunity to veto the change by throwing an exception.
- ▶ For an asynchronous event subscription, your event action handler is called after the change to the object has been committed to the database. Your handler does not run within the context of the original transaction of the update request. Instead, it has its own transaction started by Content Platform Engine. You can make changes to the triggering object, but those changes are just normal, additional changes as you might make from a client program. Your handler cannot veto the original change, because it has already happened and been committed.

Recommendations: The event model is powerful and can be a useful component of your overall solution design. Become familiar with the three types of handlers. Do not be tempted to violate the usage rules for the three types. Even if something non-compliant happens to work in a simple test, it can fail later in mysterious ways.

By using the event subscription model in Content Engine, you can create handlers that monitor changes to objects not just from your application or components, but from all sources.

6.3.9 Logging

The Content Manager APIs and the server have built-in logging that focuses on providing details of round-trips between the client and server. The reason for that focus is because those details are typically interesting information for resolving both performance and functional problems. The main purpose of the logging is to have artifacts for diagnosing problems when hands-on debugging is not possible. Those logs are intended to be examined by IBM Support and development engineers. They are not documented in detail, but you might easily develop an informal familiarity with them if you work with them.

When designing logging for your own applications, you are likely to have similar goals. You might want to consider the following points:

- ▶ Determine the interesting interactions in your application. Focus your logging efforts on those interactions first. You can always add more logging as your application evolves or as you become more familiar with the types of problems that occur in production. Think of logging those interesting interactions as a unit, whether they are all contained within a certain software module or not.
- ▶ Do not log uninteresting details. Logs can become quite large, and many details that are logged turn out to be distracting clutter when you are looking at log files later. If something is likely to help solve a problem, log it. If there is just a remote possibility that it will help, skip it.
- ▶ Be careful about tying things to source code. It is fine to assume that the people looking at the logs will have access to the source code to see what entries mean, but only do that if that is actually true. Otherwise, log entries must be reasonably self-explanatory so that you can teach someone what they mean.
- ▶ Log the impossible. In any application, there are conditions that are supposed to be impossible. It is tempting to silently ignore those conditions in program logic. If one of those conditions actually happens, it must be logged, because

it is an indication of a design flaw or something seriously strange in the runtime environment.

- ▶ Pick a few severity and verbosity levels. It is probably better to have fewer rather than more levels of granularity in your controls for logging. Modern logging toolkits often give you the freedom to control things with many levels. Do you really need them all? You probably do not. You probably do not need much more than “on”, “off”, and perhaps one level in between. For each combination, ask yourself who will really use it and why it is better than another combination that you already need. One reason to have an intermediate level is because voluminous logging usually has an impact on performance. You can sometimes get ideas for narrowing your focus by using only intermediate logging.
- ▶ When logging error conditions, log the entire exception message and stack trace, including any nested exceptions. Some people consider it a security risk to display this information to users, but this is not a problem for logs seen only by administrators.
- ▶ Make it possible to reconfigure logging dynamically without restarting the application. Some logging toolkits have this capability built in. If they do not, code your application layer so that it periodically checks for a change.

Recommendations: The 5.2 release introduces a new Java class, `HandlerCallContext`, with several logging-related methods. That class is intended for use by event handler code and other custom code that runs inside the Content Engine server. Use those logging methods so that your logging output is integrated with trace and error logging from the Content Engine server itself. Enable it with the Handlers trace logging subsystem in ACCE.

6.3.10 Creating a data model

Designing applications goes along with the design of how you plan to store your permanent data. In P8 Content Manager, the available mechanisms in the repository fully support your use of object-oriented programming models. We describe here just a few of the items that might be overlooked by developers unfamiliar with an object-oriented persistence layer. In certain cases, there are features that are not commonly available in object-oriented programming languages.

Inheritance

Repository classes support a convenient inheritance model. You can define new subclasses that add properties or change various characteristics of existing properties for the subclass.

You can also add new properties to most system classes, although it usually makes more sense to define a subclass just for that purpose and extend it (by adding properties or further subclassing) for your application's needs.

Property value constraints

The repository metadata model also allows you to define default values and constraints on properties that will be enforced by the server. For example, you can define an integer property constrained to a specific range or set of allowed values. Although you might traditionally put that validation logic into your application, having it in the metadata ensures that no other application can put invalid data in those properties. After the constraints are in the metadata, your application can read the metadata and use that to guide application layer validation.

Object-valued properties

One of the more powerful features of the data model is object-valued properties (OVPs). When one object needs to reference another object, use OVPs instead of storing the ID or path to the object. By using OVPs, you can directly navigate from object to object. For an OVP, the metadata provides type safety by only allowing you to point to objects of a certain class (or subclass), just like an object reference in a programming language. The server provides features for referential integrity and configurable cascading deletion (automatically controlling the deletion of pointed-to objects or preventing the deletion of pointing-to objects).

Reflective properties

A particularly useful form of OVPs is a *reflective property*, also known as *association properties*. More than one object can point to a particular other object. When that happens, the reflective property mechanism is used to simplify the bookkeeping and let Content Engine perform most of the work. The usual examples have a parent and many children. Suppose you have an `Invoice` object with many `LineItem` child objects. With the reflective property mechanism, define an `Invoice` property on the `LineItem` class and a `LineItems` property on the `Invoice` class. The naming is just a convention that works well in practice. Any property names can be used. To affiliate a new `LineItem` with the `Invoice`, you need to only populate the `Invoice` property on the `LineItem` object. Because it was created as a reflective property, the `LineItems` property on the `Invoice` class automatically reflects the new line item being added. When you access the multi-valued property (the `LineItems` property in our example), the Content Engine automatically performs a query for applicable objects with the appropriate value in the single-valued property (the `Invoice` property in our example).

Many-to-many relationships

Especially because of reflective properties, it is easy to use OVPs to model one-to-many and many-to-one relationships. You might find the need to model a many-to-many relationship. The usual solution for that is to use an intermediate object to express a single pair of relationships. The system class, `ReferentialContainmentRelationship` (RCR), is an example of this solution for the special case of containing objects in folders. A single object can be contained in many folders, and a folder can contain many objects. The document class has a reflective property, `Containers`, which identifies all the RCRs (and, therefore all the containment relationships) that reference a specific document instance. The folder class likewise has a `Contains` property.

You can see that this intermediate relationship object, combined with reflective properties, is a powerful tool for simplifying your modeling of many-to-many relationships. Not only does it express the relationship, but it can also have properties specific to that particular relationship. For example, an RCR has a property, `ContainmentName`, that gives a unique name to a contained object for the purposes of path-based navigation. When you use an intermediate object for a relationship, you can add whatever properties are appropriate to your business needs. Both `ReferentialContainmentRelationship` and `DynamicReferentialContainmentRelationship` classes are subclassable, and you can use them for your own relationships if they happen to fit the folder containment model. Other good choices for the intermediate object are subclasses of customer object and link system classes.

Custom objects

You will often find yourself with a need to hold a collection of related properties for one reason or another. In a database programming environment, you might create a new table with rows representing the collection of information. The Content Manager solution for this is to create a subclass of the custom object class. The custom object system class has only a few properties of its own, and it exists specifically to be subclassed for this use. The invoice and line item example used for reflective properties can also be modeled this way.

As part of the persistence architecture the Content Engine stores all custom objects, regardless of class, in a single database table. It sometimes happens that different kinds of custom objects are used in significantly different ways by applications. For example, an object store might have numerous custom objects that represent business object entities, and it might also have custom objects that represent configuration items. The latter custom objects are relatively few in number and can get lost in the volume of business objects. That can result in performance problems at the database level. Because of this occasional database issue, the 5.2 release introduces *custom root classes*. A custom root

class has its own table in the database but otherwise is similar to a custom object subclass.

Recommendations: When contemplating the use of custom objects in your data model design, consider using a subclass of `CmAbstractPersistable` as a custom root class. This is useful if your objects will not be typical business objects.



Business continuity

In this chapter, we describe how to provide for business continuity with IBM FileNet Content Manager (P8 Content Manager).

We discuss the following topics:

- ▶ Defining business continuity
- ▶ Defining high availability (HA)
- ▶ Implementing a high availability solution
- ▶ Defining disaster recovery (DR)
- ▶ Implementing a disaster recovery solution
- ▶ Best practices
- ▶ Reference documentation

7.1 Defining business continuity

Business continuity is defined as maintaining business services to an organization's customers despite disruptive events that have the potential to interrupt service. Disruptions range from human errors or component failures to full-scale human-caused or natural disasters. Providing for continued business operations in the event of a local component failure is called *high availability*. Business continuity in the event of a full-scale disaster is called *disaster recovery*.

Business continuity is concerned with resuming all critical business functions after disruptive events. High availability and disaster recovery are concerned primarily with the subset of business continuity devoted to keeping information technology (IT) services available during and after disruptions. Beside IT services, business continuity covers all aspects of continuing business operations, including crisis management and communications, alternate work sites for employees, employee disaster assistance, temporary staffing, emergency transportation, physical security, and chain of command.

Business continuity planning (BCP) involves all aspects of anticipating possible disruptions to mission-critical business functions and putting in place plans to avoid or recover from those disruptions. BCP focuses on planning for the successful resumption of all mission-critical business operations after a disruption, not just restoring IT functions. It involves much more than IT professionals. It touches every department in an enterprise from upper management to human resources, to external communications professionals, telecommunications staff, facility management, healthcare services, finance, sales, marketing, and engineering.

Business continuity planning in the limited scope of IT functions will involve the IT department, facility management, telecommunications, and line of business management who can assist in evaluating which IT functions are mission-critical after a disruption or disaster. High availability and disaster recovery plans need to be formally developed and reviewed by all these stakeholders, implemented, and then regularly tested by all staff to be certain that they will function as expected during and after a real disruption.

This chapter covers the part of business continuity that concerns restoring IT functions, in particular P8 Content Manager, after a disruptive event.

7.2 Defining high availability (HA)

What is high availability (HA) and how is it measured? We start by defining availability. A business system is said to be available whenever it is fully

accessible by its users. *Availability* is measured as a percentage of the planned uptime for a system during which the system is available to its users, that is, during which it is fully accessible for all its normal uses.

Planned uptime is the time that the system administrators have agreed to keep the system up and running for its users, frequently in the form of a service level agreement (SLA) with the user organizations. The SLA might allow the system administrators to take the system down nightly or weekly for backups and maintenance, or, in an increasing number of applications, rarely if at all. Certain mission critical systems for around-the-clock operations now need to be available 24 hours a day, 365 days a year.

The concept of *high availability* roughly equates to system and data available almost all of the planned uptime for a system. Achieving high availability means having the system up and running for a period of time that meets or exceeds the SLA for system availability, as measured as a percentage of the planned uptime for a system.

Table 7-1 helps quantify and classify a range of availability targets for IT systems. At the low end of the availability range, 95% availability is a fairly modest target and therefore is termed *basic availability*. It can typically be achieved with standard tape backup and restore facilities. The next level up, *enhanced availability*, requires more robust features, such as a Redundant Array of Independent Disks (RAID) storage system, which prevents data loss in the first place, rather than the more basic mechanisms for recovering from data loss after it occurs. Highly available systems will range from 99.9% to 99.999% availability and require protection from both application loss and data loss. At the high end of this continuum of availability is a fault tolerant system that is designed to avoid any downtime ever, because the system is used in life and death situations.

Table 7-1 Range of availability

| Availability percent | Annual downtime | Availability type |
|----------------------|-------------------------|---|
| 100% | 0 minutes | Fault tolerance for life and death applications |
| 99.999% | 5.3 minutes | Five nines (near continuous availability) |
| 99.99% | 53 minutes | High availability |
| 99.9% | 526 minutes (8.8 hours) | High availability |
| 99% | 88 hours (3.7 days) | Enhanced availability |
| 95% | 18 days (2.6 weeks) | Basic availability |

To make this more concrete, consider the maximum downtime that can be absorbed in a year while still achieving 99.999% availability, also called *five nines* availability. As Table 7-1 on page 219 indicates, five nines availability permits no more than 5.3 minutes of unscheduled downtime per year, or even less if the system is not scheduled for round-the-clock operation. This is near continuous availability, but not strictly fault tolerant. For a three nines target of 99.9%, we can allow 100 times more downtime, or 8.8 hours per year. An availability target of 99%, which still sounds like a high target, can be achieved even if the system is down 88 hours per year, or over three and half days. So the range of availability is actually quite large.

You might be asking yourself, “Why not provide for the highest levels of availability on all IT systems?”. The answer, as always, is cost. The cost of providing high availability goes up exponentially as availability approaches 99.9% and higher.

Choosing an appropriate availability target involves analyzing the sources and costs of downtime in order to justify the cost of the availability solution. Industry experts estimate that less than half of system downtime can be attributed to hardware, operating system, or environmental failures. The majority of downtime is the result of people and process problems, which comes down to a mix of operator errors and application errors.

This chapter focuses primarily on how to mitigate downtime due to hardware outages, system, and IBM FileNet software problems outside the control of an IBM FileNet client, or environmental failures, such as loss of power, network connectivity, or air conditioning. This covers less than half of the sources of downtime. The majority of the sources requires people or process changes.

Our advice is to determine what has caused the most downtime in the past for a particular system and focus first on that. Frequently, we have found that stricter change control and better load testing for new applications provide the greatest benefit. Focus on the root causes of outages first and then address the secondary and tertiary causes only after protecting against the root causes.

Here are several examples of best practices for avoiding downtime from people and process problems:

- ▶ System administrators need to be well-trained and dedicated full-time to their systems so that they are least likely to commit pilot errors.
- ▶ The applications running on the system must be designed with great care to avoid possible application crashes or other failures.
- ▶ Exception handling, both by administrators and application programs, must be carefully thought-out so that problems are anticipated and handled efficiently and effectively when they occur.

- ▶ Comprehensive testing and staging of the system is paramount to avoiding production downtime. Testing of the system under a simulated production workload is critical to avoiding downtime when the system is stressed with a peak load during production. Downtime on a test system does not affect availability of the production system, so make sure to eradicate all the problems before taking a new system, software release, service pack, or even software patch into production.
- ▶ Deploying a new application into production must likewise be planned and tested carefully to minimize the possibilities of adversely affecting production due to an overlooked deployment complication.
- ▶ Thorough user training will help keep the system performing well within the bounds for which it was designed. Users who abuse a system due to ignorance can affect overall system performance or even cause a system failure.

Make sure that all sources of downtime are addressed, if high availability is to be achieved. After the fundamental people-related and process-related problems have been addressed, you need to consider hardware and software availability next.

7.3 Implementing a high availability solution

There are a variety of building blocks for high availability, ranging from the most basic backup and restore facilities, to hardened servers and backup servers, to the best practices: server farms and server clusters.

It is important to note that server farms and server clusters, as those terms are used in this chapter, are different solutions. We will explore server farms first, and then explain how clusters differ.

7.3.1 Load-balanced server farms

Server farms are the best practice for web servers. In fact, they are the best practice, in terms of high availability, for all the server tiers in a P8 Content Manager solution where they are supported. The architecture and function of some servers do not lend themselves to a server farm configuration. But, the core P8 5.2 Content Platform Engine, as well as all the P8 web and presentation tier products, supports server farming. In addition, IBM DB2 pureScale® and Oracle Real Application Clusters (RAC) support server farming.

As we have already discussed in 3.2, “Scalability” on page 52, the key concept for a server farm is to distribute the incoming user workload across two or more

active, cloned servers. This distribution is commonly called “*load balancing*,” which can be implemented either in hardware or software.

This is a scalable architecture because servers can be added to the farm to scale it out for greater workloads. It also provides improved availability because the failure of one server in a farm still leaves one or more other servers to handle incoming client requests. This availability keeps the service available at all times.

In a load-balanced server farm, clients of that server see one virtual server, even though there are actually two or more servers behind the load-balancing hardware or software. The applications or services that are accessed by the server’s clients are replicated, or cloned, across all the servers in the farm. And all those servers are actively providing the application or service all the time.

The load-balancing software or hardware receives each request and uses any one of a variety of approaches for distributing the request workload over the servers in the farm. This can be a simple round-robin approach, which sends requests to the servers in a predefined order. A more sophisticated load balancer might use dynamic feedback from the servers in the farm to choose the server with the lightest current load or the fastest average response time, for example.

In any case, the load balancer tracks the state of each server in the farm, so that if a server becomes unavailable, the load balancer can direct all future requests to the remaining servers in the farm and avoid the down server, therefore, masking the failure.

The key enabler for a server farm is the load balancer. In most cases, IBM FileNet leverages third-party load-balancing hardware and software products, rather than building load balancing into IBM FileNet products themselves.

IBM Content Search Services (CSS) and IBM FileNet Rendition Engine (RE) are two exceptions that provide load balancing on their own, so they do not require any external hardware or software load-balancing solutions.

All the Java application server vendors provide software to balance the Java application workload running in their Java Platform Enterprise Edition (Java EE) environments. For example, the IBM WebSphere Application Server Network Deployment product includes built-in software load balancing for Java applications that are deployed in WebSphere Application Server Network Deployment clusters.

Note: The base WebSphere Application Server product bundled with IBM FileNet Content Manager does *not* include this feature, so WebSphere Application Server Network Deployment must be licensed separately for high availability deployments.

Java EE application server vendors, including IBM, use the term *cluster* for their load-balancing software feature. The other Java EE application servers, Oracle WebLogic and JBoss, also provide a similar load-balancing software feature.

Java method calls by clients of a clustered Java application, such as the P8 Content Platform Engine, are distributed across all the WebSphere Application Server Network Deployment servers running Content Platform Engine by means of the WebSphere Application Server Network Deployment Workload Management (WLM) component. WLM consists of both a client-side component and a server-side component.

The server side, in conjunction with the WebSphere Application Server Network Deployment High Availability (HA) Manager component, keeps track of the health of each instance of the Java application, and sends that information back to the client-side WLM component on the return from every Java method call from the client.

The client-side WLM component, which is part of the WebSphere Java Runtime Environment (JRE) running on the client server, is responsible for distributing the method calls from local Java applications, such as the IBM Content Navigator, over the servers running the target Java application, such as the P8 Content Platform Engine. When IBM Content Navigator makes a content-related or process-related method call to the P8 Content Platform Engine, the local WLM running on the IBM Content Navigator server will decide which currently active server in the P8 Content Platform Engine cluster to use for that call, effectively load balancing all the calls across the servers in the P8 Content Platform Engine cluster.

Network hardware vendors, such as Cisco and f5 Networks, have implemented load balancing for server farms in several of their network devices. f5 BIG-IP is a popular hardware load-balancing device.

There are also many other vendors that have load balancer products. These products are best for load balancing the HTTP network traffic from web browsers to the web application tier in a P8 system, as well as the SOAP/HTTP network traffic from P8 client applications that use the Web Services interfaces to the P8 Content Platform Engine. However, do not use hardware load balancers in combination with WebSphere Application Server Network Deployment WLM software load balancing for the native Java APIs to the P8 Content Platform Engine, because the WebSphere Application Server Network Deployment WLM load balancing is self-contained and complete on its own.

In the best case, the hardware load balancer affects only the initial Java infrastructure call to locate the instances of the P8 Content Platform Engine in the WebSphere Application Server Network Deployment cluster. After that, the WLM component takes over the routing of all the Java method calls to the P8

Content Platform Engine. In the worst case, a hardware load balancer can compete with and disrupt the software load balancing provided by WebSphere Application Server Network Deployment and cause serious performance problems.

In addition to WLM for Java method call load balancing, WebSphere Application Server Network Deployment provides another load balancing feature for HTTP load balancing. This is the WebSphere Application Server HTTP plug-in for all the popular HTTP servers. The plug-in intercepts HTTP traffic flowing through the HTTP server to P8 servers, and distributes that traffic over the P8 servers configured for each HTTP function. For example, HTTP traffic between users' web browsers and the IBM Content Navigator web application can be load balanced by HTTP servers in front of the IBM Content Navigator server instances, if the WebSphere Application Server HTTP plug-in is installed on the HTTP server and configured for that traffic. Another example is traffic from clients of the P8 Content Platform Engine that use the Web Services interface to the content and process functions of Content Platform Engine, rather than the Java API. The Web Services calls are made outside of the Java infrastructure over the SOAP protocol running on HTTP. These calls can be load balanced by any HTTP load balancer, including the WebSphere Application Server HTTP plug-in, or hardware load balancers from the network hardware vendors, such as f5 Networks.

Figure 7-1 shows a logical diagram of a load-balanced server farm. This figure shows a pair of hardware load balancers and multiple servers in the server farm. Redundancy is essential to prevent the failure of one load balancer from taking down the server farm.

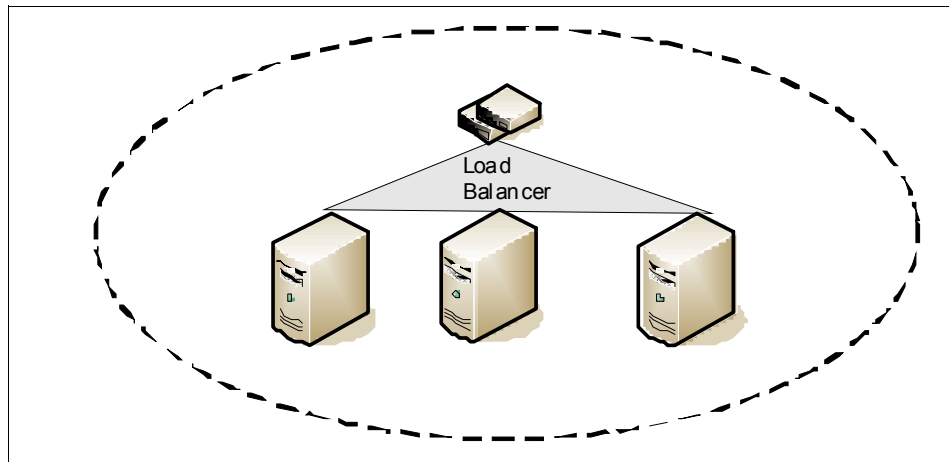


Figure 7-1 A load-balanced server farm

This concept of no single point of failure is key to high availability. Every link in the chain, that is, every element in the hardware and software, must have an alternate element available to take over in case the first element fails. Software load balancers, for example, are designed to avoid any single point of failure; therefore, each server in the farm has a copy of the load-balancing software running on it in configurations using software instead of hardware for load balancing.

The software running on each server in a farm is functionally identical. As changes are made to any server in the farm, you must replicate those changes to all the servers in the farm. In this regard, a key benefit of WebSphere Application Server Network Deployment is its facility for rolling out software changes across all the nodes in a WebSphere Application Server Network Deployment cluster, after one of the nodes in the cluster has been updated. So, it facilitates keeping the software the same across all the nodes of a WebSphere Application Server Network Deployment cluster. (Recall that a WebSphere Application Server Network Deployment Java EE cluster is actually what we call an active-active load-balanced server farm in this chapter. We describe next how that differs from the concept of an active-passive server cluster.)

Load balancing offers a good solution: Any client calling into a load-balanced server farm can be directed to any server in the farm. The load can be evenly distributed across all the servers for the best possible response time and server usage. However, load balancing can be a problem if the servers in the farm retain any state between calls. For instance, if a user initiates a session by providing logon credentials, it is beneficial for those credentials to be cached for reuse on all subsequent calls to the server for that user session.

We cannot ask the user to log in over and over every time the application needs to communicate with the server. Therefore, in one solution, the server keeps a temporary copy of the user's validated credentials in its memory. This works fine if there is only one server, but in a load-balanced server farm, the load balancer can easily direct subsequent calls from the same user session to different servers in the farm. Those other servers will not have the session state in their memory.

Load balancers can be configured for session-based load balancing to solve this session state problem. This is also known as *sticky sessions*, *session affinity*, or *stateful load balancing*. The load balancer keeps track of which server it selected at the beginning of a user session and directs all the traffic for that session to the same physical server. Session-based load balancing is required for the Application Engine, but not for the Content Platform Engine, because the Application Engine caches session state, while the Content Platform Engine does not.

Load-balanced server farms (or Java EE clusters) that manage persistent data stored on disk need to have a way for all the servers in the farm to share the same set of disks. In data stored in databases, such as DB2, all the database vendors provide interfaces with locking and transaction features that enable multiple database clients in a load-balanced server farm to share read/write access to the same database.

In addition to data housed in databases, the IBM FileNet Content Platform Engine manages data stored in file systems, such as the file storage areas for content objects, such as documents and annotations. So all the servers in a Content Platform Engine farm have to be able to read and write to one or more common file systems. The solution is a shared network file system, which network-attached storage (NAS) devices provide natively over the Network File System (NFS) or Common Internet File System (CIFS) protocols. NFS is supported by the UNIX and Linux operating systems, and CIFS is supported by Microsoft Windows. Another option for AIX and Linux based P8 Content Platform Engine servers is the IBM General Parallel File System (GPFS™), which can be deployed with storage area network (SAN) storage devices to provide a shared network file system for P8 servers.

Now, we turn to active-passive server clusters and explore how they differ from active-active load-balanced server farms (or from load-balanced Java EE server clusters, such as WebSphere Application Server Network Deployment clusters).

7.3.2 Active-passive server clusters

Historically, active-passive server clusters were commonly required for the business logic and data tiers beneath the web and presentation layer tier of servers. Examples include business process servers, library or repository servers, and database or file system servers. For instance, IBM FileNet Image Services (IS) is a content repository that requires an active-passive server cluster configuration when deployed for high availability.

Business logic and data tier servers all differ from web and presentation servers in that they directly manage substantial dynamic data, such as content or process data. A stream of dynamic data, by definition, is a stream of new or rapidly changing data. For business logic or data tier server products that have not been specifically designed to allow multiple servers to manage this kind of dynamic data in a safe, cooperative process, a single server must manage the dynamic data set, in order to avoid data inconsistency or corruption from multiple servers trying to make changes to the data simultaneously.

Fortunately, more and more server products, including the IBM FileNet 5.2 Content Platform Engine and its predecessor Content Engine and Process Engine products, make use of transactional software and locking to allow multiple

server instances to manage dynamic data sets safely. Those products can take advantage of active-active load balancing, described previously. But other products, notably IBM FileNet Image Services, do not have this capability, so each data set must be managed by only one server.

Because of that single server architecture, a server farm with two or more active servers does not fit well with servers that have not been designed for cooperative data management. Yet a second server is still needed for continued availability, in case the first server fails. The solution in this case is an active-passive server cluster, where the second server stands by until the first server fails, before stepping in to take over the data management.

The second server needs to have access to the data that was being managed by the first server, either the same exact copy, or a copy of its own. The common solution allows both servers to be connected to the same copy of data either via a network file share or, more commonly, a SAN storage device that both servers can access, but only one at a time. The active server owns the SAN storage, and the passive server has no access.

Shared access to SAN storage in this way is an alternative to replicating the data to a second storage device accessed by the second server. However, maintaining a replica of the data, sometimes called a *mirror*, on a second local storage device is a good practice, as protection against the failure of the primary SAN storage device. Even highly available SAN storage devices, which have internal protection against the loss of a disk drive through redundant copies of the data, have been known to fail completely. Active-passive server clusters can still be configured such that all servers can take over the primary storage in the event of the active server failing, with the local mirror as a standby copy that is used only if the primary storage device failed. The IBM DB2 High Availability Disaster Recovery (HADR) product is an example of a product that provides both active-passive server clustering as well as data replication so that the passive server has its own separate copy of the data.

If there is no local mirror, recovering from the loss of a primary storage device involves either time-consuming restoration from a previous backup, or declaring a disaster and failing everything over to the recovery site, which is also time-consuming. Data updates that have occurred in the time since a backup was taken will necessarily be lost when a backup is restored. If the sources of those updates are still available, the updates can be made a second time to avoid data loss. In comparison to restoring from backup or switching over to a disaster recovery site, switching over to a local replica by reconfiguring the server managing the storage is faster, simpler, and avoids any data loss.

Figure 7-2 on page 228 shows two servers in a server cluster with access to the same shared storage. Recall that some server farms typically do not have this requirement for shared storage. DB2 pureScale, Oracle RAC, and the Content

Platform Engine are exceptions, in that they exhibit both server farm and server cluster characteristics. They take advantage of load balancing, combined with cooperative data management using storage that is simultaneously shared by all the servers. In a load-balanced server farm with shared storage, all the servers are active and thus need to access the storage in parallel, so a network file share is required. An active-passive server cluster, however, is designed to allow only the active server to access the storage, so the single-owner model of SAN storage works well. The typical server cluster does not support load balancing, but it does support shared storage via SAN. The storage is shared in a server cluster in the sense that both servers are connected to the same storage, so they share access to the same storage, but never concurrently in the case of SAN storage.

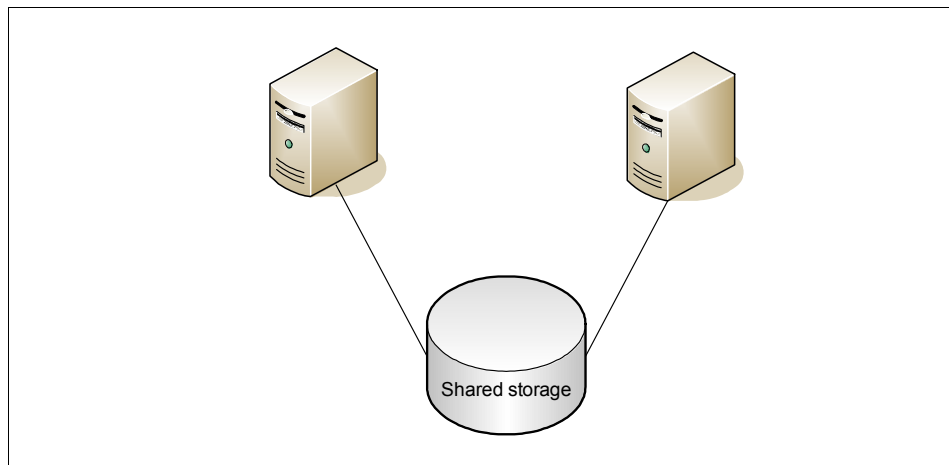


Figure 7-2 Active-passive server cluster

As with server farms, clients of a server cluster see one virtual server, even though the physical server they interact with will change if the primary server fails. If the primary server fails, a *failover* occurs, and the second server takes over the data copy and starts the software to manage the stored data. It also takes over the virtual network address, which is shared by the two servers, making the failover transparent to the client of the server cluster.

Both triggering a failover and actually accomplishing the failover cleanly are the responsibility of clustering software running on both servers. This software is configured on the secondary server to monitor the health of the primary server and initiate a failover if the primary server fails. The active server in an active-passive cluster owns the storage resources, commonly called a *resource group* or *shared volume group*. The resource group is visible from both cluster nodes but only dedicated to the active node. If the active node fails, the clustering software will move the resource group to the remaining passive node. The

passive node sees the resource group but does not write to it until the clustering software ensures consistency. This is called a *shared volume group* for IBM PowerHA® clusters.

After the failed server is repaired and running again, a *failback* is initiated via the clustering software to shift the responsibility back to the primary server and put the secondary server in waiting mode again. This failback is necessary to get back to a redundant state that can accommodate another server failure.

In certain cases, intentional failovers can be used to mask planned downtime for software or hardware upgrades or other maintenance. You can upgrade and test the secondary server offline. And then, you can trigger a failover and apply the upgrade to the primary server while the secondary server is standing in for the primary server.

This type of configuration, in which the second server is inactive or passive until it is called to step in for the active server, is called an *active-passive server cluster*. Several clustering software products also support an active-active cluster configuration, which is similar to a server farm where all servers are active. An active-active cluster configuration is useful for data managing servers that are designed to share the management across more than one server.

However, IBM FileNet products that use active-passive clustering software for high availability all require an active-passive configuration. IBM FileNet products that work with an active-active configuration always use a server farm and load balancing rather than clustering software. (Server farms are always active-active.)

Server cluster software requires agents or scripts that are configured to manage key server processes on a particular server. These agents or scripts allow the cluster software to monitor the health of the application software, as well as start and stop the application software on that server. Cluster software typically comes with predefined agents or scripts for common server types, such as database servers.

A failover in an active-passive server cluster is not instantaneous. It will typically take ten or fifteen minutes or longer, depending on how long it takes the clustering software to stop the failing server, shift the virtual IP address and the storage to the passive server, and start the application software on the passive server. Before the system is accessible again, additional internal steps can take place, such as database transaction recovery. Depending on the state of a database and the number of in-flight transactions at the time of a database server failure, it can take substantially more than fifteen minutes to roll back incomplete transactions before the database is once again online and available.

7.3.3 Geographically dispersed server clusters and server farms

Most server clusters consist of two side-by-side servers. However, certain software vendors also support geographically dispersed clusters. The Symantec Veritas Cluster Server, for instance, supports both *stretch clusters* and *replicated data clusters*. A stretch cluster is defined as two servers in a cluster separated by as much as 100 km (62 miles). The distance limitation is due to the requirement to connect both servers via fiber to the same SAN device for shared storage and also due to the maximum amount of time allowed for the heartbeat protocol exchange between the two servers. The two servers in a stretch cluster always share the same SAN storage device, just as though they were side by side, and operate identically with the way a local server cluster operates.

A replicated data cluster is similar to a stretch cluster, but the remote server always has its own replicated copy of the data. In the event of a failover, the second server comes up on its local copy of the data. In certain cases (but not all cases), this capability removes the need for an expensive fiber connection between the two sites, because neither server needs the speed of fiber to access storage at the other site. Data replication can be done over an IP network. There is still a 100 km (62 mile) distance limitation to ensure that the heartbeat between servers will not time out due to transmission delays and to allow for synchronous replication. See 7.5.1, “Replication” on page 236 for an explanation of synchronous and asynchronous replication.

A replicated data cluster cannot provide the same level of availability as a local cluster, because of the additional downtime required for a data resync to the primary site on a site failback. In addition, particularly in stretch clusters, the network connectivity between the two sites is typically much more expensive and substantially more prone to failure than the local network connectivity between two servers in a local cluster.

Similarly, some server farms can be dispersed geographically across multiple sites. In that case, load balancing must be done across sites. Servers that manage persistent data, such as database servers, need to share a single copy of the data, which necessarily must live at just one of the sites. The network connectivity issues with geographically dispersed server clusters apply to server farms as well. Some vendors supporting server farms caution against geographically distributing their farms. Notably, IBM strongly discourages stretching WebSphere Application Server Network Deployment clusters across sites, due to the added risk of communication failures between the sites and the complexity of this kind of deployment.

Because of the availability trade-offs and communication costs, geographically dispersed server clusters and server farms are generally not the best practice for high availability. However, some organizations have chosen to deploy twin data

centers within a single metropolitan area, typically less than 40 kilometers (25 miles) apart. The motivation behind twin data centers is to reduce the risk of downtime from one data center being unavailable, because of planned or unplanned downtime for the whole data center. By limiting the distance between sites, the networking costs and risk of network failure are reduced, making this approach more feasible.

Still, the simplicity of keeping server clusters and farms local to a single data center is the best practice for high availability, because this minimizes the risk of failure due to the complexity of running these clusters or farms across multiple data centers, and the increased risk of network problems.

As we will see later in the disaster recovery discussion, the best practice solution for the loss of the production data center is to fail over to a standby recovery data center that is typically located hundreds of miles or more away from the production data center.

Twin data centers in a metropolitan area are less attractive for true disaster recovery, because both can be lost in a single local disaster. Even if one of the nearby data centers survives a disaster, the IT staff living in the metropolitan area surrounding the two data centers can effectively become a single point of failure for the two data centers. If their access to the remaining data center is cut off, or they are otherwise unable to work due to effects of the disaster, the remaining data center can be effectively lost without suffering direct damage itself from the disaster.

Some organizations have combined twin nearby data centers with a third recovery center farther away, in order to have both a local disaster recovery option as well as a remote disaster recovery option. That is the best practice when twin nearby production data centers are a company standard. But a more cost-effective and lower-risk solution is to have a single production data center configured for full local high availability, and a remote standby disaster recovery data center located at least a hundred miles away, preferably more.

7.3.4 Server cluster products

All the server vendors offer their own server cluster software products (see Table 7-2), as well as several software vendors.

Table 7-2 Server cluster products

| Server and software platform | Server cluster software products |
|-------------------------------------|---|
| IBM System p® AIX | PowerHA |
| Microsoft Windows Server | Microsoft Cluster Server |

| Server and software platform | Server cluster software products |
|----------------------------------|--|
| Hewlett-Packard (HP) 9000 HP-UX | HP ServiceGuard |
| Sun Solaris | Sun Cluster |
| AIX, Solaris, Windows, and Linux | Symantec Veritas Cluster Server (also supports HP-UX) or IBM Tivoli System Automation for Multiplatforms |

7.3.5 Comparing and contrasting farms to clusters

Table 7-3 summarizes the differences and similarities between load-balanced server farms and active-passive server clusters.

Table 7-3 Comparison of farms to clusters

| Feature | Farms | Clusters |
|--|--|---|
| Clients see one virtual IP address and one virtual server | Yes | Yes |
| All servers active | Yes | No |
| One server active and one server passive | No | Yes, typically |
| Capacity and performance scalable by adding servers | Yes | No |
| Instantaneous failover | Yes, all servers active all the time | No, must wait for software to be started after failover |
| Shared storage between the servers | Not necessarily, but can include a network file share for parallel accesses from all the servers in the farm | Yes, typically SAN storage, which allows just the active server to access the storage |
| Used for web servers, presentation tier, and certain services tier servers | Yes | Not usually |
| Used for data tier servers | No | Yes (except active-active database products such as DB2 pureScale) |

| Feature | Farms | Clusters |
|---|--|---|
| Requires hardware or software load balancer | Yes, such as BIG-IP or WebSphere Network Deployment clustering | No |
| Requires failover cluster software | No | Yes, such as PowerHA, IBM Tivoli System Automation for Multiplatforms, Microsoft Cluster Server, or Symantec Veritas Cluster Server |

Now that we have covered the differences between server farms and server clusters, we explore the advantages of farms over clusters and the advantages of clusters over farms. Server farms have no idle servers, by definition, because all servers in a farm are active. Server clusters always have one or more idle servers in a steady state. Even more importantly, you can expand server farms by simply adding a server clone, thereby scaling out the farm to handle larger workloads. This horizontal scalability is not possible with active-passive server clusters. The last advantage of a farm over a cluster is faster recovery time. Server cluster failovers are delayed by the time that it takes to start the software on the passive server on a failover. All the servers in a server farm are active and immediately available to accept work that has been redirected away from failed servers.

There are also some advantages that clusters have over farms, but on balance farms have the advantage. The chief advantage of a cluster over a farm is that the passive server can be configured identically with the active server, guaranteeing no performance drop-off in the event of a failover. With server farms, even if the initial server sizing is done to allow one server in a two-server farm to handle 100% of the workload, the workload can increase over time to the point where a single server is unable to handle the full workload after a failure. Careful capacity monitoring and periodic testing can prevent this problem from occurring with farms, however.

7.3.6 Inconsistent industry terminology

The terminology used in this book to distinguish load-balanced server farms from active-passive server clusters is not unique to the book, but also not standard across the industry. As you can see in Table 7-4 on page 234, many vendors use the term “cluster” for both farms and clusters. Microsoft, for example, uses both terms for server farms. Symantec/Veritas uses “failover group” for a cluster and “parallel group” for a farm. Both Oracle and IBM call their Java EE application

server farming configurations clusters. As we have seen, farms and clusters, under our definition of those terms, are quite different, therefore the emphasis here on distinct terms for these HA approaches.

Table 7-4 Inconsistent industry terminology for HA

| Vendor | HA terminology |
|------------------|---|
| Microsoft | “NLB cluster” and “cluster farm” = farm “Server cluster” and “cluster server” = clusters |
| Symantec Veritas | “Failover group” = cluster “Parallel group” = farm |
| Oracle | WebLogic “cluster” = farm |
| IBM | WebSphere “cluster” = farm PowerHA “cluster” = cluster |

7.3.7 Server virtualization and high availability

Chapter 3, “System architecture” on page 37 introduced the concept of server virtualization and its promise of consolidating data center hardware and thus reducing total cost of ownership for the data center. This has considerable appeal, but it can also have a negative impact on availability. If a server farm or server cluster with two physical servers is consolidated into two virtual servers hosted on the same physical server, you must be careful to ensure that the physical server has no single points of failure. Does it have redundant power supplies, network interface cards, processors, memory, and so on? If any single component failure on a server can take down all the virtual servers hosted on it, that server cannot act as host for all the servers in a cluster or farm. Two of the virtual servers must be hosted by different physical servers in this case to avoid downtime caused by a single component failure.

7.4 Defining disaster recovery (DR)

Now, we turn from high availability to disaster recovery. How do they differ? Both high availability and disaster recovery are part of business continuity, that is, making sure that critical business systems and processes can continue to operate despite system failures and disruptions. However, disaster recovery and high availability solutions perform under different circumstances that require different solutions.

Disaster recovery concerns restoring service after the loss of an entire business system or data center due to natural or human-made disasters, such as fire,

flood, hurricane, earthquake, war, criminal action, or sabotage. In contrast to that, high availability concerns keeping a business system available despite a local component failure – such as a server power supply failure, a network switch failure, or a disk crash – that leaves most of the system untouched.

For recovery from the loss of an entire production system in a disaster, a full remote system with its own up-to-date copy of the data is needed. All users and operations must be switched over to the remote system. Compare that to when just a single component fails in a data center: the optimal solution then is an automated, localized, and limited substitution of a single replacement component for the failed component. Server farms and clusters substitute a single replacement component with minimal disruption to the rest of the system and its users. Disaster recovery solutions are much more drastic, disruptive, time-consuming, and heavyweight, because they have to replace an entire system or data center, not just a single failed component. Therefore, disaster recovery solutions are an inappropriate choice for high availability.

Disasters, such as the World Trade Center destruction on 11 September 2001 (9/11) or Hurricane Katrina in New Orleans and the Mississippi Gulf Coast, can have a devastating effect on businesses in their path. Organizations with business continuity, HA, and DR plans were much more likely to rebound and recover from 9/11 and Katrina than those without such planning. Analysts estimate that a significant number of businesses that suffer an extended IT systems outage due to disaster go out of business within a year or two; other businesses never resume operations at all. The obvious inference is that planning and preparing for disaster recovery is a best practice for businesses of all sizes.

7.4.1 Disaster recovery concepts

There are two key metrics that play important roles in determining an appropriate disaster recovery (DR) solution for a particular business and application. They are Recovery Time Objective (RTO) and Recovery Point Objective (RPO).

In certain cases, the most recent data changes at the production site, which stretch back to a point in time prior to the disaster, do not make it to the recovery site because of a time lag that is inherent in how the data is replicated. The magnitude of this time lag is dependent on the particular type of data replication technology that you choose. Assuming a disaster occurs, the *recovery point* is the point in time before the disaster that represents the most recently replicated data. How far back in time is the business willing to go after the disaster happens? That is, the RPO translates to how much recent data the business is willing to lose in a disaster.

The duration of time that passes before the systems can be made operational at the recovery site is called the *recovery time*. The RTO is the business's time requirement for getting the system back online. That is, how much downtime can the business endure?

RPOs and RTOs for different businesses and industries range from seconds to minutes or days, even to weeks, depending on business requirements.

7.5 Implementing a disaster recovery solution

Disaster recovery can be greatly facilitated by two key technologies. One is data replication to a remote recovery site, and the other is software or scripting that can automate most of a site failover to a recovery site after a disaster takes away the primary site. The RTO and RPO for a particular business determine when these two technologies are required for the disaster recovery solution for that business. With an RPO and RTO measured in days to weeks, that is, if the business is willing to lose days to weeks of data and can wait days to weeks for the system to come back online, tape backup and restore are sufficient. But if an RPO of seconds to hours is desired, a form of data replication is required. If an RTO of hours to weeks is acceptable, replication alone might suffice. But if an RTO of seconds to hours is desired, both replication and automated site failover will be required.

Next, we explore replication and automated site failover in more detail.

7.5.1 Replication

Backing up to tape or other removable media is the minimum for copying data for use after a disaster. You must ship the media off-site to a location outside of the projected disaster impact zone. The greater the distance of the location from the production site, the lower the risk that both production and recovery sites will be affected by the same disaster. One general rule is that a backup tape vault and recovery site must be at least 48.28 km (30 miles) away from the production system, which in most cases is sufficient to avoid a flood or fire disabling both sites. However, sites that are close together can still be in the same impact zone for earthquakes, hurricanes, or power grid failures, so more cautious organizations separate their production and recovery sites by hundreds, if not thousands, of miles.

Companies usually perform backups once a day, which meets only a 24 hour RPO. That means that as much as 24 hours of data can be lost. The recovery time required for data restoration from tape can be days due to the need to restore a series of tapes that represents a full backup and subsequent

incremental or differential backups. So, you measure both RPO and RTO in days if the only DR provision is tape backup.

For a better RPO, that is, to reduce the potential data loss in a disaster, you need to periodically replicate the data to a remote disk, because periodical replication can be done more often than tape backup. This effectively reduces the window of data loss. Continuous replication that is done in real time can avoid any data loss at all.

Note: When you use continuous data replication products, point-in-time backups, such as tape backup or periodic replication, are still required in order to recover from data corruption or accidental deletion. Continuous replication copies the corruption or deletion to the replica; therefore, you need to be able to fall back on a point-in-time copy prior to when the corruption occurred.

There are several levels at which you can perform replication: the application level, the host level, and the storage level. Database replication is the best example of application-based replication. Host-based replication is beneath the application level, but it still resides on the server host and typically runs at the file system or operating system level. Storage-level replication is implemented by the storage subsystem itself, frequently, a SAN device or a NAS device.

Application-based replication

Application-level software that understands the structure of data and relationships between data elements can copy the data intelligently, so that the structure and relationships are preserved in the replica. Database and object-based replication are examples. *Database replication* ensures that the replica database is always in a consistent state with respect to database transactions. *Object-based replication* ensures that content objects that include both content and properties are replicated as an atomic unit, so that the content and properties are always consistent with each other in the replica.

Each database vendor has replication products that replicate just the database, but not other data. Examples include IBM DB2 High Availability and Disaster Recovery (HADR) and Oracle Data Guard. Database replication products are typically based on shipping database logs to the recovery site to be applied to a database copy there. The advantage of these products is that they keep the database replica in a fully consistent state at all times, with no incomplete transactions, which reduces the recovery time required when bringing up the database after a disaster. The disadvantage of these products is that they have no means to replicate anything other than databases. File systems that need to be kept consistent with the database, for instance, have to be replicated by a different replication mechanism, which introduces the possibility of inconsistency between the database and file system replicas.

Host-based replication

In contrast to application-based replication, *host-based replication* has no understanding of the data content, structure, or interrelationships. It detects when a file or disk block has been modified and copies that file or block to the replica. Symantec Veritas Volume Replicator and Double-Take Software Double-Take are examples of host-based replication products. Unlike application-based replication, they can be used to replicate all forms of data, whether it is in a database, a file system, or even a raw disk partition. Several of these products use the concept of *consistency groups*, which tie together data in different volumes and allow all the data to be replicated together, therefore maintaining consistency across related data sets, such as databases and file systems. In contrast to application-based replication, however, the replica is not guaranteed to be in a clean transactional state, because the replication mechanism has no visibility into database or file system transactions. Recovery can take longer, because incomplete transactions must be cleaned up prior to making the data available again.

Storage-based replication

All of the storage vendors offer storage-based replication for their SAN and NAS products. The storage products themselves provide storage-based replication and do not use server host resources. Examples include IBM Metro Mirror (PPRC) and Global Mirror (XRC), EMC SRDF and MirrorView, Hitachi Data Systems TrueCopy, and Network Appliance SnapMirror.

NAS products replicate changes at the file level. SAN products replicate block by block. In both NAS and SAN replication, as with host-based replication, there is no knowledge of the structure or semantics of the stored data. So, databases replicated in that way can be in any transient state with regard to database transactions and therefore might require more database recovery time when the replica is brought online. That increases the overall recovery time.

NAS replication covers any data in the file system, whereas SAN replication, which is at the lower level of disk blocks, covers all data stored on the disk.

An emerging specialization of storage-based replication uses a SAN network device to intercept disk writes to SAN storage devices and manage replication independently of both the server host and the storage devices. IBM SAN Volume Controller is an example of this type of product. It has the advantage of being able to span heterogeneous SAN storage devices and replicate data for all those devices in a consistent manner. You can think of the IBM SAN Volume Controller as a new form of storage-based replication, because it resides in the Fibre Channel infrastructure used to access SAN storage. Analysts have a new term for this kind of replication: *network-based replication*.

Synchronous as opposed to asynchronous replication

Host-based and storage-based replication commonly support two modes of operation: synchronous and asynchronous. *Synchronous replication* writes new data to both the production storage and the remote recovery site storage before returning success to the operating system at the production site for the disk write. So, when the operating system signals that a disk write is complete, it has actually been completed on both storage devices. You can think of synchronous replication as logically writing the data at both sites at the same time. That means that after a disaster strikes the production system, we know that the recovery site has all the data right up to the last block that was successfully written at the production site. Synchronous replication ensures that there is no data lost in a disaster, as long as the recovery site survives the disaster. (However, incomplete transactions can still be rolled back when the recovery system is started, leading to unavoidable loss of the data in those transactions, even with synchronous replication.) But to make the latency for disk writes short enough, synchronous replication is typically feasible only for sites that are separated by 96.5 km (60 miles) or less. Above that separation, the wait for the write to the recovery site slows the overall speed of the system significantly. The wait is a function of the distance between sites, because signals can travel no faster than the speed of light between sites. At more than 96.5 km (60 miles), the latency becomes too great in many cases, although certain storage vendors are now extending this distance to 290 km (180 miles).

For sites that are separated by more than 96.5 km (60 miles), *asynchronous replication* is the choice. Asynchronous replication is not done in lock step, the way that synchronous replication is. Instead, the local disk write is allowed to complete before the write is completed to the second site. The update to the second site is said to be done “asynchronously” from the local update, that is, not in the same logical operation. This method frees the production system from the performance drag of waiting for each disk write to occur at the remote site. However, it opens a time window during which the production site data differs from the recovery site copy. That difference represents data that is lost in a disaster when asynchronous replication is used. In exchange for that data loss, the two sites can be any distance apart, although the further apart they are, the greater the typical data loss.

Storage vendors have devised a way to ensure no data loss over any distance, however, by a configuration involving a third copy as shown in Figure 7-3 on page 240. This solution requires a nearby synchronous replica and a remote asynchronous replica. The data from the production site is replicated synchronously to a backup site within 96.5 km (60 miles), which is Site 2 in Figure 7-3 on page 240, and replicated asynchronously to a remote site, Site 3, any distance away. As long as only one of the three sites is lost in a disaster, it is always possible to recover all the data from the remaining two sites. In the diagram in Figure 7-3 on page 240, if Site 1 is lost in a disaster, the synchronous

copy at Site 2 holds all the data up to the moment of the disaster. From there, the data can be replicated asynchronously to Site 3, the actual recovery site, therefore extending zero data loss all the way to Site 3. It works, but the added replica and site can be expensive.

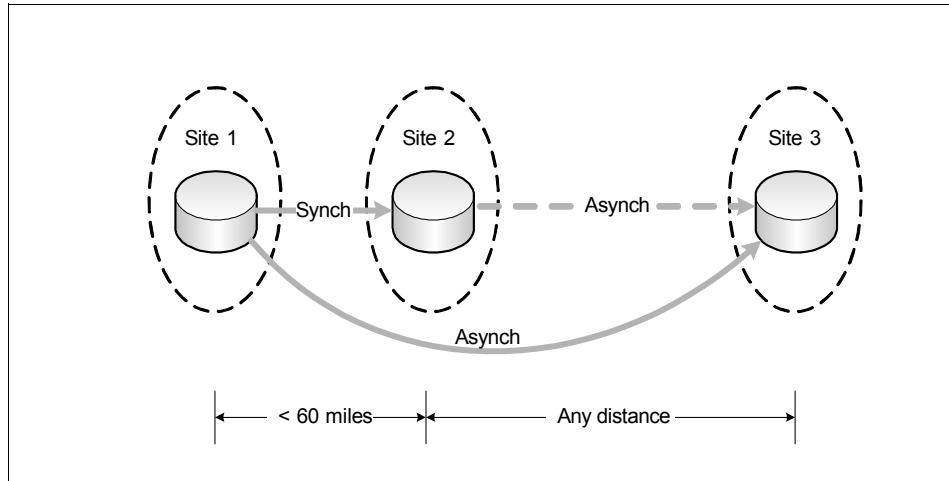


Figure 7-3 Zero data loss replication over any distance

Several vendors support an optimized version of the second site called a “*bunker site*” where only the blocks not yet replicated are stored and no others. The list of the blocks that have not yet been replicated is typically a small list, so a bunker site can be configured with minimal storage space, which reduces the overall cost of this solution. IBM Asynchronous Cascading Peer-to-Peer Remote Copy (PPRC) is an example of this three-site zero data loss solution.

Comparing the replication options

What makes host-based or storage-based replication better than database-based replication? First, storage-based replication has the advantage that it allows a single replication product to be used for all data. With database-based replication, the database is replicated separately from the rest of the data, which can lead to inconsistency between the databases and the other data stored in a file system, such as content data. Second, using a common replication product for all data also simplifies the DR solution, which leads to less required training of system administrators and less total cost of ownership overall. Third, synchronous storage-based replication prevents any data loss other than incomplete transactions. Database-based replication typically is asynchronous and thus is vulnerable to more data loss in a disaster. Host-based replication shares these three advantages over database-based replication.

Lastly, storage-based replication is implemented entirely by the storage device. Database-based or host-based replication runs on the server and takes up server resources. (Vendors of host-based replication products counter that the load on the server is minimal and just a small percent.)

Why choose database-based replication over storage-based replication after you see these disadvantages? The key reason is the lower recovery time that can result from the database replica being in a cleaner state and therefore requiring less recovery processing. A database replicated via its native replication facility is always in a clean database transaction state, so no incomplete database transactions have to be rolled back when the backup database is activated. This allows the system to recover more quickly, which can be viewed as more critical than a small amount of data inconsistency, when minimal recovery time is of paramount importance. Moreover, if all the content and property data is stored in the database, which is an option with the P8 Content Manager for small deployments, database-based replication has no consistency disadvantage or cost of ownership disadvantage.

7.5.2 Automated site failover

The second key technology that is used in many disaster recovery solutions is automated site failover. Some software vendors offer a product to do this called a *global cluster option* or a *geographic cluster manager*. We use the generic term global cluster manager here to distinguish it from geographically dispersed clustering, which we described previously. Recall that geographically dispersed clusters are still clusters in the sense of a heartbeat between the nodes and failover if the active server fails; they just have the servers dispersed over a distance as great as 96.5 km (60 miles). A global cluster manager, however, extends an ordinary server cluster with the capability to oversee multiple sites that are any distance apart. It manages local server clusters at each site, controls replication between sites, and updates Domain Name System (DNS) servers to redirect users to the recovery site system. Its major function is to automate most or all of the process of failing over from a production site to a recovery site after a disaster.

Most organizations prefer to have at least one manual decision step before declaring a disaster, because of the gravity and cost of switching all operations and users to a recovery site. But after that decision has been made, a global cluster manager can automate the rest of the process. This is advantageous, because automating the process reduces the chances of human error, makes the process repeatable and testable, and thus increases the chances of a successful site failover in the highly stressful period following a disaster. Symantec Veritas Global Cluster Option is one example of a global cluster manager. The Geographic Logical Volume Manager (GLVM) configuration of IBM PowerHA

SystemMirror® Enterprise Edition offers some global cluster management features for the AIX platform.

In the absence of a global cluster manager, server command-line scripting is another way to automate key parts of a site failover.

7.5.3 Disaster recovery approaches

IBM ECM Lab Services defines three common approaches for disaster recovery:

- ▶ Build it when you need it.
- ▶ Third-party hot site recovery service.
- ▶ Redundant standby system.

Build it when you need it

The lowest cost approach, but the slowest and the hardest to test, is to build a replacement system after a disaster has occurred. There is nothing in place prior to a disaster, which makes it extremely low cost, but it allows no testing either. This approach has an RTO of days to weeks.

Third-party hot site recovery service

The second approach is to contract with a third party for a hot site recovery service. Third parties, such as SunGard, IBM, and HP, have shared recovery sites around the world that you can reserve by contract for use in the event of a disaster. This approach costs more than the first approach, of course, but it also offers a shorter recovery time, because the site is equipped and hot at the point of disaster. Data has to be restored at the hot site, but no hardware has to be acquired or configured. The third-party providers include regular testing of failover to their site as a part of their service. IBM ECM Lab Services has an offering to assist you in setting up and testing the hot site and activating it in the event of a disaster. This approach has an RTO of hours to days.

Redundant standby system

The third and most frequently chosen approach is a standby redundant system in place at a client-owned and operated remote recovery site or at a third-party site. This approach is the highest cost approach, because the cost of the redundant system is not shared with anyone else. But it offers the shortest recovery time, particularly if the data replica is constantly updated and available for use. It also can be tested on a regular basis, which is in keeping with best practices for ensuring that a disaster recovery plan will actually work as expected when needed. This approach has an RTO of minutes to hours.

Comparing the costs and technologies

No matter which of these DR options you choose, it is essential to have a copy of the data off-site. Table 7-5 on page 244 summarizes the data backup or replication choices and costs, as well as the recovery site choices. Table 7-5 on page 244 shows the relationship between recovery time, recovery point, and the type and cost of data replication required to achieve that recovery time and recovery point. Like high availability choices, the choices for disaster recovery become exponentially more expensive as RTO and RPO approach the minimums of hours to minutes. The cost increase is due to the changes in disaster recovery technologies required to meet increasingly more ambitious recovery times and points.

For an RTO of three days or more, the minimum level of data replication, backing up to tape, is sufficient. As we noted earlier, a form of point-in-time backup, such as tape backup, is always required, regardless of RTO, as a means of recovering from data corruption or accidental deletion. The solution is to retrieve the latest backup tape or other point-in-time backup from the off-site storage location and restore the data to a point in time prior to the corruption or deletion of the data. Full data restoration from tape is a slow and laborious process, which typically involves a full backup tape and a number of incremental backup tapes after that, which takes days for completion. Backups are done periodically, usually once a day, possibly multiple times a day, so the RPO for this minimum solution is hours to days of lost data.

Periodic replication to off-site storage characterizes the next two solutions up the cost curve with an increase in cost for communications links, but providing an RPO and RTO of hours, not days. Periodic point-in-time backup to remote storage, usually disk storage, is the first step up from standard local tape backup. The next step up consists of shipping database or file system update logs to the remote recovery site, where they are applied to a copy of the data to bring it up-to-date with that log. These are both done on a periodic basis, but as the period is shortened, it approaches the limit of continuous replication, which is the next step up the cost curve.

Table 7-5 Range of disaster recovery solutions

| Recovery time | Recovery point | Cost | Technologies |
|--------------------|-------------------------------|--------------------------|--|
| Minutes to an hour | Zero data loss | \$\$\$\$\$\$\$\$\$\$\$\$ | Hot standby site, synchronous replication, or global clustering |
| 1 - 6 hours | Minutes of data lost | \$\$\$\$\$\$\$\$ | Hot or warm standby site, asynchronous replication, or global clustering |
| 6 - 12 hours | Hours of data lost | \$\$\$\$ | Warm standby site, continuous or periodic replication, or log shipping |
| 12 - 24 hours | Hours to days of data lost | \$\$\$ | Warm or cold standby site, or periodic backup to remote storage |
| Days to weeks | One or more days of data lost | \$ | Cold or no standby site, or nightly tape backups shipped off-site |

The cost now starts to accelerate upward. As the name implies, *continuous replication* is the process of replicating data to the recovery site as it changes, that is, on a continuous basis. Near continuous and continuous replication greatly decrease the potential for data loss when compared to periodic replication, which brings the RPO down to seconds worth of data loss, or even zero data loss in synchronous replication.

Disaster recovery time is similarly decreased with synchronous and asynchronous replication, because the data is kept continuously in sync, or close to it, at both sites. In the event of a disaster, no time is required to bring the data up-to-date, as is the case with restoring from backup, periodic replication, or log shipping, but time might be required for configuring and bringing up a duplicate of the application environment on the replicated data. The RTO is in the range of hours in that case, or, if a complete application environment is maintained at all times at the recovery site, and global clustering is used to automate and speed site failover, RTO can be in the range of just minutes.

7.6 Best practices

Having defined the concepts of high availability and disaster recovery and having detailed the key technologies and approaches used for HA and DR solutions, what are the best practices for configuring P8 Content Manager for high availability and disaster recovery from the available options and approaches?

Best practices for high availability

We start with high availability, which is summarized on the left side of Figure 7-4 on page 246. For P8 systems that will be accessed from an untrusted network, including the public Internet, a DMZ is required for shielding the P8 system from attack. Users or client applications on an untrusted network are shown on the far left, with a DMZ interposed between them and the P8 system. The DMZ consists of an external firewall shielding the DMZ from the untrusted network, a pair of load-balanced HTTP servers behind the external firewall, and an internal firewall shielding the internal, trusted network from the DMZ. The HTTP servers in the DMZ intercept all HTTP traffic from the untrusted network and forward traffic (for authenticated users only) through the internal firewall to the P8 system on the trusted network.

External users typically use a URL address for the P8 web applications that references a public HTTP port, such as port 80, by default. For authenticated users, the HTTP servers in the DMZ map that public port to the specific private port configured for the intended P8 web application, enabling the users' requests to be forwarded through the internal firewall to the P8 web application server.

The first part of the P8 architecture, to the right of the DMZ inside the trusted internal network, is the web and presentation tier. For this tier, where the IBM FileNet P8 Application Engine, WorkPlace, FileNet Workplace XT, and IBM Content Navigator predefined web applications live, as well as custom applications, the best practice is load-balanced server farms. All the servers in this tier are active with incoming user/client HTTP requests directed to the load balancer via virtual host names mapped to virtual IP addresses assigned to each application, and then distributed by the load balancer across the servers running those applications. IBM FileNet P8 eForms, IBM Enterprise Records, and IBM Case Manager are all hosted on this tier as well and thus must be deployed in load-balanced server farms for high availability.

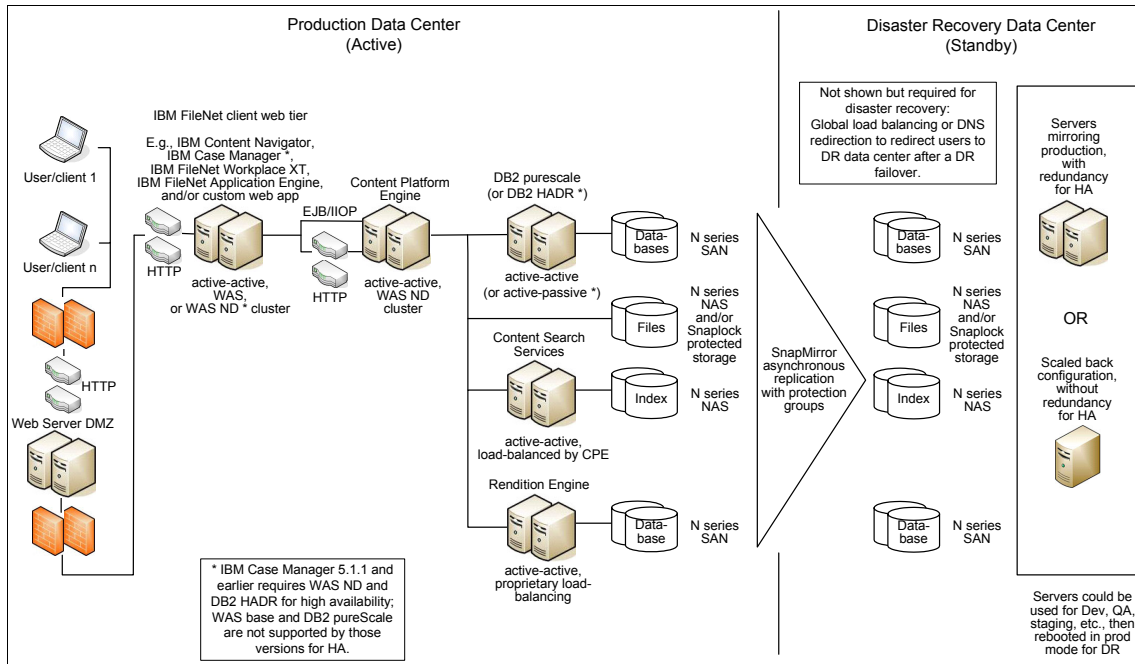


Figure 7-4 Recommendation for IBM FileNet P8 5.2

At the business logic tier, sometimes also called the *services tier*, the HA best practices for the core P8 components shown in Figure 7-4 are all load-balanced server farms. Only a few optional P8 components, not shown in Figure 7-4, require active-passive server clustering because they do not support active-active load balancing. Process Simulator is an example. (Many clients choose not to make Process Simulator highly available, because it does not play a runtime production role.)

Two or more P8 Content Platform Engine servers must be deployed in a load-balanced server farm when high availability is required.¹ The Content Platform Engine has been qualified with both hardware and software load balancers.

¹ Prior to P8 4.0, the Content Engine supported both farming and clustering for its Object Store Services component, but only active-passive clustering for its File Store Services component. Starting with P8 4.0, these components were unified and have since supported farming across the board. Prior to P8 4.0, the Process Engine required active-passive server clustering for high availability, but has also supported farming since P8 4.0. In P8 5.2, the Content Engine and Process Engine were merged together into the Content Platform Engine.

Note: A Content Platform Engine deployment typically requires both Java EE software load balancing via Java application server load balancing (for example, WebSphere Application Server Network Deployment in WebSphere), as well as HTTP load balancing via hardware or software load balancing, when deployed for high availability.

Most client applications, such as IBM Content Navigator and IBM Case Manager, use the Java EE EJB interface and transport when they access the Content Platform Engine. They therefore impose a requirement on the Content Platform Engine servers to be deployed on the clustering version of a Java EE application server (WebSphere Application Server Network Deployment in WebSphere) in order to provide the Java EE software load balancing required for the EJB transport. Other client applications, such as IBM Content Collector, use the content and process Web Services interface and transport when interacting with the Content Platform Engine, and therefore require the deployment of HTTP load balancing for the Content Platform Engine. So, the typical P8 system will require both forms of load balancing, Java EE load balancing such as WebSphere Application Server Network Deployment, and HTTP load balancing, for the Content Platform Engine when it is deployed in a high availability configuration. Figure 7-5 on page 248 shows both Java EE load balancing, in this case implemented by WebSphere Application Server Network Deployment WLM, and HTTP load balancing, in this case implemented by a pair of hardware load balancers, for a pair of Content Platform Engine servers.

IBM FileNet Image Services repositories can be federated with the P8 Content Manager via Content Federation Services. Image Services must be deployed in active-passive server clusters for high availability; it does not support being deployed in load-balanced server farms.

At the data tier, all the database servers can be deployed in active-passive server clusters for HA, such as DB2 HADR. In addition, DB2 pureScale and Oracle RAC are active-active load-balanced alternatives.² The Content Platform Engine makes use of network file shares for file storage areas for content storage and index areas for content-based search indexes, so the network file servers or NAS devices underlying the Content Platform Engine file storage areas and index areas need to be highly available as well. For a network file server, the typical HA configuration is an active-passive server cluster; NAS devices typically have internal support for either active-active or active-passive configurations for HA. NAS devices are purpose-built for high performance and scalability, so they generally scale and perform much better than generic server clusters providing a network share to SAN storage.

² IBM Case Manager 5.1.1 only supports active-passive database clusters, due to a Business Space constraint.

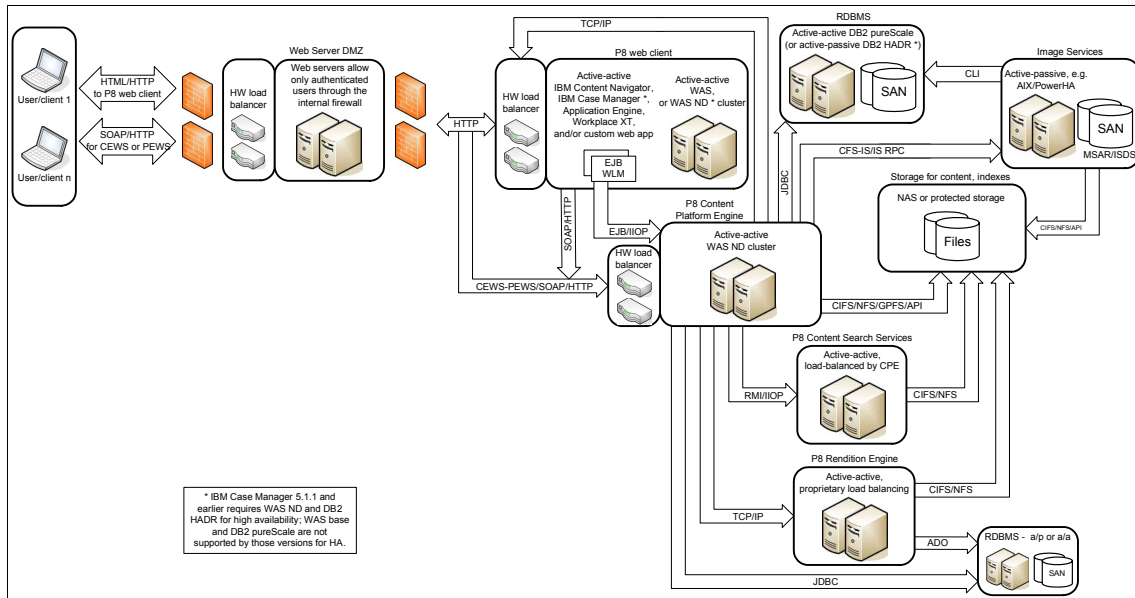


Figure 7-5 High availability best practices for P8 5.2 with protocol detail

Best practices for disaster recovery

For disaster recovery, the best practice is dependent on RTO and RPO. In all cases, point-in-time backup to tape or disk is the best practice for protection against data corruption or accidental or malicious deletion. For any RTO/RPO values less than days to weeks, we suggest data replication to a remote standby site as a best practice, as shown in Figure 7-4 on page 246. At the high end, with RTO and RPO in the range of minutes to hours, a dedicated warm standby recovery site and automated site failover are the best practice, and near continuous to continuous replication is also the best practice. Zero data loss requires synchronous replication to a bunker site or intermediate site if the distance to the remote recovery site is too great. For the absolute minimum RTO, on the order of minutes, database-based replication, in addition to storage-based or host-based replication for the other data, is the best practice. For the best data consistency after a disaster, at the risk of adding minutes to an hour of database recovery time to RTO, the use of a single replication mechanism for all data, combined with consistency groups, is the best practice.

The best practice for redirecting the user community to the replacement systems at the recovery site is via DNS updates or DNS load balancers such as f5 BIG-IP Global Traffic Manager, Cisco Global Site Selector, or similar products from other network device vendors. DNS aliases (CNAMEs) must be used by the user's client computers to locate the P8 Content Manager services, so that the aliases can be redirected after a disaster through DNS updates or DNS load balancing.

This redirection allows reconnection to the recovery site without making any client computer changes. The DNS servers or DNS load balancers themselves must be redundant, of course, to avoid being a single point of failure.

Combining HA and DR into a single solution

There is a common temptation to try to simplify business continuity by combining high availability and disaster recovery into a single solution. The idea is to locate a second site within the same metropolitan area as the production site and make both sites active with each site having a full copy of the data. This is a workable approach when the data being managed is essentially static, as in a corporate website. Changes to the website are carefully reviewed and managed and then pushed out to multiple hosting sites in parallel, and incoming user requests can be load-balanced across the sites. If one of the sites goes down or even is lost in a disaster, user requests can be directed to the other site for continuous access to the largely static content (assuming the second site is far enough away to be out of the disaster's impact zone).

Why does this approach not work with Content Manager? The key is the nature of the data and how it must be managed. P8 Content Manager, as the name suggests, is designed to manage rapidly changing and growing collections of data that are being accessed and modified in parallel by users across an enterprise. Unlike the largely static data of a corporate website, which is published or released to the site in a carefully controlled authoring and information publication process, content in a typical P8 Content Manager object store is being collaboratively authored, enhanced, deleted, created, and processed in a dynamic manner under transaction control to avoid conflicting changes. As a result, only a single active copy of the data can be online and changeable at any point in time so that transaction locking can be enforced and changes are saved in a safe, consistent manner. Therefore, the basic idea of two sites, in which each site has an active copy of all the content, is not the best practice for a transactional system. It is not supported by the P8 Content Manager.

A related temptation is to deploy a disaster recovery solution with a standby (inactive) copy of the data at the recovery site and depend on this single solution for both high availability and disaster recovery. This can be done with P8 Content Manager, but there is a clear trade-off that you need to carefully consider. Relying on a disaster recovery configuration for high availability compromises the availability target for the system, because any failure leads to a full site failover as though the entire production site had been lost in a disaster. A site failover is a time-consuming, complicated process that necessarily takes much longer than a single server failing over to a local passive server in a cluster, and even longer than the nearly instantaneous switch to another, already-active server in a server farm when a server fails in that farm. The net result is that high availability (in the

range of 99.9% and higher) is not reachable when every local failure triggers a full site failover (and later a full site failback to return to a protected state).

How about using geographically dispersed farms and clusters, that is, with the farms and clusters split between the two sites? If one server fails, the server at the other site takes over, either coming up at the time of failure in an active-passive server cluster or simply taking on redirected client requests in server farms. Again, there is an availability trade-off because of the added risk of communication problems between the two sites. We do not recommend geographically dispersed farms and clusters as best practice because of the added risk and higher networking costs.

So the best practice is to deploy local server farms and clusters for high availability in order to provide for continuing service in the event of local component failures and to deploy a second site with data replication and, optionally, global clustering, to provide for rapid recovery from disasters. The best practice is to locate the recovery site outside the disaster impact zone of the production site.

7.7 Reference documentation

For additional information on high availability, see the IBM FileNet Version 5.2 Information Center section devoted to high availability:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.sysoverview.doc/p8pha001.htm>

See the IBM Redbooks publication *IBM High Availability Solutions for IBM FileNet P8 Systems*, SG24-7700, for more details on P8 (and IS) high availability deployments:

<http://www.redbooks.ibm.com/abstracts/sg247700.html?open>

See the IBM Redbooks publication *Disaster Recovery and Backup Solutions for IBM FileNet P8 Version 4.5.1 Systems*, SG24-7744, for more details on P8 (and IS) disaster recovery deployments:

<http://www.redbooks.ibm.com/abstracts/sg247744.html?open>

See “Images Services 4.1.2 High Availability Procedures and Guidelines.” This document describes both high availability via clustering software (Microsoft Cluster Server or Veritas Cluster Server) and disaster recovery via data replication software (Veritas Volume Replicator - see Appendix C) for Image Services:

<ftp://ftp.software.ibm.com/software/data/cm/finenet/docs/isdoc/412x/HACluster.pdf>



Capacity planning with IBM Content Capacity Planner

In this chapter, we briefly discuss capacity planning and the use cases for the IBM Enterprise Content Manager system capacity planning tool. This tool is called IBM Content Capacity Planner, formerly known as Scout.

We cover the following topics:

- ▶ IBM Content Capacity Planner:
 - Example use cases for IBM Content Capacity Planner
 - Capacity planning for new systems
 - IBM Content Capacity Planner output
 - Predictions from a baseline
 - Best practices
- ▶ IBM FileNet Disk sizing Tool spreadsheet
- ▶ Performance-related reference documentation:
 - Standard product documentation
 - Benchmark papers

8.1 IBM Content Capacity Planner

When you introduce a new system or extend an existing one, choosing the correct hardware is an important consideration in your planning. The IBM sales team supports you in planning the capacity of the system during this phase.

The marketing team uses IBM Content Capacity Planner to model transactions and to obtain answers to various questions:

- ▶ Based on the projected use of the IBM FileNet ECM system, what servers are needed?
- ▶ Given a certain hardware configuration, how busy will the servers be?

IBM Content Capacity Planner is generally used by IBM FileNet ECM System Engineers, IBM FileNet ECM Lab Services, and IBM FileNet ECM Partners.

After modeling a workload, IBM Content Capacity Planner produces utilization reports that show the demand placed upon a certain set of hardware by that workload.

Figure 8-1 illustrates the basic modeling process for capacity planning.

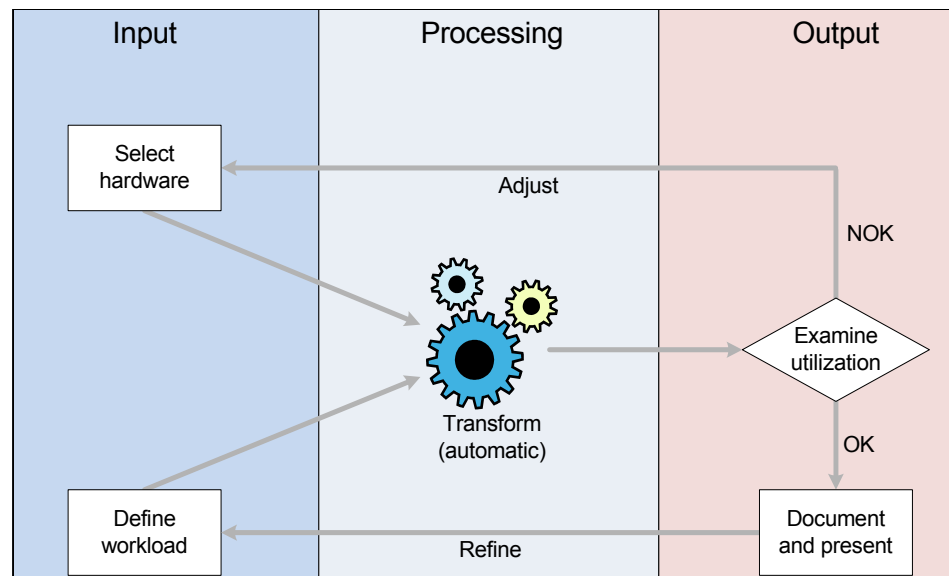


Figure 8-1 Basic modeling process for capacity planning

IBM Content Capacity Planner uses at least two input sources. One is the hardware configuration, and the other is the defined workload that consists of one or multiple transactions. The output from IBM Content Capacity Planner consists of performance charts. If the system utilization of all components is below a threshold, the system is deemed adequate to meet the workload requirements. The results are documented. If system utilization is at or above the threshold, you need to change the hardware configuration.

When defining a workload in a presales situation, the details of a model might not be obvious. Therefore, it might be easiest to develop your general model first and refine it as you learn more details.

You might want to start with a moderate hardware configuration. When defining your workload, after each transaction, you can immediately see the result in the chart and scale the hardware with the transactions. This provides a better understanding of the cost per modeled transaction. However, there is a chart option to view utilization by transaction function to get the explicit cost per modeled transaction function.

When modeling the workload, IBM Content Capacity Planner provides a walk-through wizard for a quick start that helps you to configure the basic parameters of the components that you want to size. We found it useful to use the wizard and save the result to another file. The wizard helps you learn which transaction functions to add to your workload but it creates a simplified model. Some of the lesser used functions can only be obtained by manually adding them to your workload from the Transaction Templates in the tree view.

8.1.1 Example use cases for IBM Content Capacity Planner

Use IBM Content Capacity Planner to help you prepare for the following tasks:

- ▶ A new system is planned, and you need to select the hardware.
- ▶ During a system implementation, the IBM Content Capacity Planner sizing is refined reflecting the latest requirements.
- ▶ An existing system is extended. Additional users and additional applications are rolled out.
- ▶ An existing system needs to be migrated to new hardware. This can occur in conjunction with reorganization and moving into new buildings, system consolidation, new outsourcing contracts, or simply replacing outdated hardware.

- ▶ The current system needs to be analyzed. For example, a client wants to know what additional workload the system can handle or requests a detailed performance analysis. In this case, current production data is available and can be used by IBM Content Capacity Planner.

8.1.2 Capacity planning for new systems

In this section, we list typical questions for sizing a system.

In Chapter 2, “Solution examples and design methodology” on page 17, the following P8 Content Manager solutions were introduced: policy document management process, invoice archiving, email archiving, insurance claim processing, and social enterprise content management. Each solution focuses on a different functionality:

- ▶ Versioning and document management
- ▶ Scanning and processing via a business process and records management
- ▶ High volume ingestion and storage using IBM Content Collector
- ▶ Ingestion, storage, and compliance
- ▶ Social features of IBM FileNet Content Manager in conjunction with IBM Connections

Each system sizing is individual. Avoid the “one fits all” approach after sizing one initial IBM FileNet Content Manager environment. Each solution is built to fulfill defined functional requirements and has a different sizing of required hardware, number of CPUs, memory, disk capacity, and network bandwidth.

We concentrate on general sizing questions. The typical questions to ask the client when preparing to size a system usually fit into the following categories:

- ▶ Client environment
- ▶ Content ingestion
- ▶ User activities
- ▶ Configuring records management
- ▶ Business process management specifics

Client environment

The following list provides questions to ask during sizing that are related to the client environment:

- ▶ Does the client prefer specific hardware? If yes, which vendor?
- ▶ Are there standard machine types that the client wants to use? If yes, what is the standard server, which processor, and how many CPUs?
- ▶ What application server will be used?
- ▶ What database server will be used?

- ▶ What are the default working hours? You can overwrite this default value in each transaction if needed.

Content ingestion

The following list provides questions to ask during sizing that are related to content ingestion:

- ▶ If content is ingested through scanning:
 - What are the scanning hours?
 - What is the average number of scanned documents during the scanning hours?
 - What is the total number of documents usually scanned?
 - What is the average size (in KB) of a scanned document?
 - In how many batches are these scanned documents processed?
 - How many documents are in a batch?
- ▶ If content is ingested through file import:
 - What are the importing hours?
 - What is the average number of documents imported during that time?
 - What is the total number of documents usually imported?
 - What is the average size (in KB) of an imported file?
- ▶ If ingested content is email via IBM Content Collector for Emails:
 - Will original emails be archived?
 - What is the average email size (in KB)?
 - What is the average properties set?
 - What is the number of duplicate email pointers?
 - What is the number of original attachments?
 - What is the number of duplicate attachment pointers?
 - What is the average size of attachments (in KB)?

User activities

After the content is ingested, corresponding actions are started. The content can be processed by IBM Case Foundation or simply stored and used for retrieval later. A user can work on the content using a custom application or FileNet Workplace XT. How the user uses the content might determine the sizing of the system.

The following questions relate to user activities:

- ▶ For logon and logoff activities:
 - How many times does a user generally log on and log off per day or per week?
 - Are there peak hours of logon and logoff activities during the day or during the week?
 - Are there different logon and logoff behaviors for different users (for example, are there different behaviors for power users compared to occasional users)?
- ▶ For search, browsing, and retrieval activities (the same questions can be asked for different user groups):
 - At what times do browsing and retrieval take place?
 - Are there peak hours during the day?
 - Are there deadlines (such as all orders have to be reviewed by noon)?
 - What is the average document size of the documents to be retrieved?
 - How many searches are usually performed per day?
 - How many documents are returned on average per search action?
 - How many custom properties (metadata fields) are retrieved on average per document?
 - How many folders are browsed on average per day by a user?
 - How many folders are accessed via a bookmark?
 - How many documents are retrieved per day by a user?
- ▶ For new document creation:
 - Will new documents be created evenly during the work hours?
 - How many documents on average will be created during the work hours?
 - What is the average document size (in KB)?
- ▶ For check-out and check-in activities:
 - Will check-out and check-in be distributed evenly during the work hours?
 - What is the number of documents checked out and in during the work hours?
 - What is the average document size (in KB)?

Note: After documents are checked out, they usually are viewed. This viewing is modeled as an additional retrieval.

- ▶ For metadata modification activities:
 - Are there major updates of metadata? If yes, in what time frame?
 - How many documents are usually updated during the working hours?
 - Before they are updated, how many properties are retrieved?

Configuring records management

We distinguish records management actions by the Records Manager role and by the users who declare records. Records can be declared through a system step in a business process or manually by users. Ask the following questions when sizing an IBM FileNet P8 records management solution:

- ▶ For Records Managers:
 - What is the logon and logoff pattern of the Records Managers?
 - How many searches for records are performed in a certain time period?
 - How many browse actions in the file plan are performed?
 - How many times are details retrieved? Examples of details are access security, detail, history, holds, and so on.
- ▶ For general users who declare records:
 - How many existing documents are declared as records in a certain time period?
 - How many new documents are declared as records in a certain time period?

Business process management

If the solution involves business process management, ask the following questions for each workflow:

- ▶ What is the time pattern for launching workflows?
- ▶ How many metadata fields does the workflow contain?
- ▶ What is the average field length (in bytes) of the metadata?
- ▶ How many workflows are launched in the time pattern?
- ▶ How many user steps does a workflow contain?
- ▶ How many system steps does a workflow contain?
- ▶ How often are workflow fields updated?
- ▶ How often are users updating their views?

Note: IBM Techline provides dedicated ECM sizing questionnaires that cover all these questions and many more questions for other IBM FileNet products:

- ▶ *Industry Solutions ECM Sizing Questionnaire Oct2012*, PRS5034
- ▶ *IBM FileNet P8 Platform Sizing Questionnaire Jan2012*, PRS3071

8.1.3 IBM Content Capacity Planner output

For every server and certain infrastructure components, IBM Content Capacity Planner produces a utilization chart for one day. The system is adequately handling the workload if the CPU utilization is below 40%. This threshold is used, because response time is exponential, not linear (Queuing Theory).

By sizing the system for 40%, the system can handle temporary peaks with acceptable wait times. Figure 8-2 shows a sample output of IBM Content Capacity Planner with the threshold at 0.4 (40%).

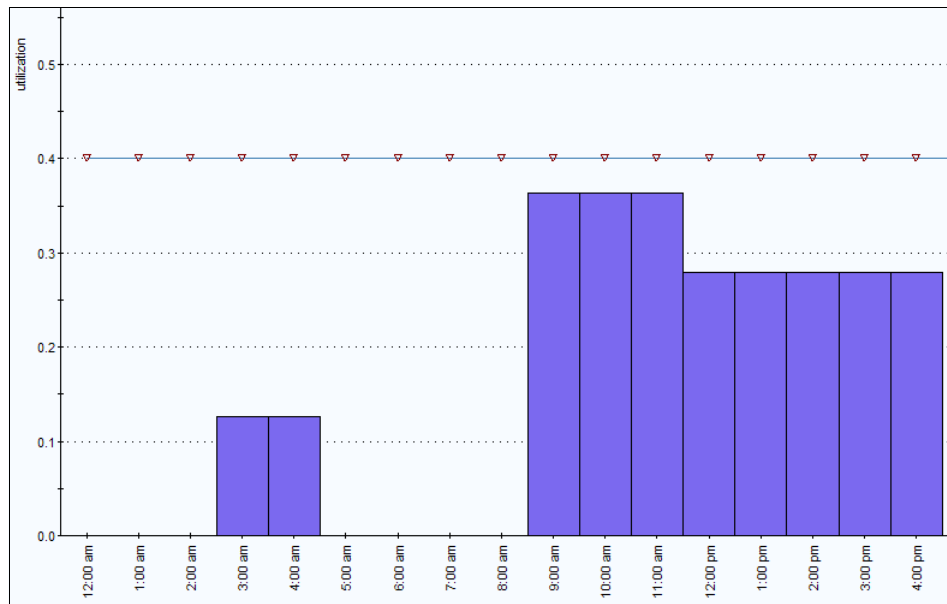


Figure 8-2 Sample IBM Content Capacity Planner output

You can see the Content Platform Engine load throughout the day. In the morning hours between 8:30 a.m. to 11:30 a.m., the system load is higher due to scanning activities. From 11:30 a.m. to 4:30 p.m., the activity level is lower, because only retrieval and processing activities occur. Between 3 a.m. and 4 a.m., prefetching takes place. Documents that are needed for the next day are retrieved and loaded into the cache for better performance.

8.1.4 Predictions from a baseline

When sizing a new system, IBM Content Capacity Planner converts a certain workload to utilization data for a selected, dedicated kind of server. If you are sizing a system upgrade, you already have current data (an existing baseline) available on which you can perform additional modeling. Examples are migration to new hardware, added applications, or added users.

The first step is to collect baseline data for the involved systems. For the Content Platform Engine baseline, you use the System Manager Dashboard. A *dashboard* is a tool for gathering performance data and provides current Content Platform Engine utilization data. If an Image Services system is also involved, data can be exported by the integrated performance data collecting function (`perf_mon`). The baseline data can be imported to IBM Content Capacity Planner, and the utilization data can be used as the basic workload.

Regular capacity planning is important for business continuity. All baseline data must be taken and analyzed on a regular basis. That information is helpful in forecasting upgrades of the actual IBM FileNet environment at the client site. Schedules for upgrades of hardware and software are planned and managed with minimum or no interruption to the production system during normal working hours.

Figure 8-3 is an extension of the capacity planning process. It includes the collection and importation of baseline data from running systems.

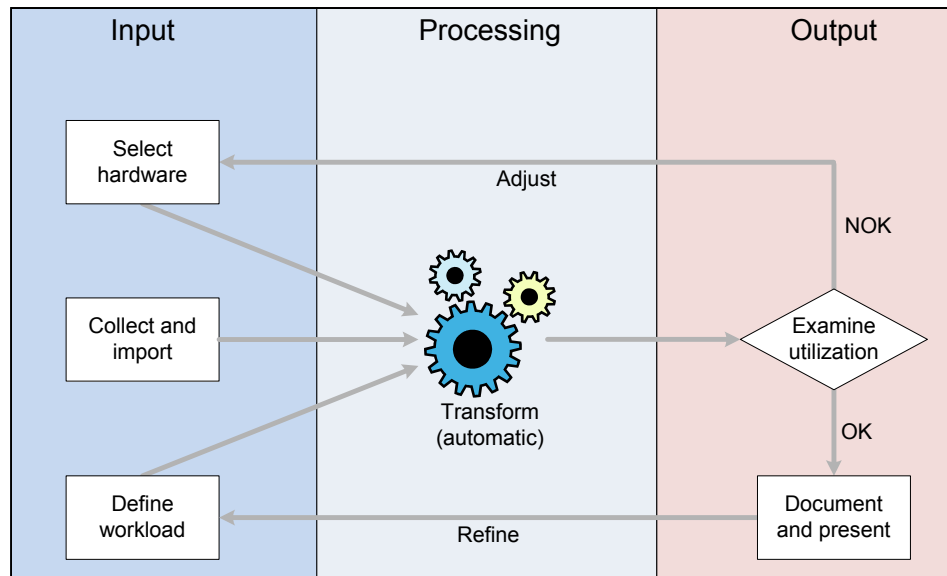


Figure 8-3 Import from a baseline

For example, we show an existing IBM FileNet P8 system, including IBM FileNet Image Services. The client is planning to roll out another application on Content Platform Engine that is expected to double its workload. In addition to that, a third-party application is installed that adds about 20% additional load.

For modeling purposes, we import the current Content Platform Engine utilization with a factor of two, import the Image Services utilization, and add an application that accounts for an increased workload of 20%.

Figure 8-4 shows the utilization for the Image Services system.

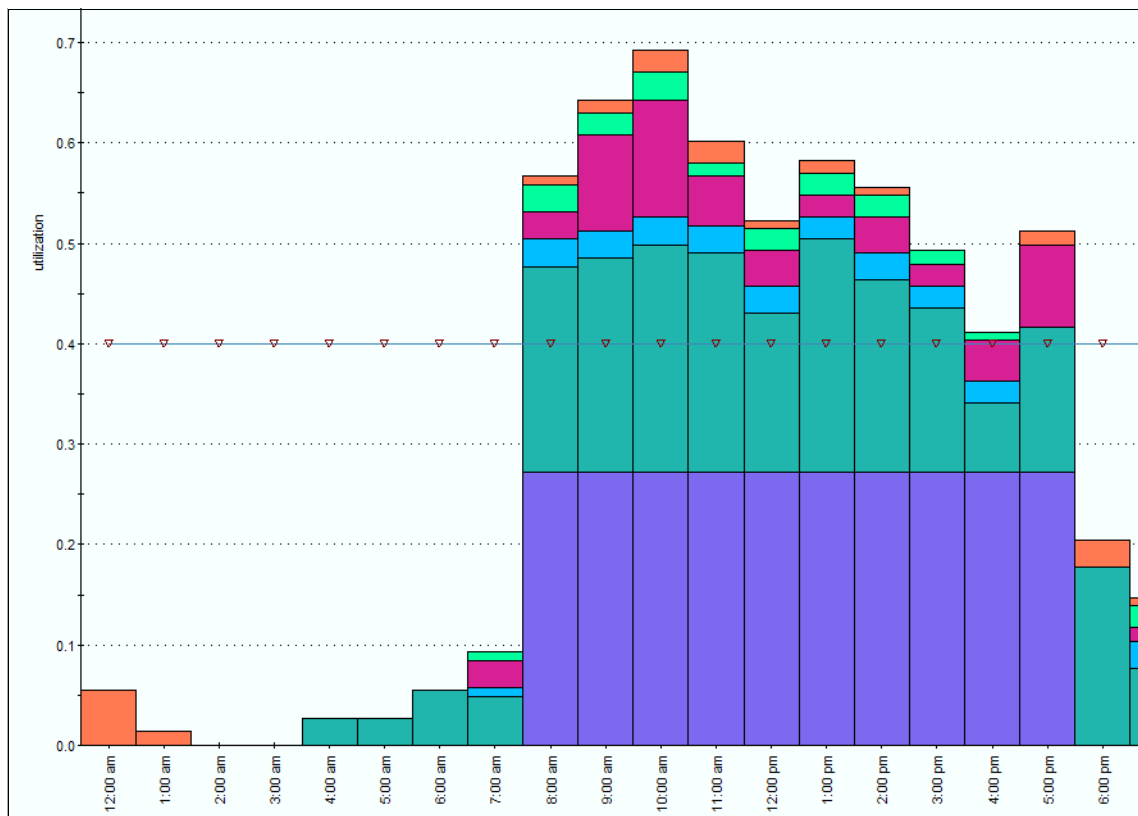


Figure 8-4 Utilization of an Image Services system

The chart shows the workload summary after importing the three workload profiles: one for Content Platform Engine, one for the Image Server, and one for the additional third-party application. The various colors represent single services that run simultaneously. The chart illustrates the imported workload together with the new application workload.

The result is that with the additional application, the Image Services server, exceeds its threshold at 7:30 a.m. It needs to be scaled up with two additional CPUs.

8.1.5 Best practices

The following bullets summarize several recommendations when working with IBM Content Capacity Planner:

- ▶ When initially performing an IBM Content Capacity Planner sizing, the client will not have the exact answer to all of the questions; therefore, make assumptions and document them. Get the clients to sign off on the assumptions used for sizing. Be conservative when making assumptions. Configure the system for peak loads.
- ▶ Add a document to the IBM Content Capacity Planner calculations describing which data was provided by the client, which assumptions were made, and what the IBM Content Capacity Planner output was. Also, document how the IBM Content Capacity Planner input fields were calculated from the data given by the client. This helps you to review an IBM Content Capacity Planner calculation after a certain amount of time and helps you to understand why transactions were modeled in a particular way at a later refinement.
- ▶ Use project variables to ensure consistency throughout your transactions.
- ▶ When you start, you might want to choose medium performance hardware to better see the effects of the configured transactions.
- ▶ If you are unsure about the parameters of a transaction, use the online help. Use the Help topic icon that lists the details quickly.
- ▶ Split a complex scenario into several steps to reduce complexity.
- ▶ After changing parameters, immediately check the output to learn what effect the change has created, which gives you an idea of the costs of the transactions.
- ▶ Common mistakes are defining workload hourly instead of daily (and therefore creating eight times the load) or making mistakes when entering the number of transactions (for example, entering 1,000,000 instead of 100,000).
- ▶ If the system looks misconfigured, change the chart to the Average Utilization view instead of the Transaction Functions view. The Average Utilization view allows you to compare the utilization by function and helps you to localize the function that most influences the system load.

Figure 8-5 shows an example in which the Content Platform Engine is under a heavy load.

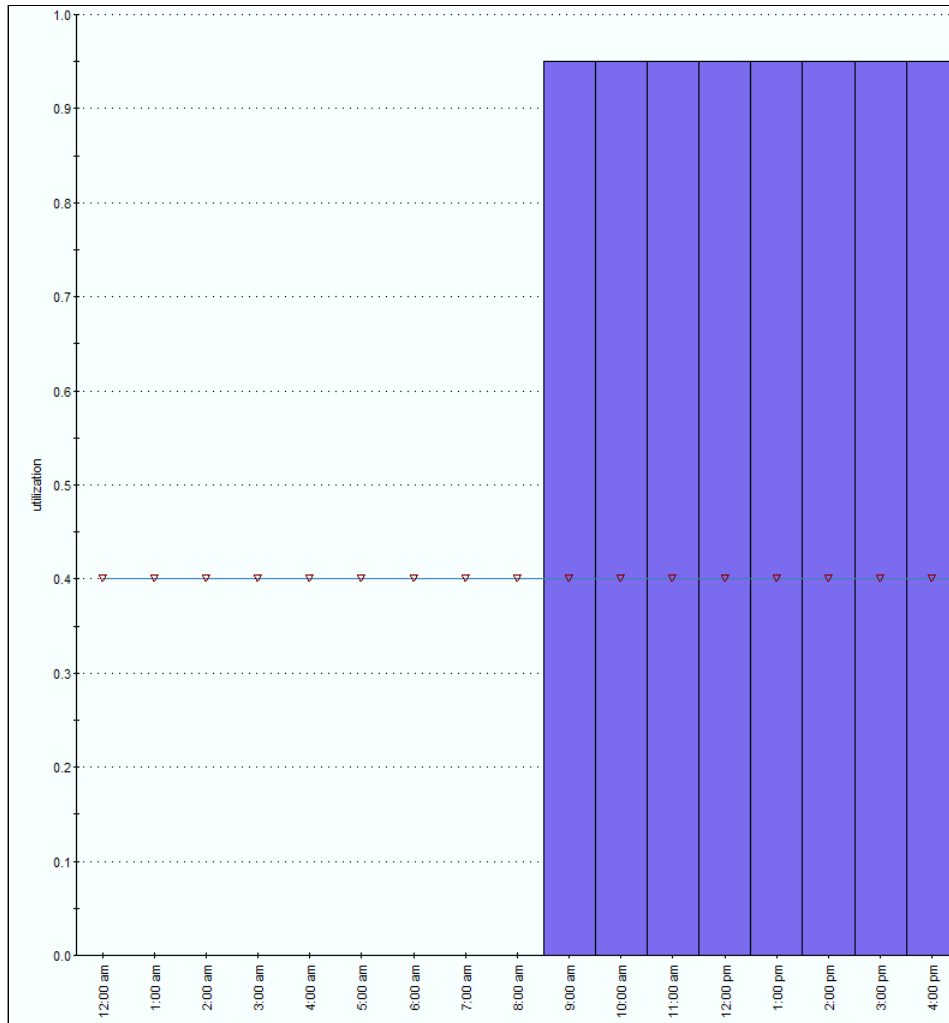


Figure 8-5 Content Platform Engine under heavy load (utilization is more than 90%)

We want to discover what transaction led to the workload. So, we switch to the Transaction Functions view. Figure 8-6 on page 265 shows the result and the transaction responsible for the workload.

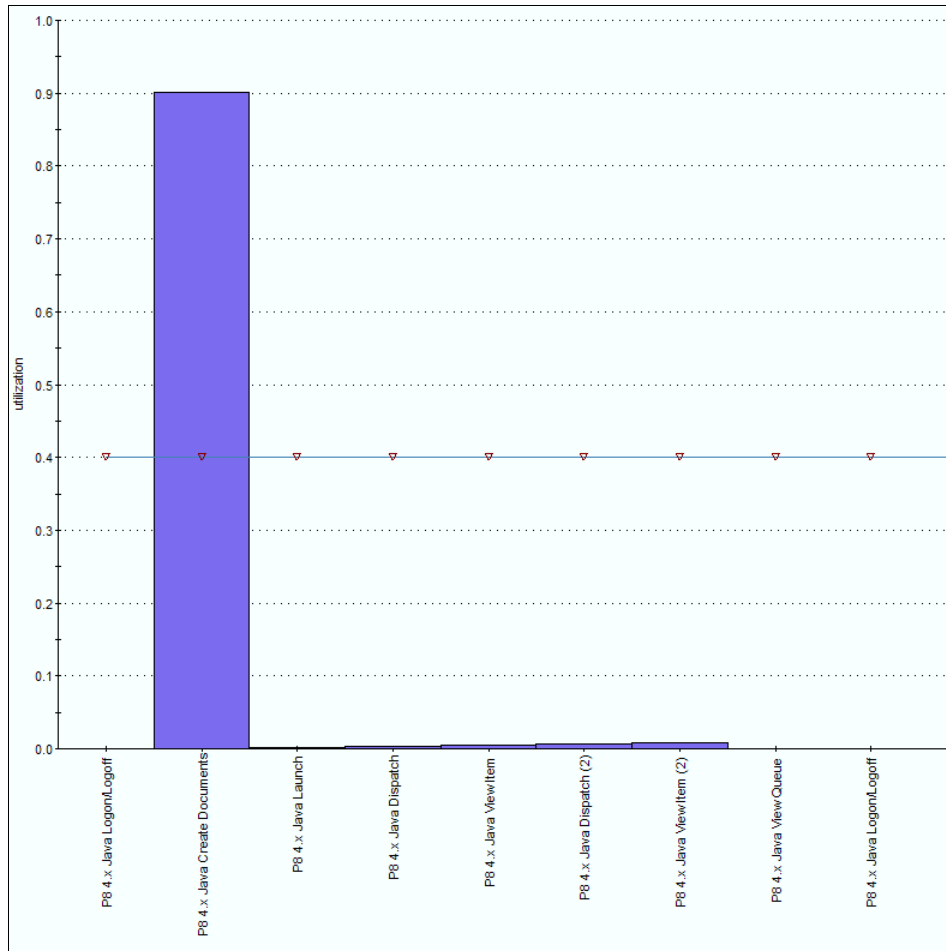


Figure 8-6 Transactions Function view (showing the transactions causing the workload)

As shown in Figure 8-6, we see that the IBM FileNet P8 4.x Java Create Documents transaction creates the most intense workload. When verifying with the system, in this example, we realize a typographical error in the number of input documents and correct it.

With the correction made, we see in Figure 8-7 on page 266 that the system operates well under the threshold.

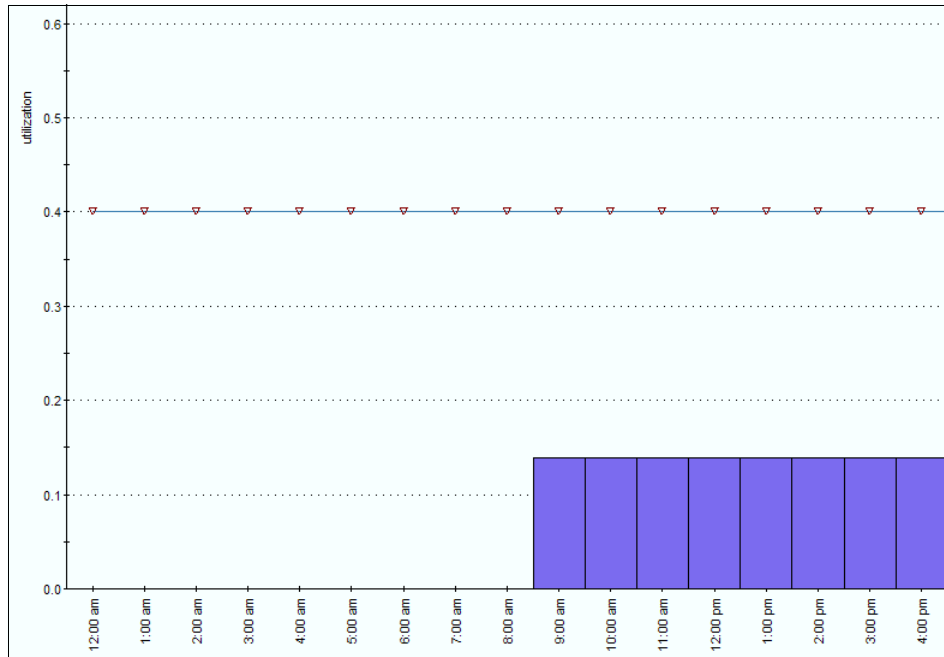


Figure 8-7 Normal system workload

For international users, two troubleshooting tips might be helpful:

- ▶ If you encounter IBM Content Capacity Planner runtime errors, change the regional settings of the operating system to English.
- ▶ Do not use region-specific special characters. If you do, IBM Content Capacity Planner might not be able to open the files and informs you at which line the problem occurs. In that case, you can edit the IBM Content Capacity Planner files (.sct), which are in XML format, to remove the special characters.

8.2 IBM FileNet Disk sizing Tool spreadsheet

In addition to sizing hardware by calculating the utilization, which was derived from a modeled workload, another important point is the sizing of disk space for the managed content. The IBM FileNet P8 Disk sizing Tool spreadsheet enables you to enter key system values, and then, it produces the estimated required disk space.

Figure 8-8 shows an extract of the spreadsheet that contains the input system values and the output, which is the estimated disk space required for IBM FileNet Content Platform Engine and additional components.

| P8 Diskizing Tool for the Content Engine and Process Engine | |
|--|-----------------------|
| Version 2.3 Beta | About |
| Enter values for the sections in green. Appropriate disk sizes will be displayed in the yellow sections. | |
| Assumptions | |
| Global settings | |
| Headroom | 1.25 |
| Search Engine | CSS |
| Concurrent active collections for Search Engine | 5 |
| CSE Maximum Collection Size | 8.0 |
| CSS Maximum Index (Collection) Size Criteria | Max Objects per Index |
| CSS Max Size per Index (Collection) | 100.0 |
| CSS Max Objects per Index (Collection) | 8.0 |
| CSS Object stores for CM | 1 |
| CSS Batch size for CM | 100.00 |
| CSS Threads for CM | 4.00 |
| CSS Object stores for ICC | 1.00 |
| CSS Batch size for ICC | 100.00 |
| CSS Threads for ICC | 4 |
| Index to Content Ratios for various file types | 22% |
| | 20% |
| | 25% |
| | 25% |
| | 50% |
| Content Manager | |
| How many objects associated with content will be stored? | 1.00 |
| How many custom objects will be stored? | 1.00 |
| How many versions of each object will there be? | 1 |

Figure 8-8 Extract of IBM P8 Diskizing Tool spreadsheet

There are also several additional sizing spreadsheets provided by IBM for other IBM FileNet P8 products.

8.3 Performance-related reference documentation

In this section, we provide additional references about where to find performance-related material.

8.3.1 Standard product documentation

This documentation is available for IBM FileNet P8 Content Manager:

- ▶ *IBM FileNet P8 Performance Tuning Guide*

Provides information about tuning parameters that can help improve the performance of your IBM FileNet P8 system. This document covers operating system, database, and application server parameters and IBM FileNet P8 component parameters to help you tune an existing system. You can retrieve this white paper directly from the following website:

ftp://ftp.software.ibm.com/software/data/cm/filenet/docs/p8doc/50x/p850_performance_tuning.pdf

- ▶ *IBM FileNet P8 Performance Tuning*

There are several web pages that provide additional information for improving the performance of IBM FileNet P8 components. Go to the following website:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.performance.doc/p8ppt000.htm>

- ▶ For the latest performance-related documentation and technical papers, go to the product documentation website for the IBM FileNet P8 Platform:

<http://www.ibm.com/support/docview.wss?rs=86&uid=swg27036917>

8.3.2 Benchmark papers

These papers are system performance tests of specific configurations performed either by independent companies or in the IBM FileNet test environment. All these documents are only available for IBM technical sales, IBM lab services, or IBM Business Partners. For more details, contact with your IBM Sales Team or IBM Business Partner.

8.4 Conclusion

This chapter offers a brief look of the concept and process to define and size hardware, network bandwidth, and storage by using IBM Content Capacity Planner. It also offers hints about the input information that is needed to size the environment with IBM FileNet Content Manager.

Note: The sizing with IBM Content Capacity Planner will only be as good as the provided input information.

Now that you have a general understanding of how to plan and lay out an IBM FileNet Content Manager environment, we explore the basic deployment concepts of IBM FileNet Content Manager in Chapter 9, “Deployment” on page 271.



Deployment

In this chapter, we describe the deployment for your IBM FileNet Content Manager solution. We provide advice about how to automate deployment from organizational and technical points of view. We describe the repository design elements and the repository infrastructure components that are part of a FileNet Content Manager solution.

When you read through this chapter, you will understand the deployment of a FileNet Content Manager system.

The chapter will give you the following insights:

- ▶ Overview
- ▶ Deployment by using a formal methodology
- ▶ Deployment approaches
- ▶ Deployment based on cloning
- ▶ Deployment by export, transform, and import
- ▶ FileNet Content Manager deployment
- ▶ Summary

9.1 Overview

Why is deployment important to you?

The most important reason is to ensure consistency in two areas:

- ▶ Ensure consistency in your deployment.
- ▶ Ensure consistency of the metadata model across the environments and in different stages.

Deployment has different technical meanings. For instance, if you talk to your WebSphere Application Server administrator, the administrator will mostly associate deployment with web or enterprise application deployment via IBM WebSphere Network Deployment Manager Console. As another example, in the IBM Case Manager product, deployment refers to the process of migrating and installing an IBM Case Manager solution that was developed in one environment into another environment.

In this chapter, we discuss the following aspects of deployment:

- ▶ FileNet Content Manager data model deployment
- ▶ FileNet Content Manager repository deployment
- ▶ Discuss the technical and organizational dependencies of FileNet Content Manager deployments in general

In the following sections, you will get information that deployment is defined as a collection of related assets used in an application. At the end, the assets will be packaged together and delivered as an application solution. Therefore, we will treat this package as a solution and talk about solution deployment.

Each deployment starts with the following questions and considerations:

- ▶ Which objects need to be deployed?
- ▶ What is the source and the destination?

This chapter describes deployment methods and approaches. It provides details about the tools and features available in the IBM products that can be used in the deployment process.

The deployment discussed in this chapter does not cover content migration, upgrade scenarios, or switching to a different platform. Chapter 11, “Upgrade and migration” on page 371 covers some topics regarding upgrading to the current release of FileNet Content Manager as of this writing.

Note: This chapter assumes that you are performing the deployment.

The next section gives us an overview about technical environments and their organizational role in our deployment strategy.

9.2 Deployment environments

With multiple environments, more people can work on different tasks simultaneously without interfering with each other. For example, you can have an environment for developers to create code, an environment for developers to perform functional testing, and another testing environment for system integrators to test everything. Every company needs a production environment in which only tested and deployed software runs. Development and testing *must* not be done in the production environment.

Synchronizing the various environments becomes a new challenge. You want to make sure that every environment behaves identically after having the same changes applied. This verification ensures that no surprises occur after deploying to the production environment. Of course, because development and even the test environments usually do not have the same hardware as the production environment, performance and load test results typically differ.

9.2.1 Single stage development environment

A single stage development environment is a FileNet Content Manager development system installed manually or by using Composite Platform Installation Tool (CPIT). Refer to 3.1.9, “Setting up a sandbox or demo environment” on page 49 for more information about CPIT.

Typically, this environment is disconnected from the destination environment where you want to deploy your solution. To be able to test deployment to a different environment, you can create additional object stores within the same FileNet Content Manager domain. The main purpose is to perform a *regression test* of your solution before it is deployed to the disconnected destination environment. These additional object stores will simulate the object stores targeted by the deployment process in the destination environment for the solution. The additional object store is often called the *target object store* and the destination environment is also called the *target environment*.

Recommendations: Create at least one additional object store in your development environment to simulate the deployment of your solution to another environment.

Another reason to have several object stores in each environment is so that your repository design objects are separate from your data object store when the application runs. In your design object store, you have no instantiated application or solution data. This is advantageous in that you have no technical constraints by deleting a property or changing repository design objects. The repository where the design objects are stored is also called a *metastore*. For more information about designing a metastore, see Chapter 4, “Repository design” on page 81. IBM Case Manager maintains a metastore automatically for you and is part of the solution design process that is seamlessly integrated in the solution deployment model. In IBM Case Manager, the design object store acts as the metastore.

For more information about IBM Case Manager, see this website:

<http://www.ibm.com/software/advanced-case-management/case-manager>

9.2.2 Multi-stage deployment environments

The term *environment* in this section describes a collection of servers that typically belong to one FileNet Content Manager domain for one particular purpose. The purpose can be development, regression testing, acceptance testing, load testing, performance testing, or production.

Typical projects split their infrastructure into at least three environments:

- ▶ Development
- ▶ User acceptance, testing, and quality assurance
- ▶ Production

Recommendations: Development must not be done in the *same environment* as the production site or test site. Segregating these activities in different environments avoids the introduction of unwanted configuration changes or code changes by developers before those changes are ready to be tested or put into production. Furthermore, we highly advise that you use the same Lightweight Directory Access Protocol (LDAP) foundation across all stages except the development environment. We discuss the security later in “Exporting user and group information” on page 306.

While trying to isolate phases of the software development cycle into different environments, the complexity of maintaining different stages becomes challenging.

Note: Maintain deployment in an organized manner to ensure consistency in any case.

Larger companies tend to add these additional environments to the basic three environments identified earlier:

- ▶ Performance testing
- ▶ Training
- ▶ Staging

You might add more environments for the following reasons:

- ▶ A need to mitigate risks associated with multiple projects running at the same time interfering with each other, while retaining the ability to reproduce errors from the production system in a test environment
- ▶ A need for multiple training environments so that many people can be educated in a short period of time

The more environments that you have, the more important it is to maintain and synchronize them correctly.

The segregation of environments by the FileNet Content Manager domain is a best practice. The isolation achieved by this approach is optimal to allow people to work simultaneously and independently on the same project but in different phases without adversely affecting each other. In particular, giving each environment its own FileNet Content Manager domain makes it easy to grant domain-wide permissions in each environment to different groups. For example, developers can be given full permission to configuration objects in the development environment but no permission to configuration objects in the production environment.

The next section provides guidance to set up a formal deployment process before you start using the IBM FileNet Deployment Manager utility in later sections.

9.3 Deployment by using a formal methodology

Establish clear guidelines and common processes for the deployment to ensure that you have the ability to consistently deploy an application. The explanations and graphics in this section will help you to achieve this goal.

In this section, we focus on the following areas of common process management to establish a common understanding of deployment in a software development project:

- ▶ Release management
- ▶ Change management
- ▶ Configuration management

Figure 9-1 provides an overview of a standard software development process.

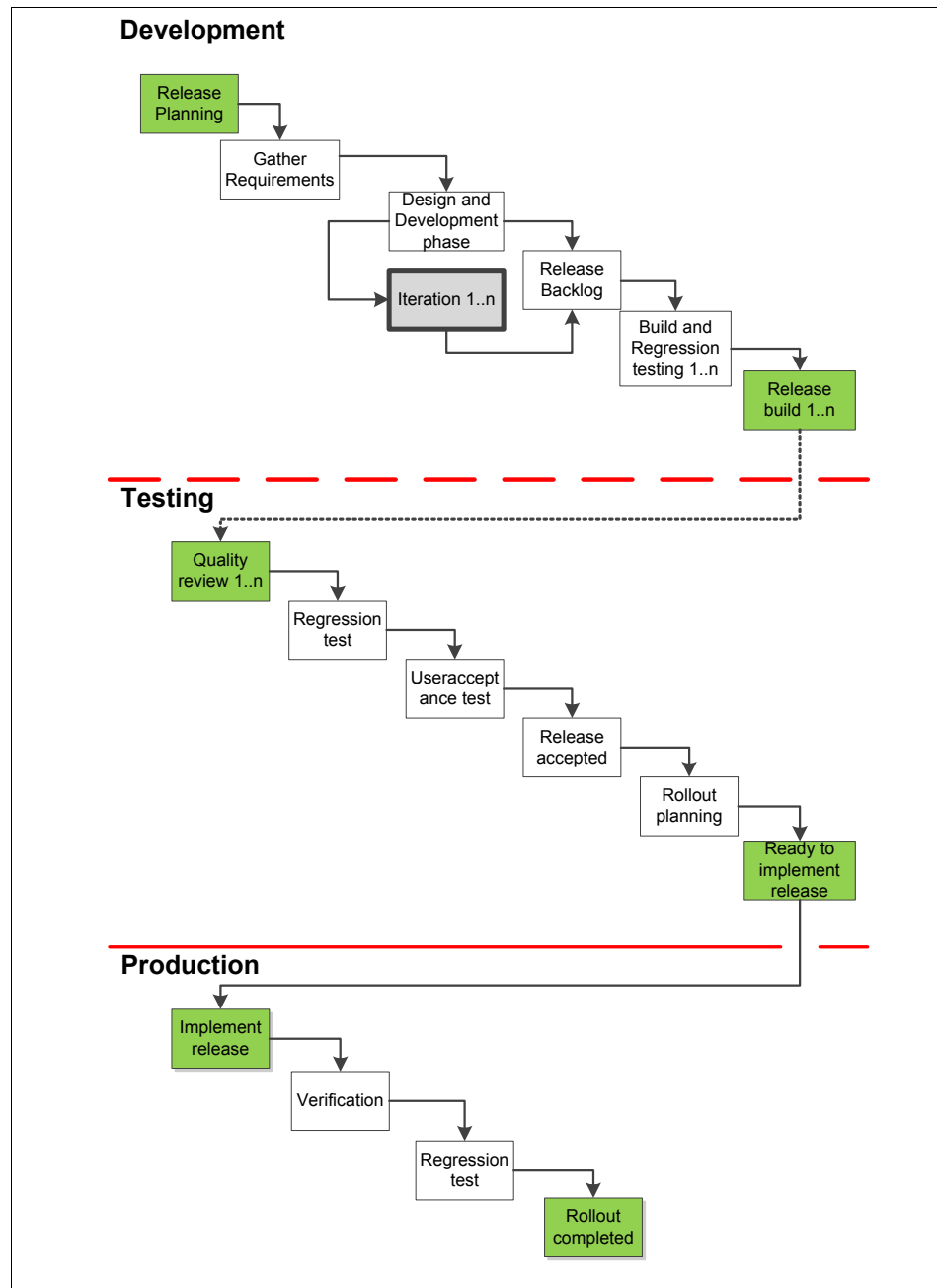


Figure 9-1 Overview of the processes for software development

Figure 9-1 on page 276 shows three phases of a software development lifecycle: development, testing, and production. Each phase can correspond to one or more individual environments. This lifecycle is the groundwork for our deployment strategy.

Note: Development or change running directly on production without using the staging-based approach is a *bad practice* and can cause loss of consistency.

In each phase, the regression testing has a different focus. In development, your automated regression test is done on your local resources. You cannot map these results to another environment. You will get a statement about the functional verification test (FVT) in development. In testing, you can perform system verification testing (SVT) with load and performance tests to get the business requirements verified and qualified. In production, a short functional verification test is typically performed by starting the application and creating test data that has no impact on the production environment and must be reversible.

Recommendations: In each environment, always create test data in a dedicated storage area that has no special requirements for retention or security and that is separate from the production storage. You must sanitize the production data that is used as test data for FVT/SVT in each environment.

The illustration in Figure 9-1 on page 276 also shows that a change management process is indispensable.

How do you document deployment requirements in a change request as the deployer? As a best practice in typical client situations, you include at least one spreadsheet with multiple worksheets inside that are added to your change management system. This deployment sheet allows you to track the changes that you made across the stages. It also provides detailed history about the implementation schedules and results. We agree that it is absolutely crucial to have a good change management process in place be able to track the same level of detail also for the non-production environments.

In the next sections, we present more details about release, change, and configuration management, as well as testing, before we dive into a discussion of moving the applications from development to production. To understand how deployment works, it is important that you understand how a software development process works.

9.3.1 Release management

Today, it is popular to run component-based architectures (Java EE and .NET applications) and service-oriented architectures (SOA). These approaches transform the enterprise into a highly interoperable and reusable collection of services that are positioned to better adapt to ever-changing business needs.

For example, typically, you can implement a web service that is reused and called by different components of our applications. This can be also a workflow that was exposed by using a WebSphere Service Registry and Repository.

As architectural approaches lead to more reuse and separation, the development of enterprise applications continues to require well-defined processes and more tiers of technology. As a result, certain areas of enterprise application development increase in complexity. In enterprise development (for example, Java EE and .NET), software vendors have made many efforts to reduce this complexity. They provide advanced code generation and process automated tooling and simplify complex aspects of enterprise development through the use of proven design patterns and best practices.

Note: It is important to understand the business and technical requirements and to implement them in a systematic and formal way. In a deployer role, you often only get to work with the technical requirements.

One way to meet this challenge is by introducing the role of a *release manager*. In most large companies, this role becomes crucial for large-scale deployments. A release manager is not always one person and can be implemented as a team with diverse technical skills. Additionally, some tasks are distributed to different teams within the company or to external partners as a result of the release management process.

A software release manager is responsible for handling the following tasks and requests:

- ▶ Risk assessment
- ▶ Deployment and packaging
- ▶ Patch management (commercial or customized bug fixes)
For example: operating systems, application servers, and FileNet Content Manager fixes
- ▶ From the software development area:
 - Software change requests (modifications)
 - New function requests (additional features and functions)

- ▶ From the quality assurance (quality of code) area:
 - Software defects of custom code/commercial code
 - Testing (code testing)
- ▶ Software configuration management (the rollout of new custom application releases, hardware, and supplier software upgrades)

Select one or more of these roles and match them to job functions in your company to manage deployment.

Later sections of this chapter describe the use of IBM FileNet Deployment Manager. It is important to understand and create a sequence of activities first as a part of the deployment planning.

In general, *release management* relates to the features and functions of the software; how the software is designed, developed, packaged, documented, tested, and deployed.

A solid release management process can produce the following documentation:

- ▶ Project plan
- ▶ Release notes
- ▶ Test matrix, test plan, and test results
- ▶ Installation scripts and documentation
- ▶ Support documentation
- ▶ User documentation
- ▶ Training material
- ▶ Operations documentation

The following documentation and information can help the release manager for a FileNet Content Manager solution perform release management tasks:

- ▶ Hardware and software compatibility matrices from all involved vendors of your solution.
- ▶ Release notes, technotes, and the latest fix packs with their descriptions.
- ▶ Available export and import options to deploy the solution between development and production environments.
- ▶ Search and replace scripts used to prepare exported assets for use in the target environment where object stores, users, or groups differ from the source environment. Tools required for needed data transformations.
- ▶ Deployment guidelines in the IBM FileNet P8 Information Center
- ▶ Online help

In addition, the IBM Rational® product line from IBM can be helpful in supporting release management, change management, and testing:

<http://www.ibm.com/software/rational>

Release management delegates several of the underlying support processes to the change and configuration management that is discussed in 9.3.2, “Change management” on page 281 and 9.3.3, “Configuration management” on page 286.

A *release* can consist of multiple components in specific configurations of the involved components. Release management handles the validation of combinations of application releases, commercial components, customized components, and others. While a specific component is developed on the basis of a concrete version of its underlying commercial application programming interface (API), at the moment of deployment to production, this combination might have changed in the bigger context of our solution. The management of combinations of versions of involved components is a time-consuming activity and needs to be scheduled and planned carefully and early.

Another aspect of release management deals with objects that have been created in production that affect the configuration of the solution and might affect deployment. In FileNet Content Manager solutions, these types of objects include folder structures, entry templates, search templates, and others. Release management must have a strategy in place to handle or restrict bidirectional deployment between multiple environments.

Recommendations: Have a strategy in place to handle or restrict changes to the production environment that might affect the overall solution configuration and future deployment. Have a policy that all application changes must be made first in development and test environments, then deployed to production.

Typically, a bidirectional deployment is only defined between a development environment and a testing environment.

We talked about the role of release manager and their organizational tasks in a FileNet Content Manager deployment process. The release manager works with the change coordinator that we describe in the next section of this chapter.

9.3.2 Change management

In most organizations, change management is the process of overseeing, coordinating, and managing all changes to the following areas:

- ▶ Hardware
- ▶ System software
- ▶ All documentation and procedures associated with running, supporting, and maintaining production systems

In the previous section, we discussed the role of the release manager. In this section, we describe the change coordinator role.

With FileNet Content Manager solutions, there are two important points associated with change management:

- ▶ The number and details of configuration items needed for a proper deployment
- ▶ The creation of an impact analysis to evaluate the effect of the deployment on the satellite systems (Table 9-1)

Table 9-1 Shows a sample impact analysis worksheet

| Type | Component | Name | Task description | Tool | Package | Notes |
|------|-------------------|-------------------|----------------------|-------------|---|---------------------------------|
| A | Property template | MyProperty1 | Remove default value | FDM | 004_SRC_TGT_A _ Properties_ YYYY_MM_DD | FDM import option always update |
| B | Stored procedure | MyProcedureA B | Replace | DB2-CLI | 004_SRC_TGT_B _ MyProcedureAB_ YYYY_MM_DD | Delete all previous versions |
| C | Web service | MyWebService 3 | Deploy | WAS console | 004_SRC_TGT_C _ MyWebService3_ YYYY_MM_DD | Uninstall old web service first |

Table 9-1 shows an example communication between the change coordinator role and a FileNet Content Manager solution development team. It shows information that is needed to create change requests that relate to different deployment types. In this sample sheet, the change coordinator first identifies the type of deployment used. This worksheet indicates that three IT departments (A,B, and C) of the company will potentially be affected by the entire master change.

The package column shows an example of a naming convention:

```
DEPLOYNO_SRCSYSTEM_TARGETSYSTEM_TYPE_NAME_TIMESTAMP
```

With this naming convention, the change requester, if a complaint is made, can determine the package that caused the query. The owner of the deployer role can find the associated log files and change comments. For more information about naming conventions, see 4.3, “Repository naming standards” on page 87.

You create one master change request with three different tasks assigned to your three different departments with their responsibilities and schedules. The *order of execution* was previously identified by the development team.

Tip: Ensure that the dependencies of the deployment types are mapped to your change management system.

What are the advantages of separating deployment into different types?

You can see that type A is the job of the IBM FileNet Content Manager administration department. Type B is the responsibility of the database administration department. And, type C is the department that works with application server infrastructure. The dependencies between them might be that a type A deployment can only happen if a type C deployment occurred first.

With this layer-based approach, you can delegate responsibilities to those departments with the appropriate skills. These entities usually have their own process model to implement such changes. At the end, you receive a well-documented and detailed change. This approach improves quality and accuracy due to the involvement of many people, the “many eyes” principle.

Let us look at the change management process in relation to the different software development phases. When the development system is not part of your change management process, the following situations can occur when changes are applied to the development environment without being documented or in an uncontrolled manner.

Recommendations: Every change applied to the production system must be carefully tracked, documented, and, where possible, automated. Automation of changes reduces the risks of error-prone manual deployment processes. Every bidirectional change from test to development must also be strictly documented because in a typical three-stage environment, the test environment becomes the master system for production from the deployment point of view.

If you inspect the data in Table 9-1 on page 281, you see that some areas *outside* of FileNet Content Manager are affected but still belong to our FileNet Content Manager solution.

Consider the following areas related to FileNet Content Manager when managing the change process:

- ▶ Commercial code and assets (versions of FileNet Content Manager, as well as individual patches and levels of its add-ons, such as IBM Case Manager)
- ▶ Custom code and assets (for example, the versions of the application leveraging a commercial API, such as the FileNet Content Manager API, or versioned assets, such as FileNet Content Manager code modules)

Recommendations: Deployment starts in the development phase. Incorporate a defined build process that acknowledges changes to commercial components and custom components in a controlled manner. A build process can be established with commercial products, such as Rational, or a manually defined set of process steps and shell scripts that build the custom application. As the deployer, work together with your development team to implement a common build process.

In software development companies, there is often a department called the *build team*. Investigate whether you have a department that can assist you in creating a build process.

At the beginning of the deployment of every custom application or FileNet Content Manager deployment by using IBM FileNet Deployment Manager, we advise you to handle commercial code and custom code separately. For the targeted solution release, everything must be assembled via an automated process if possible in observance of the dependencies.

Note: Always keep an integral point of view on the deployment process. If you follow these guidelines, you will treat deployment as a solution.

Figure 9-2 on page 284 shows the areas for which you distinguish between custom code (red area) and commercial code (green area):

- ▶ IBM Content Navigator
- ▶ FileNet Content Manager

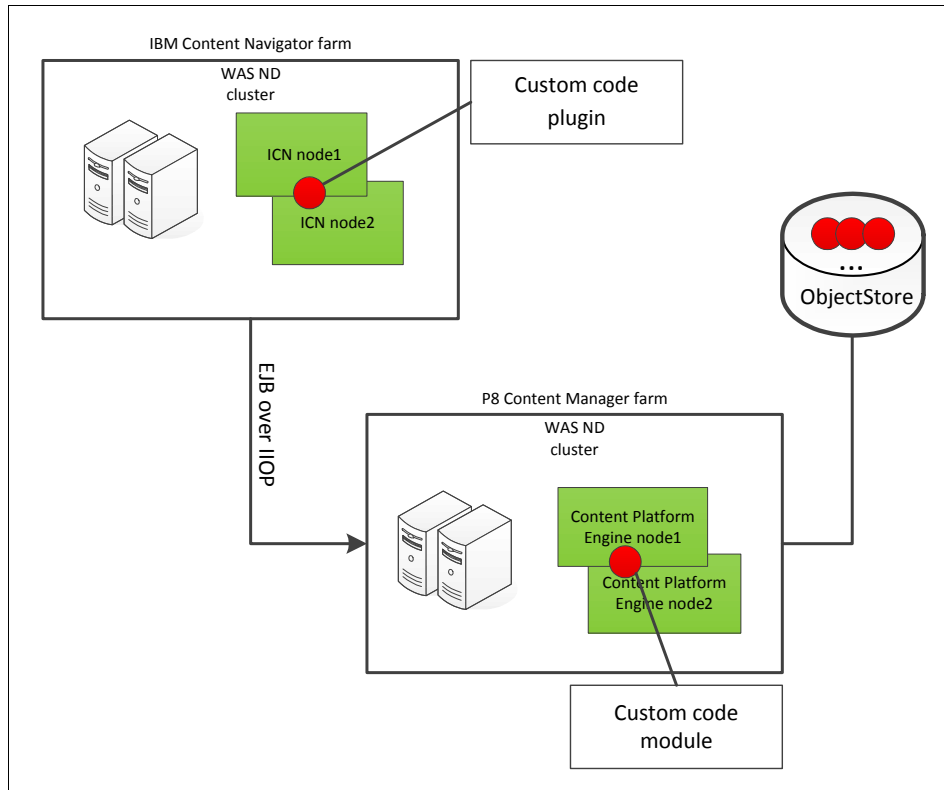


Figure 9-2 Custom code at the level of object store and IBM Content Navigator

Figure 9-2 shows you that a custom application is dependent on the commercial code but treated separately. Part of code works inside the application but has dependencies to parts of the commercial code to make a functional verification successful. Therefore, it makes sense to declare deployment into at least two or three layers with different technical and organizational responsibilities. To determine whether the custom code plug-in inside of IBM Content Navigator has been deployed correctly, conduct a regression test by calling the plug-in inside IBM Content Navigator. The functional verification is done by the business department that belongs to the application layer, not by the owner of the implemter or deployer role.

A regression test of a custom code module in FileNet Content Manager determines whether a new version in FileNet Content Manager has been generated for the code module after running the related IBM FileNet Deployment Manager import package.

Figure 9-3 on page 285 shows the layer-based deployment approach.

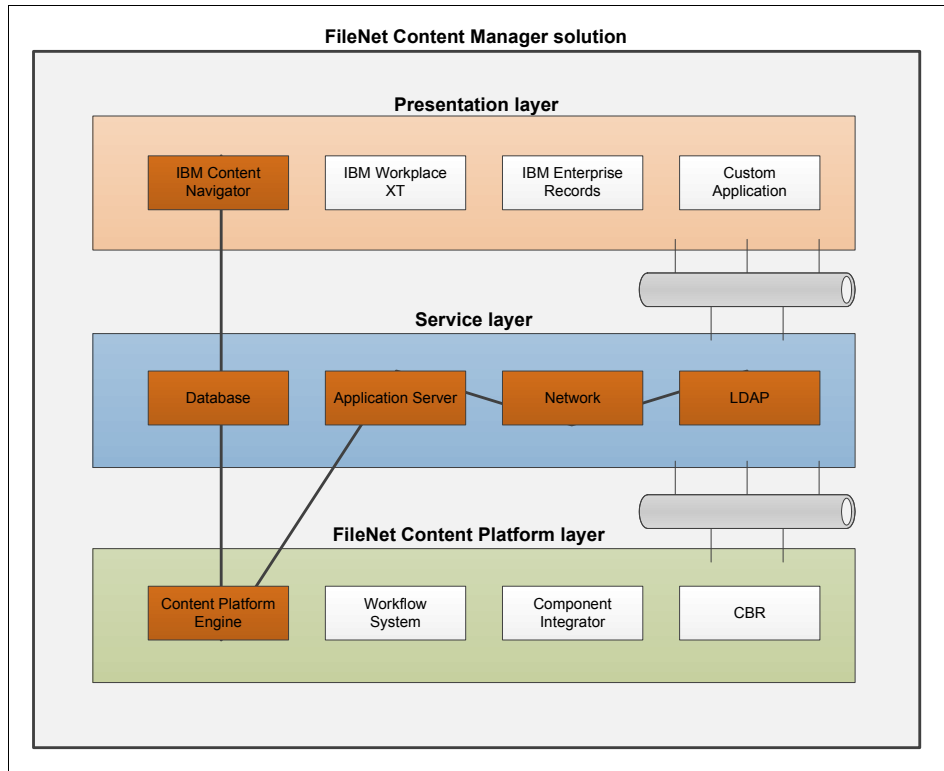


Figure 9-3 Layer-based deployment approach

Note: From an architectural point of view, this diagram might differ from your environment in distinguishing between the back end and the front end.

This three-layer approach shows the differences of deployment types and also the different departments with their responsibilities for the solution. The solution contains the *presentation layer* (the application tier as shown in Figure 9-2 on page 284) where IBM Content Navigator is running. The IBM Content Navigator depends on a *service layer* where the underlying configuration database is deployed. The service layer also provides the support for the FileNet Content Manager *base layer* where your FileNet Content Manager is running. In our example, it is IBM WebSphere Application Server Network Deployment. Each layer has its own deployment type and process.

For more information about the separation of custom code and commercial code and the build process for IBM Content Navigator, see 9.7.5, “Exporting and importing other components” on page 311 and *Customizing and Extending IBM Content Navigator*, SG24-8055. For more information about the separation of custom and commercial assets during repository design, see Chapter 4, “Repository design” on page 81.

In summary, a layer-based approach is necessary to split responsibilities, reduce risk, and improve the quality of the IBM FileNet Content Manager solution. These layers represent different change requests based on deployment types that are determined by the development team and documented in a common worksheet. A layer-based approach also shows the complexity of deployments.

To speed up the change requests and reduce their complexity, it is essential to automate. The next section describes tooling that can be used to achieve useful configuration management.

9.3.3 Configuration management

Typical FileNet Content Manager projects use three environments, but many projects use five or more environments to satisfy the diverse needs of development, training, testing, staging, performance measuring, and production. Every environment has its own set of configuration items, such as server names, IP addresses, and versions of the various components (commercial and customized).

While an enterprise configuration management database might not be suitable to track all parameters needed for the deployment process, it is the responsibility of configuration management to track applied changes.

From our experience, when performing automated deployments for FileNet Content Manager-based applications, it is generally a good practice to employ a centralized datastore. The centralized datastore tracks the specific values of parameters:

- ▶ Object store name
- ▶ Object store Globally Unique Identifier (GUID)
- ▶ Directory objects’ prefix per environment
- ▶ Virtual server name of Content Platform Engine farms
- ▶ Virtual server names of Application Engine farms
- ▶ Database names
- ▶ Database server names
- ▶ Ports

These parameter values can be used by the build process for specific environments. It makes sense to keep track also of the access control entries (ACE) from an access control list (ACL) of a FileNet Content Manager repository design object.

Retain a zip/tar file of all release-specific data, including code, exported assets, and documentation, in a central datastore. Typically, you maintain release-specific data by using a code version control system. IBM clients can use IBM Rational ClearCase®, for example.

Recommendations: Implement a central datastore that tracks the parameters, such as GUIDs, object store names, and project names, that you need for the deployment. The datastore needs to be implemented for all target environments in one location that is accessible to every environment. The best way is to use this datastore via FileNet Content Manager API to automatically track changes and specifications vigilantly.

9.3.4 Testing

There are multiple ways to address environments associated with testing. One way is to split testing into two major phases that typically happen in different environments:

► Development environment

In this environment, the following tests are commonly conducted:

- Unit testing verifies that the detailed design for a unit (component or module) has been correctly implemented.
- Integration testing verifies that the interfaces and interaction between the integrated components (modules) work correctly and as expected.
- System testing verifies that an integrated system meets all requirements.

► Testing environment

In this environment, the following tests are commonly conducted:

- System integration testing verifies that a system is integrated into the external or third-party systems as defined in the system requirements.
- User acceptance testing is conducted by the users, customer, or client to validate whether they accept the system. This is typically a manual testing process with documented expected behavior and the tested behavior.
- Load and performance testing.

Recommendations: Whenever a software system undergoes changes, verify that the system functions as desired in a test environment, before deploying to production. Include time and resources to test and make corrections based on testing whenever planning and scheduling a software release. Applying this best practice without fail helps avoid costly and time-consuming problems in production.

Regression tests

For all environments, automated regression testing must be implemented to be able to verify earlier test cases. In production, you mostly perform a *smoke test*, which means testing only the most critical functions of a system. The automated regression testing suite might include from each relevant aspect one test object, such as a test document class, a test search template, a test folder, and a test workflow. In the deployment of repository design objects, it is best practice to deploy repository design objects on the same environment in another object store by using IBM FileNet Deployment Manager utility.

The regression test must be used after having modified software (either commercial or custom code) for any change in functionality or any fix for defects. A regression test reruns previously passed tests on the modified software to ensure that the modifications do not unintentionally cause a regression of previous functionality. Regression testing can be performed at any or all of the previously mentioned test levels. The regression tests are often automated. Automating the regression test can be an extremely powerful and efficient way to ensure basic readiness. The implementation of automated regression tests is time-consuming and the test cases must be adjusted every time a change occurs in the business functionality.

Recommendations: Establish a small suite of automated regression tests in each environment. The best synergies are achieved by having the deployment of the test assets and the test script as automated as possible. One side effect is that this automation of regression tests affects the repository design and you must be able to revert changes.

Test automation

Two areas of consideration for automating tests are available:

- ▶ Load and performance test
- ▶ Regression test

While the load and performance test might be executed only on major version changes (commercial or custom releases), the effort to maintain the code for the automation might be substantial.

Hint: It is essential that you test with production data. Have a process in place to make production data anonymous before you start testing.

If you use the datastore to track all repository design objects, you can revert those changes that were made by your regression suite.

Recommendations: Ensure that your testing environment or performance load environment is preloaded with test data. The data inserted by your regression suite must simulate the amount of data generated by average concurrent users to get a baseline performance matrix.

The regression test must be generic enough so that the scripts are written once, and maybe updated if there are minor changes, but typically stay pretty stable over time. This approach can be reached by developing a regression test framework. We advise that you store the scripts together with the supporting version of the test application in one location. Typically, this location is your source code version control system.

Test automation tools are available from IBM and other vendors. For example, see the IBM Rational products website:

<http://www.ibm.com/software/rational>

Recommendations: Distinguish load and performance tests from regression tests. Each area has its own characteristics.

You can typically use the existing testing infrastructure for load and performance tests. For regression testing, it makes no sense to use a centralized large and complex infrastructure. It is more important that the tests can be executed and quickly show simple results.

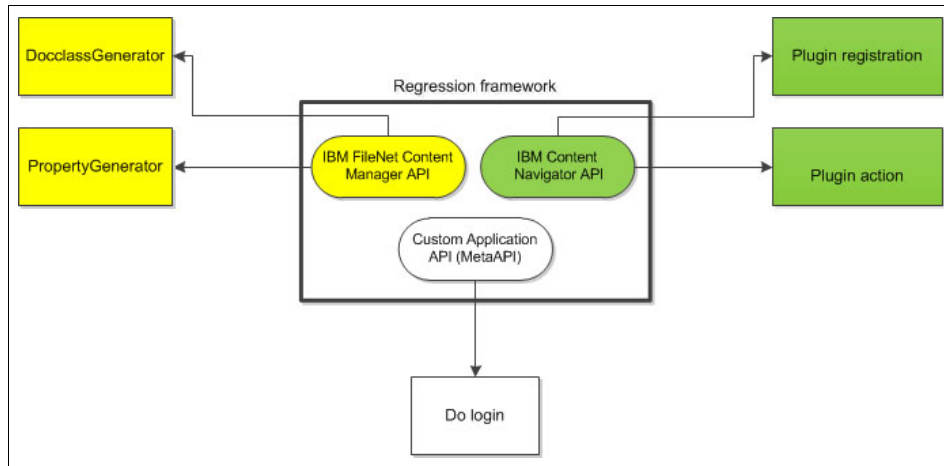


Figure 9-4 Generic framework of reusable regression tests

Figure 9-4 shows an example of a regression framework that can be used to perform basic regression testing of a FileNet Content Manager solution. A framework approach enables more flexibility and reusability during iterative development phases. The framework can be automated by using a script interface and the configuration information is provided by the central datastore.

Your regression suite can contain smoke test procedures.

Test documentation

Before we move to a discussion of the actual deployment, we must discuss the testing documentation and its importance.

In a FileNet Content Manager project, multiple departments with different skill sets are involved. It is difficult to perform user acceptance testing or integration testing without a clear concept of what needs to be tested, how it must be tested, what the expected behavior is, and how the tests must be conducted. This documentation is derived by the requirements analysis of the business needs.

Documenting the test cases with descriptions of the inputs and expected behavior is useful. Test descriptions must have enough information to achieve repeatability, which means that multiple testers can perform the same test (in an identical environment) while working from the test documentation and get the same results. After the execution of the tests, collect and document all of the observed system behaviors. Using this information, the release manager can decide to proceed with the new release or to delay the release if there are more bugs to fix.

Several of the tests might fail. It is crucial to document the behavior but also the resolution. The *knowledge base* must be part of problem management. Combining test documentation with a searchable interface to find known problems is advantageous.

In FileNet Content Manager and authorization, the detailed, different levels of security must be tested by using an intersection of security principals. More information about security is in Chapter 5, “Security” on page 151.

Recommendations: Carefully document your tests with sufficient detail before the tests are executed. Make the test documentation database searchable to search for problems previously seen by users. Create a knowledge database and publish it on a social media platform, for example, in an internal wiki.

The social media platform is available from IBM and other vendors. For example, try out IBM Connections. For IBM Connections information, see this website:

<http://www.ibm.com/software/lotus/products/connections>

9.4 Deployment approaches

The term *deployment* is typically used in two contexts when mentioned in combination with FileNet Content Manager solutions:

- ▶ In a broad sense, the term includes all of the activities that are needed to move from one entire environment to another environment.
- ▶ In a stricter sense, the term describes the actual execution. Less frequently, this type of deployment is called *migration* or *transport*. It describes the typical automated process of export, convert, and reimport of application-related items, such as application code, configuration settings, and repository assets.

Most of the implementation preparation work occurs in the development environment. In the development environment, the IBM FileNet components, the Java EE and .NET applications, and configurations provide an exportable blueprint for the same configuration that is used for testing and the quality assurance environment. The security configuration can be different and needs to be mapped separately.

There are at least three preferred practices to deploy (transport) changes from one environment to another:

- ▶ Cloning (AIX logical partition (LPAR)-based cloning or VMware based cloning)
- ▶ Exporting, converting, and importing using the IBM FileNet Deployment Manager utility
- ▶ Using scripted generation of all the necessary documents and structures

9.4.1 Cloning

You can deploy changes from one environment to another by cloning the source environment and bringing it up as a new but identical instance of the source environment.

Cloning is practical when you temporarily need a dedicated environment. It must be an exact copy of what you have already in place, for example:

- ▶ In a training class, you need to be able to quickly revert or go forward to a well-known working environment (by a teacher over lunchtime) for the next class or for the next part of the lectures. (A few students might not be able to follow the exercises and then are unable to continue with the rest of the class because their environment has not been set up correctly.)
- ▶ Many parallel identical training environments are needed to educate more people in a short period of time.
- ▶ Development environments are needed to work in parallel.
- ▶ Test environments are needed for specific tests.

You can use local VMware based images to clone a system. For a large system, however, this might not be a workable solution. Large systems are often not as flexible as small systems, or there is a lack of powerful machines that can be made available in a timely manner for cloning. Sometimes, the security and networking policies do not allow these virtual environments to connect to back-end machines.

Note: IBM AIX LPAR-based cloning and IBM AIX workload partition (WPAR)-based cloning are alternatives to VMware based cloning. The cloning methods that you use depend on your infrastructure. For more information, see *IBM FileNet P8 Platform and Architecture*, SG24-7667.

The next logical step is to use virtual farms that host applications at larger client sites. This approach might not be practical for the following reasons:

- ▶ From the corporate network, they cannot be accessed unless using remote desktop applications. A direct interaction is not possible due to using the same host names and IP addresses multiple times in the same network.
- ▶ Single virtual images are typically not powerful enough for the full stack of components that are needed for a solution (the stack includes the directory server, database, application server, and other FileNet Content Manager components).

A good way to still rely on virtualization techniques is described in 3.3.1, “A virtualized IBM FileNet Content Manager system” on page 60. The solution is built on individual images hosting the various IBM FileNet P8 components, including the database and directory server. The images are accessed by a gateway, which shields the network topology of the FileNet Content Manager solution from the corporate network by using network address translation (NAT) and virtual private network (VPN) access.

You clone an environment by copying all files representing the storage of virtual images. With this approach, you can clone an environment within hours with little knowledge. Use this approach predominantly for development and training.

For cloning, consider the following topics:

- ▶ Action plan (execution order, manual tasks, and script environment)
- ▶ Virtualization technology (native virtualization and para-virtualization)
- ▶ Storage technology (logical volume mirroring)
- ▶ Post-cloning activities (host name and network preparation)

9.4.2 Custom-scripted export, transform, and import

A common way of deploying an environment is the process of exporting, transforming, optionally installing, and importing. A key advantage of this approach is that it allows you to carry forward incremental changes from the source to the target environment without requiring the recreation of the entire target environment. The major difficulty with this approach lies in the number of dependencies between different components and the number of manual steps that are needed to achieve a target configuration that matches the source configuration.

In the past, we have seen projects struggle for months when using a manual process to move Java EE applications that include FileNet Content Manager components. Today, we can deploy similar projects in 1 - 2 days. The following factors contribute to the improvements:

- ▶ Introduction of a solid release management process

- ▶ Separation of commercial code from custom code and automation of the build process mainly for Java EE based or .NET based applications
- ▶ Adherence to the proposed guideline of stable GUIDs to reduce dependencies
- ▶ Implementation of a central datastore (database-based or file-based) in which environment-specific information is stored and serves as the datastore for scripting
- ▶ Automation wherever possible

Deployments typically apply assets from the development environment to the testing environment. The production environment typically receives deployments from the testing and acceptance test environment.

There are cases for which you might consider the reverse:

- ▶ Documents with configuration characters, such as search templates, add entry templates, custom objects, and folders, that have been created in a production environment. (This raises the question of what a release needs to contain and restrict.)
- ▶ Hot-fixing a serious production problem in the staging area.
- ▶ Refreshing and populating a training environment or acceptance test system with anonymized or obscured production data.

For example, you can use IBM InfoSphere® Optim™ Data Lifecycle Management Solutions to anonymize or obscure production data:

<http://www.ibm.com/software/data/optim>

- ▶ The activity for transformation can take place as described in *Deploying IBM FileNet P8 applications* in the IBM FileNet P8 Information Center, before or just after import. Custom scripts can be called to make the necessary transformation. The transformation can also be conducted on the exported files before importing. The IBM FileNet Deployment Manager utility has a script interface. This script interface allows you to run a code fragment on each object before or after import. Furthermore, it allows you to run a script once before or after import. You can combine all four of them. For more information about the IBM FileNet Deployment Manager utility, see the IBM FileNet P8 Information Center where you can find an example of creating marking sets during import.

9.4.3 Scripted generation

This approach assumes that after a basic object store has been generated, every property template, document class, folder, and other structural asset is generated by a script.

This approach has been proven to work, but the effort to maintain this type of script is huge and every change must be put into the script code. All of the benefits of using a tool, such as the IBM Administration Console for Content Platform Engine (ACCE) (or IBM FileNet Enterprise Manager, are lost with this approach. There is little benefit in using this approach unless it is to overcome limitations where there is no alternative.

Nevertheless, this approach can be used to create basic structures, such as a P8 domain, create generic object stores with their add-ons, marking sets, folder structures, or maintain application roles.

For object stores and their add-ons, you can customize the schema script that is used to generate the object store tables over Java Database Connectivity (JDBC). The advantage of customizing the script is to create custom indexes. With a customized object store add-on, you can create specific metadata for your custom FileNet application.

The script-based approach has a problem because every change within the FileNet Content Manager master system needs to be synchronized with the script source. You can develop a wrapper to perform this work but developing a wrapper can be time-consuming. In the next sections, we describe other FileNet Content Manager population techniques.

9.5 Deployment based on cloning

One of the biggest challenges to making an environment clonable is the system dependencies that cannot be easily removed, such as the dependencies related to a Microsoft Active Directory. Changes might take time to implement and can impede the cloning process.

Figure 9-5 on page 296 illustrates that each source LDAP can be replicated to its own instance and serve as the dedicated LDAP for your FileNet Content Manager environment.

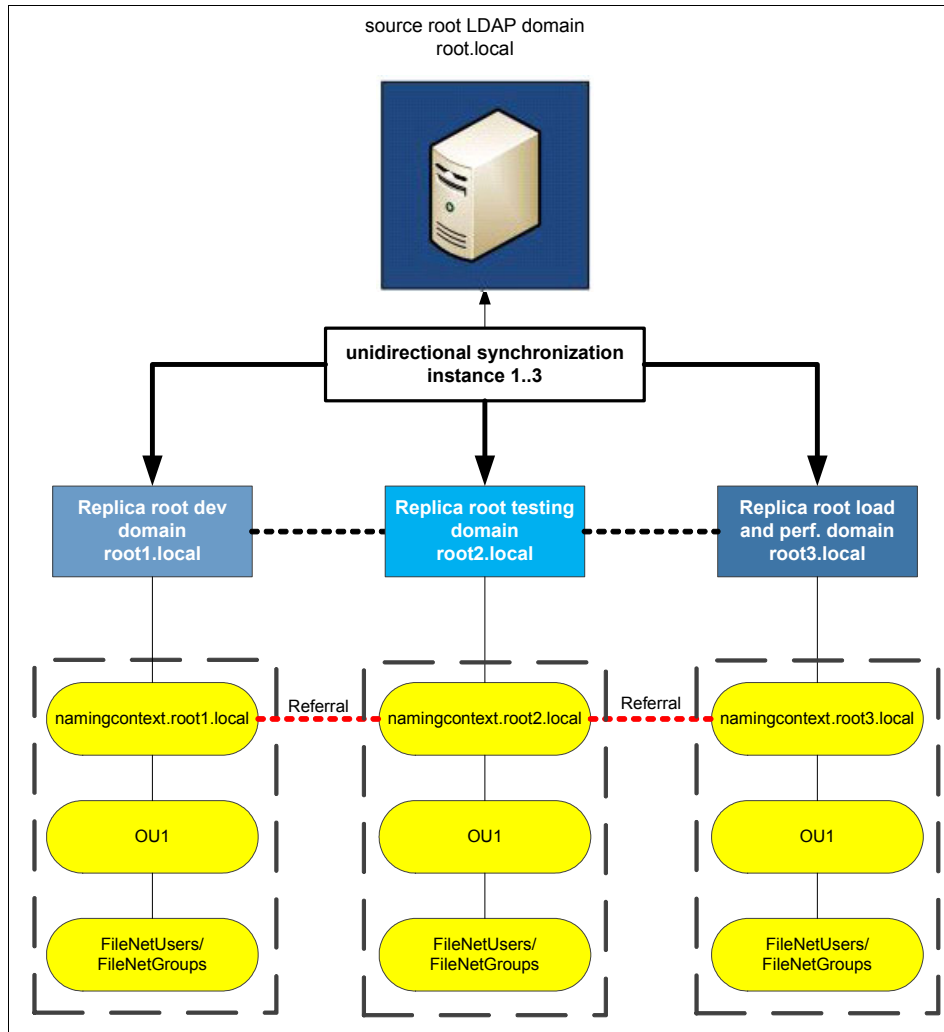


Figure 9-5 Example of LDAP replicas

Figure 9-5 shows using IBM Tivoli Directory Integrator or the Active Directory Lightweight Directory Service replicating from a Microsoft Active Directory. Additionally, you can enable referrals in each naming context to have a distinguished name (DN) available that does not exist in the current naming context. Referrals are not problematic if you have UserPrincipalNames (UPN) enabled in FileNet Content Manager across the naming contexts due to the unique short name requirement.

Recommendations: When introducing FileNet Content Manager to your company, use a dedicated LDAP provider for FileNet Content Manager. This can be achieved by using IBM Tivoli Directory Server or Active Directory Lightweight Directory Service (AD LDS) with unidirectional synchronization, if necessary, and illustrated in Figure 9-5 on page 296. This gives you the ability to maintain the security principals by yourself and reduces organizational dependencies to other departments significantly.

9.5.1 Cloning an object store

Follow this manual procedure to clone an IBM FileNet Content Manager object store where the IBM FileNet Deployment Manager reassign object store functionality is not available.

To clone an object store, follow these steps:

1. Create an empty object store database with minimum requirements.
2. Create a data source on your application server pointing to the empty object store database.
3. Create in the target P8 domain an object store that has the same name as the object store from the source P8 domain. (Now, we have created a container in the global configuration database (GCD) for our object store from the source P8 domain that will later be “partially” incorporated.)
4. Shut down IBM FileNet Content Manager.
5. Bring the source object store database online.
6. Change the data source defined in step 2 to match the database defined in step 5.
7. Start IBM FileNet Content Manager.

This procedure *does not* preserve the GUID of the source object store and can be used to incorporate an object store from a different IBM FileNet Content Manager P8 domain. If the source P8 domain uses a different naming context, the security objects *might not* match with your target P8 domain.

When this procedure is completed, you can delete the object store database that you created in step 1.

9.5.2 Topology

Figure 9-6 illustrates a clonable topology with three identical environments that use VMware images. Every domain is formed by a collection of servers that are part of multiple VMware images. All images of one domain are connected over a private network to a special image that is called the *router*. The router implements network address translation (NAT) and virtual private network (VPN) gateway functionality by using tunneling over Secure Shell (SSH) or other products. The other network link of the router is mapped to a network card that has access outside, which can be to the corporate network.

To clone the environment, only the router image needs to be modified and the public interface needs to be set up correctly. An IBM Content Navigator instance resolves the Domain Name System (DNS) of the router image.

Note: This approach is useful for development environments only.

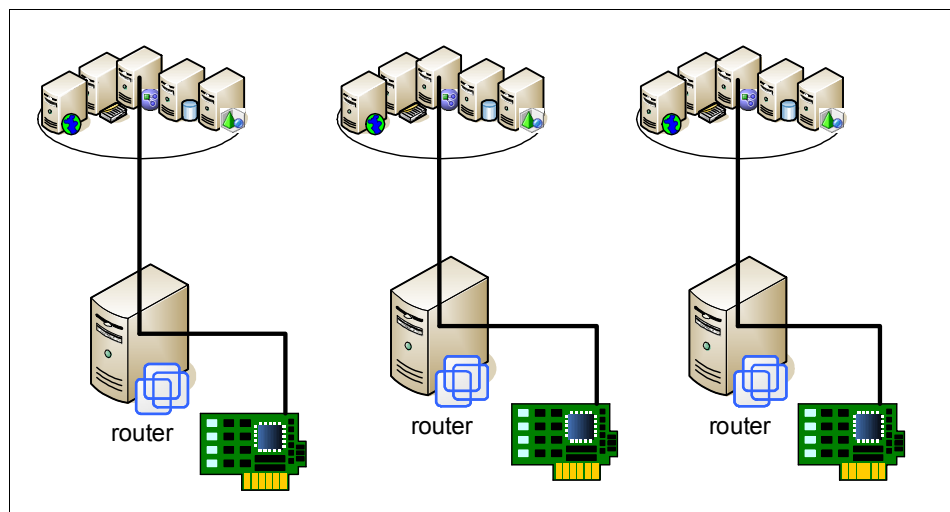


Figure 9-6 Three identical environments using VMware images

Figure 9-6 shows how an access to a FileNet Content Manager domain can be established by using a separate gateway for each environment.

Cloning offers these advantages:

- ▶ Automatic provisioning of environments by using a master clone
- ▶ Less effort to create a FileNet Content Manager environment

Cloning has this disadvantage:

- ▶ The clone system is static with a fixed version and patch level since it is based on a master clone that was created at one point in time.

9.5.3 Access to the environment

There are two clients:

- ▶ The user's workstation
- ▶ A development system hosted on VMware running on the user's workstation

Even though a large group of developers might have all of the tools necessary to perform their tasks, developers might prefer a pre-configured image to run on their individual workstations. If Microsoft Active Directory is used, use a VMware image that was initially part of the same Active Directory.

9.5.4 Post-cloning activities

After cloning this environment, consider resetting passwords and generating new users and groups for the project. These tasks need to be automated as much as possible.

9.5.5 Backup changes

There is no real benefit in backing up a cloned environment because it can be created again easily with less effort.

9.6 Deployment by export, transform, and import

In this section, we discuss deployment by export, transform, and import either for a full or incremental deployment.

9.6.1 Incremental deployment compared to full deployment

The two major types of deployments are a full deployment and an incremental deployment. A *full deployment* for a FileNet Content Manager solution means that both the structure information and the content are deployed in *one iteration*. The target environment gets everything with the assumption that the target object store is empty.

An *incremental* deployment for a FileNet Content Manager solution means deploying only the changes that have been made since the last deployment. Documents, custom objects, and other objects might have already been instantiated. New changes to the structure must respect the associated constraints.

Before you run the *import*, you must choose one of the following options:

- ▶ Update If Newer
- ▶ Always Update
- ▶ Never Update

The system property `DateLastModified` is used by the IBM FileNet Deployment Manager utility to determine whether if an object has changed. This information is required if using the “Update If Newer” option.

Full deployment is a powerful vehicle to move a project the first time through the various stages of deployment. A project includes creating test deployment systems, and, possibly, multiple production systems. So, you can perform multiple full deployments for the project. For test systems, you might clear them out and fully deploy them again. For the production environment, you only perform a full deployment *one time* for your project.

After populating a production object store with documents, it is impractical to perform a full deployment to the production object store for the following reasons:

- ▶ The number of documents that you need to move from production into development and then propagate back is typically too high.
- ▶ Security restrictions often prevent us from moving production data to other environments.
- ▶ The production system cannot be stopped during the time that it takes to create the next release.
- ▶ The duration for moving documents is much longer than just applying structural changes.
- ▶ There are documents created in production that have configuration characteristics, such as search templates and entry templates. If you perform a full deployment, you must also move them back to the development environment and propagate them back.

An *incremental deployment* means the propagation of changes that will transition an environment from a certain status (existing release) to a new status (new release).

There are multiple ways to figure out the differences between the two releases:

- ▶ Manually
- ▶ By strictly rolling forward changes from the source environment to the target environment and preventing any changes to the target environment between releases
- ▶ Automated discovery of the differences

Manually detecting the differences between the source environment and the target environment is time-consuming and error-prone. This option is only valid for small deployments or if the difference is related to only one asset type, such as an instance of the custom object class.

Clients typically choose the second option with the consideration that someone has manually verified both environments. In a multi-stage environment, there is a good chance that mistakes in this approach will be detected in the first deployment step from the development environment to the test environment. When errors are detected at this point, there is an opportunity to fix the underlying problems and retry the same procedure. As soon as the deployment to the test environment passes testing (and is documented), the future deployment to production most likely works smoothly.

The third option is extremely difficult to achieve and potentially too expensive. There are many exceptions when just comparing date times between the various environments. A development or source environment might include more objects than will be used for the target deployment. So, a selective tagging of objects, which are part of a release, seems to be mandatory.

9.6.2 Reducing the complexity of inter-object relationships

GUIDs reduce the complexity of inter-object relationships. A Globally Unique Identifier (GUID) is a 128-bit data identifier, which is used to distinguish objects from each other. If there is a need to merge objects from multiple object stores into a common object store, GUIDs ensure that the objects maintain their uniqueness and that they do not collide with each other in the common object store.

When moving objects between multiple environments, you must consider dependencies. Objects are often dependent on other objects in the object store or on external resources, for example:

- ▶ An *entry template* references a folder and vice versa. In FileNet Workplace XT, you can associate entry templates to a folder. This type of dependency is automatically considered by IBM FileNet Deployment Manager.

- ▶ An application's stored search definition is an XML document in an object store. The XML content references multiple object stores by name and ID. This type of dependency is automatically considered by IBM FileNet Deployment Manager.
- ▶ A document references an external website that contains its content. This type of dependency has to be maintained manually.

FileNet Deployment Manager can convert some external dependencies, such as URLs that are referenced by form templates.

For inter-object store dependencies, you can keep the identification of these objects (GUIDs) consistent across various environments. This is not in contradiction to the previously mentioned uniqueness of GUIDs, but it is a consequence for two reasons:

- ▶ The objects, which are considered to be kept consistent with the same GUIDs across object stores, have configuration characteristics, such as document classes, folders, property templates, entry templates, search templates, and others.
- ▶ The predefined population of an object store after you run the object store creation wizard follows the same pattern.

In Figure 9-7 on page 303, we show two options to deploy a search template that has a dependency on a folder structure. Although you might argue that there are better ways to reference folders by referring to a full path, you might discover similar situations where there are good reasons to depend on a GUID. In the first option, we followed the practice of using stable GUIDs, which we did not follow in the second option:

- ▶ Deploying the folder with the same GUID leads to no additional corrections deploying the search template above.
- ▶ Deploying the folder and letting the system generate a new GUID leads to a situation where the search template must be changed to refer to the deployed folder.

You can avoid the extra effort of maintaining the dependencies in the target environment by following the pattern of having stable GUIDs.

Figure 9-7 on page 303 illustrates the deployment from development with stable GUIDs in the top box on the right. Not following the stable GUID pattern might require maintaining the dependencies with additional deployment logic.

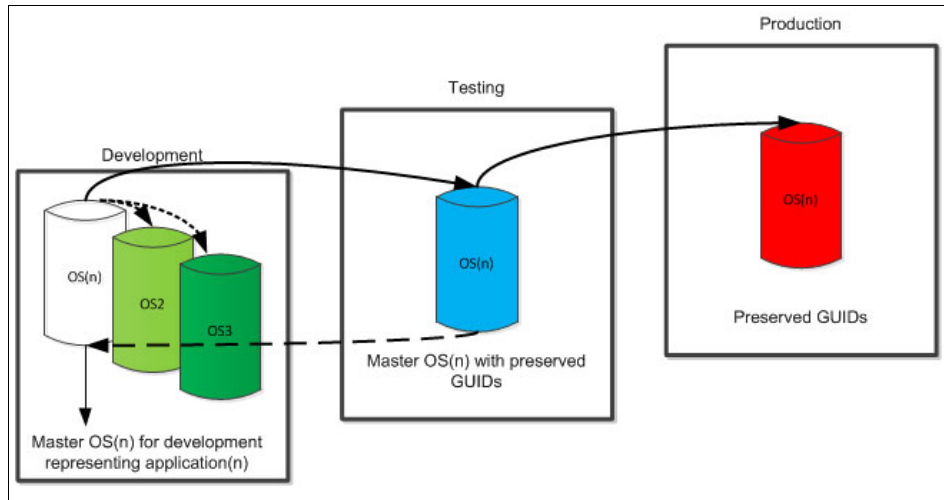


Figure 9-7 Example of using stable GUIDs

If you use IBM Forms, you must use stable GUIDs across all stages, especially for the version series IDs of workflow definitions.

9.6.3 Deployment automation

The more environments that you need to maintain, the more important it is to automate the deployment to achieve the following goals:

- ▶ Save time.
- ▶ Reduce errors.
- ▶ Reduce risks.
- ▶ Ensure similarity among environments.
- ▶ Reproduce problems.

The IBM FileNet Deployment Manager utility contains a command-line interface to perform this automation. Examples are in “Automate FileNet Deployment Manager operations from the command-line interface” in the IBM FileNet P8 Information Center.

9.7 FileNet Content Manager deployment

The process of deployment by using the IBM FileNet Deployment Manager utility is described in “Deploying IBM FileNet P8 applications” in the IBM FileNet P8 Information Center on the IBM FileNet support site.

Important: The use of IBM FileNet Enterprise Manager for performing FileNet Content Manager deployments is deprecated. Use the IBM FileNet Deployment Manager utility instead.

Tip: The IBM FileNet Deployment Manager utility has no customizable query interface so you need to use FileNet Enterprise Manager to perform a custom query and put the results into an export manifest. This export manifest, which is generated by FileNet Enterprise Manager, can be used in IBM FileNet Deployment Manager to perform the remaining export and import steps, including conversion.

The IBM FileNet Deployment Manager utility offers a consistent way of performing deployments across different stages and environments. It can be used to deploy data between disconnected FileNet Content Manager systems even if these systems use different LDAP providers and different security principals.

It offers a powerful mapping interface with automatic mapping if the symbolic name of the object store or the short name of a security principal is the same.

We need to know which data can be used for export.

There are three major types of objects to be exported:

- ▶ Structure (such as document classes and folders)
- ▶ Configuration documents (such as templates, subscriptions, events, custom code modules, and workflow definitions)
- ▶ Business documents (images)

Configuration documents do not contain business content but contain configuration information that is used by an application. Configuration documents might need a transformation step before being deployed to the target environment, because they might hold information about dependencies. The majority of configuration documents are automatically converted to match the target environment by the IBM FileNet Deployment Manager utility.

Note: Workflow definitions will be treated as content elements but not automatically transferred to the workflow system. You can use the post-save script interface of IBM FileNet Deployment Manager to transfer them to the workflow system by using a Java script.

Business documents contain business information and are viewed by users. Business documents typically do not need transformation when they are deployed, because they have no internal or external dependencies.

9.7.1 FileNet Content Manager export

When preparing an export, you need to consider the granularity of the export. It is usually a bad approach to include everything in one export run. If you need to fix a problem, addressing smaller chunks of data rather than one huge XML file makes troubleshooting easier.

Known successful deployments use the following practices:

- ▶ Break the deployment apart into a hierarchy of exports.
Example: Document class definition with or without its property definitions
- ▶ Strictly separate configuration documents from business documents.

Recommendations: Reduce the complexity of exporting by splitting a large export into smaller logical chunks. Separate structure and configuration documents from business documents.

Hierarchy of exports

Build a logical hierarchy of exports, which can help you to test the imports sequentially.

Configuration documents, such as entry templates, are documents that include content elements and they are stored as an XML file in the repository. Document class definition is a database object with no content elements. Exporting a document class definition with its property definitions is a simple transaction, whereas exporting a configuration document involves exporting the content elements.

Certain objects, which include all FileNet Content Manager domain-level objects, cannot be exported.

There are Application Engine-related objects that cannot be exported as well. See 9.7.5, “Exporting and importing other components” on page 311.

There is a general export and import sequence:

1. Marking sets
2. Choice lists
3. Property templates
4. Security templates

5. Class definitions (including document, custom object, folder, and so on)
6. Entry templates
7. Stored searches
8. Class subscriptions
9. Event actions
10. Code modules
11. Documents (configuration documents and real documents)

This list is incomplete. We outline the sequence in order to explain the dependencies. There are other workflow system-related assets, such as workflow event actions, workflow subscriptions, and workflow definitions, that are not mentioned here. For more detail, see “Deploying P8 applications” in the IBM FileNet P8 Information Center.

Recommendations: Consider the hierarchy of objects and their dependencies by importing them in an order that dependencies can be resolved manually. Inspect the “import options” for each asset in IBM FileNet Deployment Manager utility to find out which dependencies are automatically resolved. If you repeat an export/import, sometimes it does not make sense to include the dependencies in the export package again.

Exporting content

Usually, content is versioned. For configuration documents, it does not make sense to export “all versions”. The most recent version is the best choice, so as a rule, discard earlier versions. If you import them to the target system, keep only one version.

Exporting user and group information

When deploying between environments that use different directory servers or different contexts of one directory server, user and group information must be mapped to the target environment. User and group information is contained in the access control lists that are present and that control the access to almost every exported object. User and group information might also be contained in object-specific fields of certain types of objects (for example, entry templates and workplace user preferences). The user and group information must be set correctly before the actual deployment occurs.

Recommendations: If you deploy across different environments, do so in the same LDAP context. However, for large environments, the testing and load-performance environments typically do not use the same LDAP as the production environment for security purposes. In this situation, use separate replicated LDAP servers for testing.

It is time-consuming to map changes in users and groups manually, because the exported assets have different security in the source and in the target. To overcome this issue, the IBM FileNet Deployment Manager utility offers the use of a “label file” where short names of the security principals from source and target are mapped in advance. In most cases, the initial security on configuration objects does not change often and you are using LDAP groups to maintain security. Another method is to retrieve principals from the target LDAP by using the half map file of the source system. In this case, you need to have identical short names for automatic mapping.

The label file approach has two advantages:

1. Mapping security principals from a disconnected development environment to a testing environment is possible. You have to know which target LDAP principal is matching each source LDAP principal. With this method, you can reassign group A to group B even in the same LDAP.
2. No full LDAP query is performed.

If you deploy data within the same LDAP context, you will choose to retrieve from the half map file to obtain the security principals from the target system. This procedure also does not need to retrieve the entire LDAP.

Automating the export

The IBM FileNet Deployment Manager utility has a command-line interface to generate deployment packages for the three types of deployment data.

Automating the export is not limited to exporting FileNet Content Manager objects. There are other tools available for exporting workflow system-related data. For a list of export and import utilities, see 9.7.4, “IBM FileNet Deployment Manager” on page 309.

9.7.2 CE-objects transformation

Various exported assets need a different treatment for a successful import into the target object store. Table 9-2 shows a partial list to give you an idea of how to distinguish the options.

Table 9-2 FileNet Content Manager assets and transformation

| Type of asset | Transformation required | Remarks |
|--------------------|-------------------------|----------------------------|
| Property Templates | Not required | Import with the same GUID. |
| Choice Lists | Not required | Import with the same GUID. |

| Type of asset | Transformation required | Remarks |
|----------------------|-------------------------|---|
| Document Classes | Not required | Import with the same GUID. |
| Workflow Definitions | Required | Import with the same GUID. Contains references to environment-specific constants, such as Object Store Name and external references for web services. |
| Folders | Not required | Import with the same GUID. |
| Search Templates | required | Import with the same GUID. Contains reference for searching symbolic name of object store within the search XML. |
| Entry Templates | required | Import with the same GUID. Contains reference to the display name of the object store in a document property. Within the XML, there is only the symbolic name of the object store stored. |
| Events | Not required | Dependencies to subscriptions. |
| Code Modules | Not required | Dependencies to subscriptions. |
| Subscriptions | Not required | Can have dependencies, for instance, to Workflow Definition or for Class Definition. |
| Business documents | Not required | Import with the same GUID. |

The transformation handles the following issues:

- ▶ Map security principals
- ▶ Map object store references
- ▶ Map service references, such as URLs

The transformation does not handle the following example:

- ▶ Custom URLs, for instance, using the FileNet Workplace XT base URL is not covered by transformation.

Note: IBM is not responsible for testing and supporting your exported files and objects. Always test them in a non-production environment before you deploy them to a production environment.

9.7.3 Content Platform Engine import best practice

Importing the exported and transformed objects using the IBM FileNet Deployment Manager utility is straightforward by using the order that is presented in “Hierarchy of exports” on page 305.

The following common errors might occur at this stage:

- ▶ User and group information has not been updated to match the target environment, so users cannot access the objects as expected. The conversion process fails.
- ▶ Objects on which other objects depend do not yet exist in the target environment. Create the objects in the correct sequence.
- ▶ You have an object store metadata refreshing problem. Ensure that you refresh the IBM FileNet Deployment Manager after the import. Sometimes, the best practice is to close the IBM FileNet Deployment Manager utility and open it again.
- ▶ You have circular dependencies. For more information, see “Deploying P8 applications” in the IBM FileNet P8 Information Center.
- ▶ The import options have not been set correctly.
- ▶ An import process was interrupted by a transaction failure. Check the “Always update” option.

Tip: Remember that the IBM FileNet Deployment Manager utility uses multiple transactions during the import process and performs prefetching. So, if the import process fails for a particular package, some transactions are complete and others are not.

- ▶ The reuse of GUID has not been checked.

Recommendations: Do not use a generic administrative account for importing objects. Do not set the special security that is needed by IBM FileNet Deployment Manager for the entire object store administrators group. Always use a dedicated administrative user who has special security enabled for the object store.

9.7.4 IBM FileNet Deployment Manager

This section provides general information regarding IBM FileNet Deployment Manager.

There are existing documents that you can reference when you use FileNet Deployment Manager:

- ▶ Proven Practice: IBM FileNet Deployment Manager 5.1 Data Migrations
<http://www.ibm.com/support/docview.wss?uid=swg21609929>
- ▶ Migrating IBM Case Manager solutions using FileNet Deployment Manager and Case Manager Administration Client. (This document is valid also for FileNet Content Manager only.)
<http://www.ibm.com/support/docview.wss?uid=swg21612959>
- ▶ Impact of a FileNet Deployment Manager import using the “Update If Newer” option without the “Use Original Create/Update Timestamps” option
<http://www.ibm.com/support/docview.wss?uid=swg21455363>
- ▶ FileNet Deployment Manager fails to import the documents of a certain version series if one of those documents references an object that appears after the document in the deploy data set (explanation of object hierarchy)
<http://www.ibm.com/support/docview.wss?uid=swg27020038>
- ▶ MustGather: FileNet Deployment Manager (FDM)
<http://www.ibm.com/support/docview.wss?uid=swg21502186>

Hints

When using the FileNet Deployment Manager, remember this list:

- ▶ IBM FileNet Deployment Manager is able to transfer objects between *different* FileNet Content Manager domains within a single compressed file containing all the exported content.
- ▶ Object store transplantation between different FileNet Content Manager domains can occur by using the object store reassign function.
- ▶ Change the total transaction time to be longer for your connection pool to keep long running import processes active.
- ▶ Always inspect the FileNet Content Manager log and IBM FileNet Deployment Manager log after you import. Create pattern matching search strings to extract the data that you need.

Default export and import utilities

There are several default export and import utilities that you can use:

- ▶ IBM FileNet Deployment Manager utility and command-line interface
- ▶ IBM FileNet Enterprise Manager export/import manifest GUI and `ImpExpCmdTool.exe`
- ▶ `WFDeploymentTool.exe`

- ▶ WFAttachmentFinder.exe
- ▶ Process Configuration Console and peObject_export.bat and peObject_import.bat

9.7.5 Exporting and importing other components

In this section, we address exporting and importing the database, full text index, Directory Service Provider, Workflow System, and FileNet Workplace XT.

Database

All changes to rows in the object store database are covered by exporting and importing objects as previously explained.

In addition, you can consider propagating changes that have been applied at the database level, such as adding additional indexes, changing server options, and others. You can typically accomplish this function by extracting the index-related information from the SQL-based scripts that were written to configure the database in the source environment. Check whether the scripts depend on infrastructural information, such as user ID, password, server name, IP addresses, and database name.

Full-text index

Content-based retrieval (CBR) is covered in 4.12.2, “Content-based search” on page 141 and Chapter 11, “Upgrade and migration” on page 371.

Directory Service Provider

You can move parts of a directory either by exporting, transforming, and reimporting the parts by using tools, such as ldiff, or by writing scripts that create users and groups and add the users as members of the groups. The other method is to use IBM Tivoli Directory Integrator.

In any case, you must map the users, groups, and memberships to the target environment, which depends on the security settings in your company. If possible, use the same scripts and transform them based on a naming convention. This step needs to happen prior to the Content Platform Engine import.

Workflow System

Workflow System export and import are straightforward by using the Process Configuration Console or the command-line interface, which supports both full and incremental deployments.

The underlying workflow system APIs contain all the required methods to move Queues, Rosters, and EventLogs, and to validate Workflow Definitions.

You can export the Workflow System configuration by a call to the Workflow System Java API. The import into the Content Platform Engine works in a similar way.

If you currently use other Workflow System features or services, see “Deploying P8 applications” in the IBM FileNet P8 Information Center.

FileNet Workplace XT

Whenever FileNet Workplace XT applications have to be moved between environments, there are business assets and application configuration assets to be deployed.

FileNet Workplace XT stores various objects in an object store:

- ▶ Site Preferences
- ▶ User Preferences
- ▶ Entry Templates
- ▶ Stored Searches
- ▶ Search Templates
- ▶ Application Roles

We have already discussed the business assets under the export, transform, and import process. We do not need to provide further explanations from a methodology point of view.

Table 9-3 provides a short summary about these assets.

Table 9-3 Summary

| Asset type | Automatic transformation? | Remarks |
|------------------|---------------------------|---|
| Site Preference | No | Can be put in place by checking out the Site Preferences manually and checking in the new transformed version. |
| User Preferences | No | Try to avoid deployment and instead customize the general MyWorkplace style for all users; this will save you significant effort. |
| Entry Templates | Yes | Deal with security settings, folders, document classes, and default values. |

| Asset type | Automatic transformation? | Remarks |
|-------------------|---------------------------|--|
| Search Templates | Yes | Dependent on object name and GUID. Remember there are two content elements per object. |
| Application Roles | No export/import possible | Circular references cannot be maintained. |

FileNet Workplace XT is a web application that spans one war file, which contains the relevant Java APIs to connect to FileNet Content Manager.

9.8 Conclusion

We learned that deployment is divided into two major parts. The first one is the organizational part where you ensure that deployment is comprehensive by using a compliant process that is derived from the software development process. You as owner of the deployer role will be involved in each phase of development but the majority of work is performing the change and configuration management. Deployment without having a strategy in place causes inconsistency sooner or later. The second part of deployment is the procedural part of deployment.

This chapter also contained information about useful techniques and tools for deployment. We included samples of different ways of deploying based on the different kinds of data migrated between the environments. The first tool of choice for FileNet Content Manager data is the IBM FileNet Deployment Manager utility. This tool requires process-oriented procedures to ensure a quality-driven deployment in your company.

Chapter 10, “System administration and maintenance” on page 315, provides information about how to ensure the availability of your FileNet Content Manager solution from an administrative point of view.



System administration and maintenance

In this chapter, we describe some of the tools and methods that are used to monitor and maintain your IBM FileNet Content Manager system to ensure optimal performance.

We discuss the following topics:

- ▶ IBM FileNet Content Manager administrative roles
- ▶ Online help and existing documentation
- ▶ Monitoring the environment
- ▶ Capacity monitoring and growth prediction
- ▶ Tracing
- ▶ Auditing
- ▶ Managing the logs
- ▶ System administration tools
- ▶ Reducing storage costs
- ▶ Using virus scan software
- ▶ Applying fixes
- ▶ Updating security
- ▶ Backup and restore
- ▶ Task schedule

10.1 IBM FileNet Content Manager administrative roles

The task of managing an IBM FileNet Content Manager environment can be handled by a single person or the tasks can be divided up among several people. The size of the deployment and the business rules at a company, including separation of duty requirements, typically dictate who administers each part of the environment. To accommodate this flexibility in task ownership, the P8 documentation labels the different administrative roles, but it is up to each client to make these decisions:

- ▶ More than one person can be assigned to each role.
- ▶ The same person can be assigned to multiple roles.

Since the P8 security uses a directory server, a best practice is to assign Lightweight Directory Access Protocol (LDAP) groups to security roles when possible. Following this best practice avoids issues when individuals change roles, join the corporation, or leave the corporation.

The following administrative roles are key for an IBM FileNet Content Manager environment:

- ▶ Application server administrators
Have access to the application server console, are responsible for deploying Java EE applications, are allowed to stop and start Java virtual machines (JVMs), and can change application server tuning parameters.
- ▶ Database administrators
Create, delete, back up, and reorganize the databases that are required by the IBM FileNet Content Manager components. Can also tune databases, including adding complex indexes and reviewing query plans.
- ▶ IT administrators:
 - Operating system administrators
Install operating system updates, change access rights, and mount volumes and file storage areas.
 - Network administrators
Design and implement system architecture to make environments highly available, perform network traces as part of performance tuning, open ports on firewalls, and update access control lists (ACLs) on switches.

- ▶ Security administrators
Ensure that the directory server meets the requirements for an IBM FileNet Content Manager installation, provide the LDAP configuration information required for an IBM FileNet Content Manager installation, and tune performance.
- ▶ Global configuration database (GCD) administrators
Members of specific LDAP groups, who are identified during the Content Platform Engine installation, and who have rights to create object stores and other P8 domain-level artifacts.
- ▶ Object store administrators
Members of specific LDAP groups, who are identified during object store creation, and who have rights to administer the object store.

Because so many people can be involved in managing a P8 environment, use the following best practices:

- ▶ Document the following information:
 - Users and LDAP groups assigned to each role
 - Local processes that must be followed when making changes
- ▶ Keep the documentation current. Do not just create it for installation purposes and then forget about it.
- ▶ Always have at least two people in your organization who can fill a role.
- ▶ Use email distribution lists to ensure that everyone is kept informed when changes are being made to an environment.

10.2 Online help and existing documentation

A rich set of documentation is available for IBM FileNet Content Manager in an IBM hosted information center.

A new, IBM hosted information center is created for each IBM FileNet Content Manager release. Typically, only one part of the URL changes for each release, which is the release number:

- ▶ URL links to Version 5.1 of the IBM FileNet P8 Information Center:
<http://publib.boulder.ibm.com/infocenter/p8docs/v5r1m0/index.jsp>
- ▶ URL links to Version 5.2 of the IBM FileNet P8 Information Center:
<http://publib.boulder.ibm.com/infocenter/p8docs/v5r2m0/index.jsp>

During the installation of any IBM FileNet Content Manager component, you are prompted for the location of the information center. If suitable access is available from your site, use the URL of the IBM hosted information center for these reasons:

- ▶ The content is updated regularly.
- ▶ There is no maintenance overhead.

An installable version of the information center is also provided with the product software. If access to the Internet is limited at your site, install the shipped version of the information center on a local application server and give the URL for this local installation when configuring the IBM FileNet Content Manager components.

When using any of the applications or administrative tools supplied with IBM FileNet Content Manager, clicking Help displays the appropriate topic from the information center. You can also navigate to the same information by accessing the information center directly.

The following figures show the main menu for the IBM Content Navigator help. Figure 10-1 on page 319 shows the display after clicking Help from within the IBM Content Navigator application. Figure 10-2 on page 319 displays the same information but it was accessed by navigating to the help from the information center posted on the IBM website.

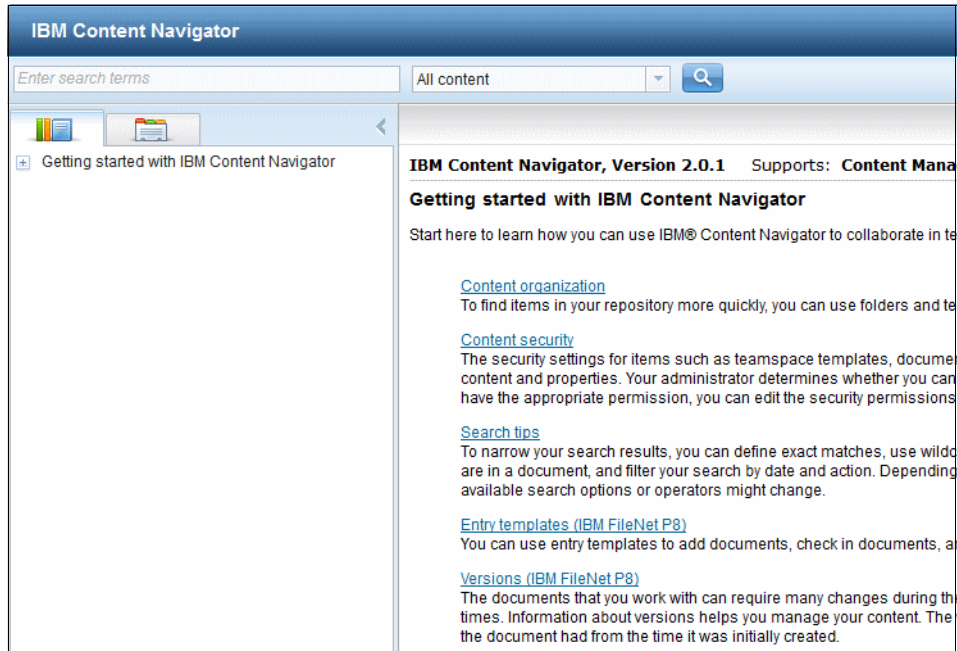


Figure 10-1 IBM Content Navigator help when accessed from within the application

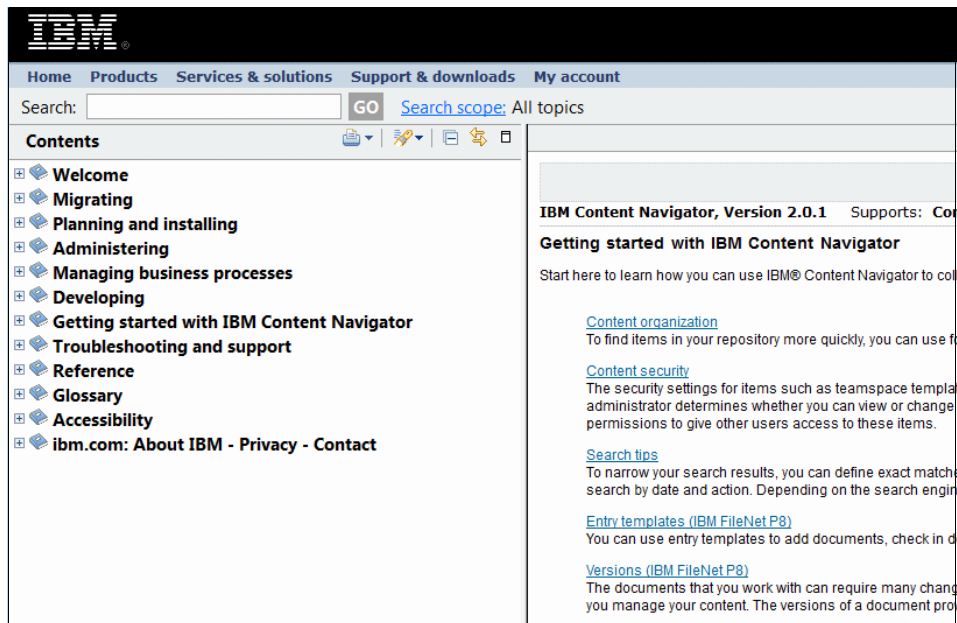
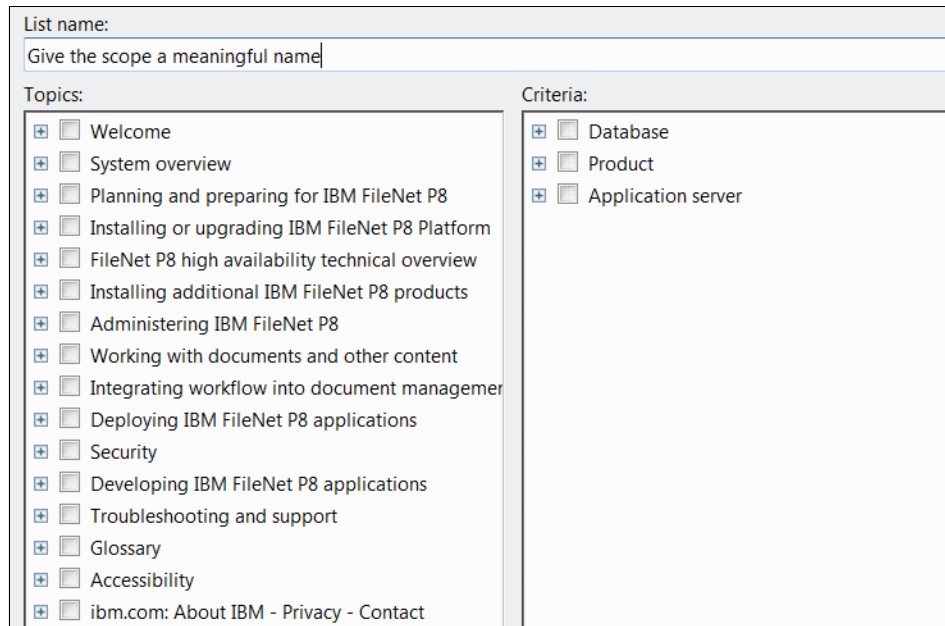


Figure 10-2 IBM Content Navigator help when accessed from the Information Center

10.2.1 Tips for working with the information center

When looking for information, use Scope on the menu bar to limit the focus when searching for information or when navigating through topics.

The Scope options limit searches to specific branches of the information center, and can also restrict the information displayed based on database, product, or application server, as shown in Figure 10-3.



The screenshot shows a web interface for the IBM FileNet P8 information center. At the top, there is a text input field labeled "List name:" with the placeholder text "Give the scope a meaningful name". Below this, the interface is divided into two main sections: "Topics:" and "Criteria:". The "Topics:" section contains a list of 18 topics, each with a plus sign icon and a checkbox. The "Criteria:" section contains three options: "Database", "Product", and "Application server", each with a plus sign icon and a checkbox.

| Topics: | Criteria: |
|--|---|
| <input type="checkbox"/> Welcome | <input type="checkbox"/> Database |
| <input type="checkbox"/> System overview | <input type="checkbox"/> Product |
| <input type="checkbox"/> Planning and preparing for IBM FileNet P8 | <input type="checkbox"/> Application server |
| <input type="checkbox"/> Installing or upgrading IBM FileNet P8 Platform | |
| <input type="checkbox"/> FileNet P8 high availability technical overview | |
| <input type="checkbox"/> Installing additional IBM FileNet P8 products | |
| <input type="checkbox"/> Administering IBM FileNet P8 | |
| <input type="checkbox"/> Working with documents and other content | |
| <input type="checkbox"/> Integrating workflow into document management | |
| <input type="checkbox"/> Deploying IBM FileNet P8 applications | |
| <input type="checkbox"/> Security | |
| <input type="checkbox"/> Developing IBM FileNet P8 applications | |
| <input type="checkbox"/> Troubleshooting and support | |
| <input type="checkbox"/> Glossary | |
| <input type="checkbox"/> Accessibility | |
| <input type="checkbox"/> ibm.com: About IBM - Privacy - Contact | |

Figure 10-3 Scope selections

Bookmark useful information by using the following procedure because clicking a topic does not change the URL in the browser address bar:

1. Right-click the topic link in the left pane.
2. Select **Open the topic in a new tab or window** from the context menu.
3. Go to the new tab or window to create the bookmark.
4. Use the Print option to create a printer-friendly version of a topic and all its subtopics.

Using a combination of the scope and print options is a great way to create an installation or maintenance guide tailored to your environment.

10.2.2 Other useful documentation

Consider creating bookmarks for the following two documents because you will reference them frequently:

- ▶ P8 Hardware and Software Guide

This guide provides detailed information about the supported underlying components that are required by the P8 suite of products, including the databases, operating systems, application servers, file storage systems, and so on. The guide is updated regularly and a new version is released with each IBM FileNet Content Manager release.

You can access the guide either from the home page of the Information Center by using the “IBM FileNet P8 supported hardware and software” link under “Get started”, or from the following URL:

<http://www.ibm.com/support/docview.wss?uid=swg27013654>

This page provides links to the versions of the guide, which is especially helpful when planning an upgrade. It is easy to compare and contrast the infrastructure requirements for the software you currently use and the infrastructure requirements of the release to which you plan to upgrade.

- ▶ FileNet P8 Fix Pack Compatibility Matrices

Microsoft Excel spreadsheets identify all the generally available fix packs and interim fixes for each of the FileNet P8 components. There is one matrix for each IBM FileNet Content Manager release. The matrices identify which components are compatible and the software build number for each fix. This information is needed in these situations:

- Planning an upgrade to ensure that you are at a supported minimum level
- Working with IBM Software Support on an issue

The matrices can be downloaded from the following URL:

<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27014734>

10.3 Monitoring the environment

In this section, we discuss monitoring the IBM FileNet Content Manager environment for these areas:

- ▶ Performance
- ▶ Functional issues
- ▶ Capacity planning
- ▶ Security

Part of monitoring your environment is setting expectations. For your environment, decide what thresholds need to be set for the following items and what action needs to be taken when the thresholds are reached:

- ▶ CPU usage
- ▶ Memory usage
- ▶ Disk usage
- ▶ Response times
- ▶ Number of objects in the object store

There also might be legal requirements that need to be monitored; for example, knowing who accessed content, or when content was deleted.

Three useful web pages are available for getting a quick check on the status of the Content Platform Engine:

- ▶ The Ping Page (Content Engine Startup Context):

`http://<Content Platform Engine server>:<port>/FileNet/Engine`

Use this URL to check that your Content Platform Engine environment is running and to gather other useful information about the environment, such as the build number. Figure 10-4 on page 323 shows the information that is provided by the ping page.

Tip: If Content Platform Engine is deployed to a cluster, replacing *<Content Platform Engine server>* with the name of the management node will show whether at least one Content Platform Engine node in the cluster is running. To ensure that a specific Content Platform Engine server is running, use the name of the server in the ping page URL.

| Content Engine Startup Context (Ping Page) | |
|--|--|
| Key | Value |
| Product Name | P8 Content Platform Engine - 5.2.0 |
| Build Version | dap511.470 |
| Operating System | Windows Server 2008 R2 6.1 |
| JVM | java.vm.vendor IBM Corporation java.vm.name IBM J9 VM java.runtime.version pwa6460_26sr2ifx-20120419_02 (SR2) java.runtime.name Java(TM) SE Runtime Environment java.vm.version 2.6 java.vm.info JRE 1.6.0 Windows Server 2008 R2 amd64-64 Compressed References 20120322_106210 (JIT - r11_20120322_22976 GC - R26_Java626_SR2_20120322_1722_B106210 JIT - r11_20120322_22976 GC - R26_Java626_SR2_20120322_1722_B106210) java.fullversion JRE 1.6.0 IBM J9 2.6 Windows Server 2008 R2 amd64-64 Compressed References 20120322_106210 (JIT - r11_20120322_22976 GC - R26_Java626_SR2_20120322_1722_B106210 JIT - r11_20120322_22976 GC - R26_Java626_SR2_20120322_1722_B106210) |
| Start Time | Tue Mar 05 11:53:45 PST 2013 |
| Classpath | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\properties;C:\Program Files\IBM\WebSphere\AppServer\lib\bootstrap.jar;C:\Program Files\IBM\WebSphere\AppServer\lib\jsf-nls.jar;C:\Program Files\IBM\WebSphere\AppServer\lib\urlprotocols.jar;C:\Program Files\IBM\WebSphere\AppServer\deploytool\ftp\batch2.jar;C:\Program Files\IBM\WebSphere\AppServer\java\lib\tools.jar |
| Log File Location | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\FileNet\server1 |
| Local Host | cm-terranovavm21 |
| Available Processors | 1 |
| Working Directory | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01 |
| J2EEUtil | com.filenet.apimpl.util.J2EEUtilWS |
| Startup Message | P8 Content Platform Engine Startup: 5.2.0 dap511.470 Copyright IBM Corp. 2003, 2013 All rights reserved on server |

Figure 10-4 Content Platform Engine ping page

- ▶ The health page:

`http://<Content Platform Engine server>:<port>/P8CE/Health`

This page shows the health of various P8 domain artifacts, including object stores and storage areas. A sample health page is shown in Figure 10-5.

Click an item on the page to get more information.

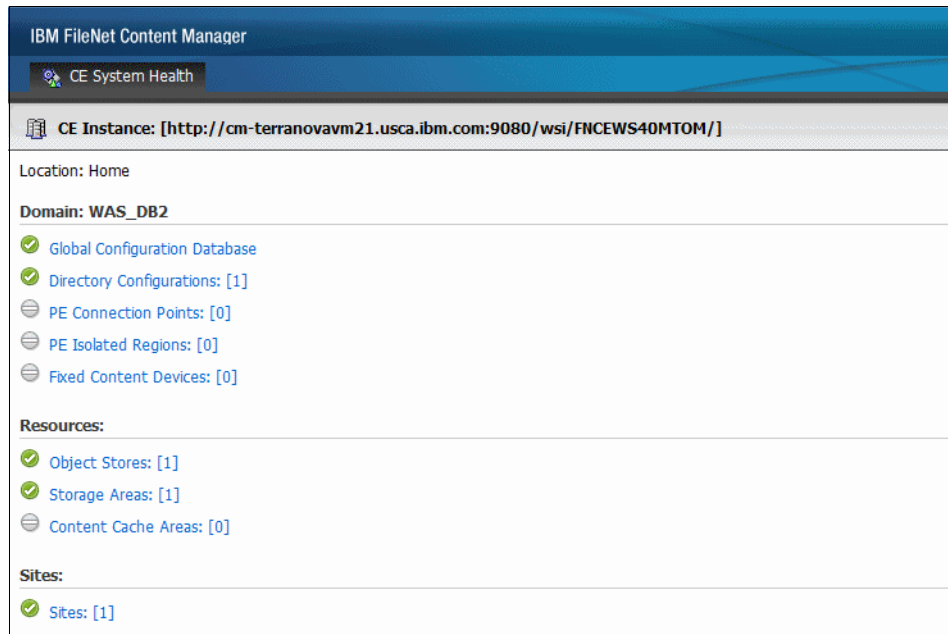


Figure 10-5 Content Platform Engine health page

- The workflow ping page:

<http://<Content Platform Engine server>:<port>/peengine/IOR/ping>

Unlike the other pages, a user name and password is required to access this page. Much of the information provided on the page is similar to what is on the Content Platform Engine ping page, but it also provides information about the workflow threads. Figure 10-6 illustrates the workflow system ping page.

| Process Engine Server Information (Ping Page) | |
|---|---|
| Key | Value |
| Product Name | P8 - 5.2.0.0 |
| Build Version | dap511.470 pe.jar:dap511.470, 02/26/2013 15:33:20 |
| Operating System | Windows Server 2008 R2 6.1 amd64 |
| JVM | JRE 1.6.0 Windows Server 2008 R2 amd64-64 Compressed References 20120322_106210 (JIT enabled, AOT enabled R26_Java626_SR2_20120322_1722_B106210_CMPRSS J9CL - 20120322_106210 |
| Start Time | 2013/03/05 11:54:42.829-0800 |
| ClassPath | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\properties;C:\Program Files\IBM\WebSphere\AppServer\Files\IBM\WebSphere\AppServer\lib\bootstrap.jar;C:\Program Files\IBM\WebSphere\AppServer\lib\jsf-nls.jar;C:\Program Files\IBM\WebSphere\AppServer\lib\urlprotocols.jar;C:\Program Files\IBM\WebSphere\AppServer\deploytool\itp\batchboot.jar;C:\Program Files\IBM\WebSphere\AppServer\java\lib\tools.jar |
| Log File Location | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\FileNet\server1 |
| Local Host | cm-terranovavm21 |
| Data Directory | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\FileNet\server1\pdata |
| Server Instance | cm-terranovavm21/server1 |
| Ping time | 2013/03/05 12:44:01.563-0800 |
| Database | PEPrimary POOL: PESecondary POOL: EventExporterTaskManager DELAYED until another 6969ms , Region=0 TaskManager DELAYED until another 110859ms , Region=0 PEHeartBeat POOL: EventExporter-DataCollector POOL: EventExporter-DataPublisher POOL: PESERVER:49733 @Tue Mar 05 12:10:51 PST 2013 PECIMsg @Tue Mar 05 12:10:51 PST 2013 |
| Active RPC Threads: | |

Figure 10-6 Workflow system ping page

Monitoring for performance, functional issues, and capacity planning can be accomplished using the following tools:

- ▶ IBM System Dashboard for ECM ships with IBM FileNet Content Manager and provides a centralized performance monitoring mechanism.
This tool is especially useful when performing stress and load testing.
- ▶ IBM ECM System Monitor is an optional add-on used to both monitor and manage the IBM FileNet Content Manager environment and the servers on which the software is installed. The tool can be extended with a customized knowledge base of corrective actions. You can use this tool to automate routine system administration and obtain historical analysis and reporting.

10.4 Capacity monitoring and growth prediction

When planning for an IBM FileNet Content Manager system, you need to estimate the average amount of content added per day, average size of the content, how many users have access to it, and other basic information about your planned application. Answers to these questions are run through a modeling tool, IBM Content Capacity Planner, by your IBM representative. The modeling tool provides details about the necessary servers, database space, and overall disk space for storage. The modeling tool also estimates the CPU utilization of the necessary servers. For more information, see Chapter 8, “Capacity planning with IBM Content Capacity Planner” on page 253.

As you deploy and begin using your application, monitor and record these server statistics:

- ▶ Disk usage
- ▶ CPU and memory utilization
- ▶ Database usage

Your database administrators can provide database details. The most important information is the overall database size, but it is also good to know whether specific tables or data fields are growing rapidly.

IBM FileNet Content Manager systems tend to grow over time. Object store content is added daily, additional applications are developed, and users are added. By monitoring and recording these statistics, you can measure how your system is performing against the initial model. More importantly, you can track how quickly you are using resources and determine the impact to the system when an increase in system usage is planned.

We advise monitoring for capacity weekly, including monitoring database, network, and disk usage. You want your capacity planning model to be as close

as possible to the needs of the live production system, so that the output from the Content Capacity Planning tool provides an accurate assessment of future needs. Proper capacity monitoring provides you with advanced notice that additional server resources need to be allocated to the system. The initial model is an estimate of what is needed by using the numbers that you provide. If your estimated content count or size was too small, you need to plan additional space.

10.4.1 IBM System Dashboard for ECM

IBM FileNet Content Manager ships with a centralized performance monitoring mechanism called *IBM System Dashboard for ECM* (also known as the *System Manager*). System Dashboard is composed of two parts: a *listener* that runs on each server collecting information and a *manager* that displays the information. The Dashboard is the supplied application for configuring, displaying, and saving the collected information. Use this tool when tuning the environment for optimal performance and to routinely monitor system performance.

IBM System Dashboard for Enterprise Content Management V5.1, SC19-3084-03, provides detailed instructions on configuring and using the listeners and manager components. To access the guide, use the following URL:

<http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss?CTY=US&FNC=SRX&PBL=SC19-3084-03>

When IBM FileNet Content Manager components are installed, a default System Dashboard listener is automatically installed and activated. The listener component collects details about the software version as well as performance data.

If needed, you can instrument your Java or Microsoft Windows 32-bit C++ applications to use the provided listeners and expose performance data that is visible in the Dashboard. This way, you can monitor your applications as well as the IBM FileNet Content Manager components. The Java and C++ APIs are documented in the information center as part of the P8 Developer Help section.

By default, each listener buffers approximately 24 hours worth of collected data details.

There are four configuration parameters: `port_number`, `secondary_port`, `output_count`, and `output_interval`. The parameters identify the ports that the listeners use, the interval over which information needs to be aggregated, and the amount of data that can be written to a summary log file before you create a file.

The listeners are activated automatically; however, there are several operating system-specific requirements. For more information, see the following page:

http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.sysmgr.admin.doc%2Foverview_activate.htm

10.4.2 Dashboard

The Dashboard generates detailed reports about performance. It displays the details and can also save the information in various formats.

The Dashboard is a Java utility that can be installed and run on Windows or UNIX/Linux clients. It is installed separately from the server installation. It can also be installed and run on the IBM FileNet Content Manager servers. On Windows machines, run the Dashboard utility. On UNIX, you must have an XWindows display exported and run the P8Manager shell script. The Dashboard installs a local copy of its online help that can be accessed from the Help menu option.

When the Dashboard is first run, you need to define clusters of IBM FileNet Content Manager components to monitor. (The *cluster* is a logical construct used by the Dashboard; it has no relation to an application or operating system cluster.) These clusters are not used for high availability but are simply a user-defined logical collection or cluster of servers to monitor. The cluster contains servers and monitoring frequency. Select the **Clusters** tab and click **New**. Enter a name for the cluster, which is typically the application system name or location, and click **OK**. See Figure 10-7.

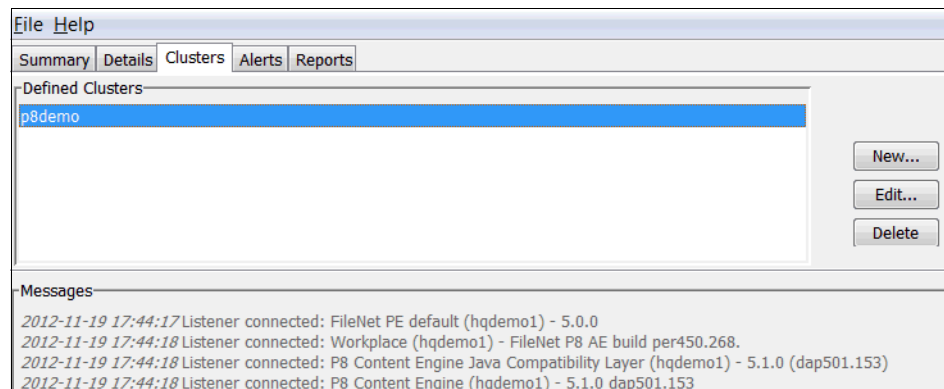


Figure 10-7 Dashboard: New cluster

Follow these steps to add a server and timing details:

1. Click **Edit**.
2. Enter the name of the host that is running the listener. Unless the port was reconfigured, the default port is 32775.
3. Enter the interval. The Interval sets the frequency that the Dashboard polls the server listeners to get details in seconds.

For a 15-minute interval, enter 900 seconds. The number of data points sets the maximum number of interval details that the Dashboard keeps in the display.

4. Click **OK**.

Figure 10-8 shows an added server and interval information.

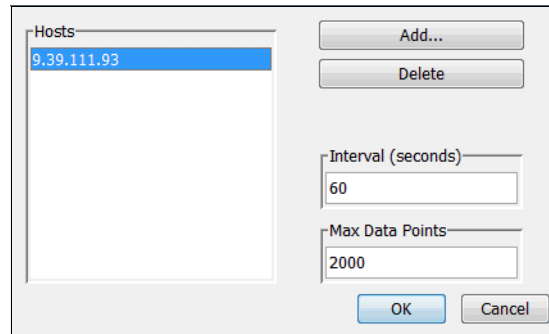


Figure 10-8 Cluster add server

The Dashboard tool queries the System Dashboard listeners on the servers and populates details in the Dashboard tool's various windows. It finds all listeners running on each server; individual servers need to be defined only once. You can save the cluster details for future use or open existing details from the file menu. The cluster file is an XML-formatted file that is saved on the local computer. You can copy the `cluster.xml` file to other computers where the Dashboard is installed for use on other workstations.

The Dashboard Summary tab shows a graph of the cluster's performance.

The Details tab contains counter details for all listeners. You can expand the IBM FileNet Content Manager applications on each server and view the following items:

- ▶ CPU, Network, and Disk utilization
- ▶ Environmental details, such as OS level, IBM FileNet Content Manager version, and Java virtual machine (JVM) settings
- ▶ Remote Procedure Call (RPC) activity shows how the IBM FileNet Content Manager subsystems are performing. It details the count and average time consumed (duration) by the calls during the interval

Figure 10-9 shows RPC count details and the number of items processed per interval.

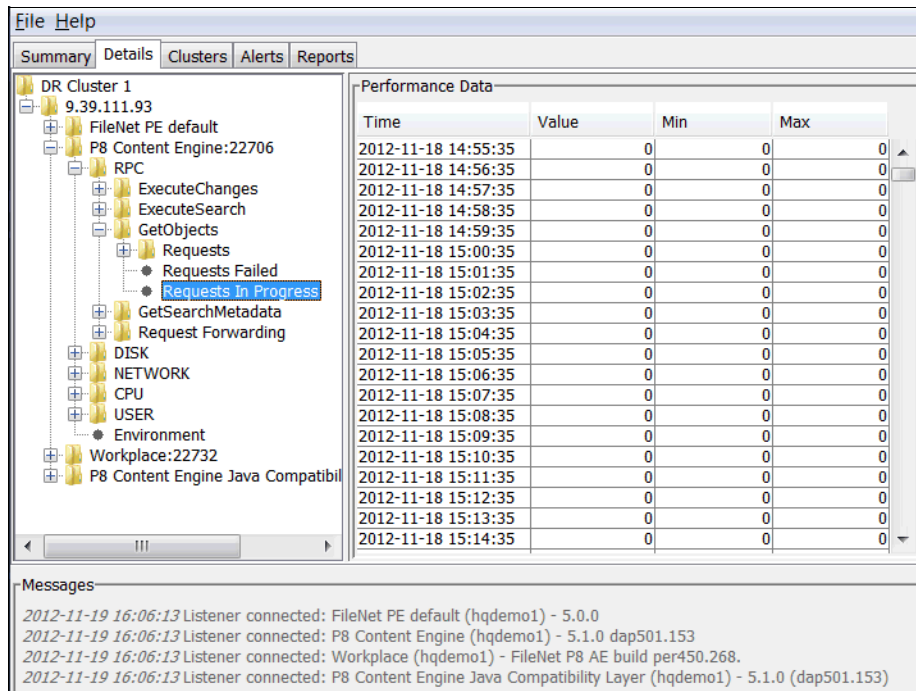


Figure 10-9 RPC count details window

The Dashboard is a Java application that holds the details in memory. If it crashes or seems unresponsive, try these actions:

- ▶ Check the network speed to ensure that it can transfer the generated data fast enough. The amount of the data being transferred depends on the number of servers being monitored and the transaction rate in the IBM FileNet Content Manager environment.
- ▶ Ensure that you have the latest Dashboard and check its documentation for any additional memory configuration details. The Dashboard is independent of the listener version; the latest Dashboard functions with older listeners.
- ▶ Increase the Java memory for the Dashboard. The Dashboard can run on machines with less than 2 GB memory. If you use a machine with less than 2 GB of memory and monitor many listeners, increase the machine's memory to 2 GB or take the following actions:
 - Reduce the number of listeners being monitored.
 - Increase the collection interval.
 - Reduce the number of data points specified.
 - Add memory.

The Dashboard has a report mechanism that allows you to save reports in comma-separated value (CSV) format, which is useful for generating spreadsheet reports. There is also an export option that enables you to generate data that can be used as input to IBM Content Capacity Planner. In addition, you can save the report template for future use. For more information, see the Dashboard's online help for reports.

Note: The Dashboard uses the name Scout to refer to the IBM Content Capacity Planner.

Figure 10-10 on page 332 shows a sample report output.

| | A | B | C | D | E | F | G | H | I |
|----|--------------------|---|---|---|---|---|--|--|--|
| 1 | os.arch | x86 | | | | | | | |
| 2 | user.language | en | | | | | | | |
| 3 | sun.os.patch.level | Service Pack 1 | | | | | | | |
| 4 | jvm_max_memor | 2.68E+08 | | | | | | | |
| 5 | app_name | CEMP_Daphne Server1 | | | | | | | |
| 6 | app_version | V1.0 | | | | | | | |
| 7 | java.runtime.ver | 1.4.2 | | | | | | | |
| 8 | os.name | Windows 2003 | | | | | | | |
| 9 | os.version | 5.2 | | | | | | | |
| 10 | processors | 1 | | | | | | | |
| 11 | hostname | hqdemo1 | | | | | | | |
| 12 | | aphne Server1:1 866\ RPC/Exe cuteChan | aphne Server1:1 866\ RPC/Exe cuteChan | aphne Server1:1 866\ RPC/Exe cuteChan | aphne Server1:1 866\ RPC/Exe cuteChan | aphne Server1:1 866\ RPC/Exe cuteChan | aphne Server1:1 866\ RPC/Exe cuteSearc | aphne Server1:1 866\ RPC/Exe cuteSearc | hne Server1:186 6\ RPC/Execut eSearch/Re |
| 13 | Time | | | | | | | | |
| 14 | 9/18/2007 16:53 | | | | | | 0 | | |
| 15 | 9/18/2007 17:08 | | 0 | 0 | 0 | 0 | | 0 | 2015000000 |
| 16 | 9/18/2007 17:10 | | 0 | 0 | 0 | 0 | | 0 | 0 |
| 17 | 9/18/2007 17:10 | | 0 | 0 | 0 | 0 | | 0 | 0 |
| 18 | 9/18/2007 17:10 | | 0 | 0 | 0 | 0 | | 0 | 0 |
| 19 | 9/18/2007 17:10 | | 0 | 0 | 0 | 0 | | 0 | 0 |
| 20 | 9/18/2007 17:11 | | 0 | 0 | 0 | 0 | | 0 | 0 |
| 21 | 9/18/2007 17:11 | | 0 | 0 | 0 | 0 | | 1 | 625000000 |
| 22 | 9/18/2007 17:11 | | 0 | 0 | 0 | 0 | | 4 | 457000000 |

Figure 10-10 Sample CSV Dashboard report

System Dashboard performance archiver

System Dashboard provides a Java `archiver.jar` application that can be used to collect data automatically. The JAR file can be run on any server or workstation with Java and connectivity to the IBM FileNet Content Manager listeners.

Running the `archiver.jar` application can be automated through host scripts. The `archiver.jar` application writes to files with one file per listener in a log directory. The archived files are binary files that can be opened via the Dashboard's File → Open Archive menu. The same view and report options apply that are available in a live system monitoring session.

Table 10-1 on page 333 lists the `archiver.jar` parameter options.

Table 10-1 archiver.jar parameter options

| Option | Description |
|--------------|--|
| -t hh:mm | Total amount of time in hours and minutes that the archiver process must run |
| -n hh:mm | The interval at which the current archived files must be closed and new ones opened |
| -i integer | The interval, which is specified in seconds, at which to poll for data from the specified machines |
| -d file path | The path to the location at which to place the archive log files |
| FileName.xml | The complete path to the saved cluster file that specifies which machines to poll |

This is an example of a command to run the archiver.jar application:

```
java -jar archiver.jar -h -d Logs -t 12:00 -n 04:00 -i 15 cluster.xml
```

In this example, the archiver collects performance details in 15-second intervals, creates new archive logs every four hours, and automatically stops after 12 hours. If you stop the archiver process early, part of the buffered performance data might not appear in the last archive file. If the archiver loses connectivity with a listener, by default, it attempts to reconnect five times at intervals that are 5 seconds apart before it stops attempting to connect to the failed listener. The -h option specifies that the available listener's history must be included in the generated archive file.

Recommendations: Start the archiver.jar application immediately before your system activity picks up during the peak times (for example, in the morning) and run it until activity slows down (for example, in the evening).

If you restart your system while the archiver.jar application is running, you must restart the archiver.

System Dashboard client API

System Dashboard includes a Java API set for clients, who want to add Dashboard monitoring into their applications. The following technote explains how to download the API and associated documentation:

<http://www.ibm.com/support/docview.wss?uid=swg21502836>

Usage Reporter

The Usage Reporter is provided with the System Dashboard and is used to monitor the number of users accessing the Content Platform Engine. The tool looks for individual user names. If access to the engine is via an application that uses a service or guest account, the tool might not reflect the number of people actually using the system.

For more details, see the following document:

<http://publibfp.dhe.ibm.com/epubs/pdf/c1930850.pdf>

10.4.3 IBM ECM System Monitor

IBM ECM System Monitor is an optional component. System Monitor provides automated, proactive system monitoring that can notify your support personnel directly or through system management consoles, such as IBM Tivoli Enterprise Console®. Use System Monitor to monitor all aspects of your IBM FileNet Content Manager servers. It provides early fault detection and prevention to aid support personnel in reducing system downtime. It monitors performance, disk utilization, event logs, and literally hundreds of IBM FileNet Content Manager application and system parameters. System Monitor contains a default set of monitors for IBM FileNet Content Manager components and allows you to create your own monitors for application-specific monitoring.

System Monitor features a web interface that authorized personnel use to monitor and manage your system. It features a knowledge base of faults and possible corrective actions. You can customize this knowledge base to offer application-specific corrective actions. When a fault is encountered, support personnel can quickly identify and correct the failing component. Figure 10-11 on page 335 shows a sample report generated with System Monitor. The right side of the report displays a graph showing CPU utilization. The left side of the report shows all the events in which the CPU thresholds were breached.

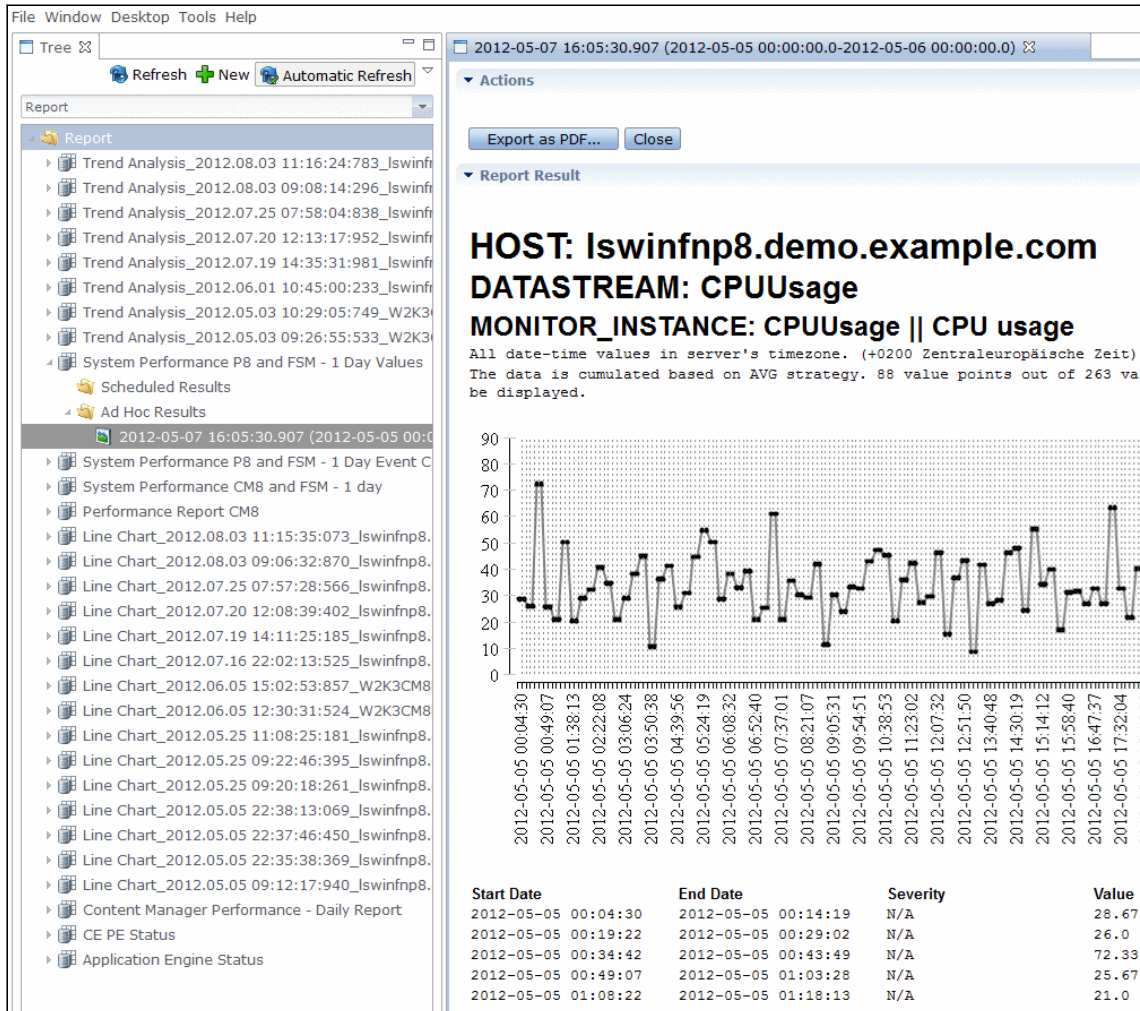


Figure 10-11 ECM System Monitor sample report

The rapid fault isolation and corrective action database make System Monitor a must-have for mission critical systems. System Monitor reduces manual efforts in the daily administration of IBM FileNet Content Manager and helps to increase system availability. System Monitor can help reduce your operational costs and help you meet your service-level agreements (SLAs) more efficiently.

For more information about IBM ECM System Monitor, go to the following links:

- ▶ <http://www.ibm.com/software/products/us/en/ecmsystemmonitor>
- ▶ <http://www.ibm.com/support/docview.wss?uid=swg27010374>

10.5 Tracing

Tracing is primarily used for debugging. Tracing can be enabled for many components. By default, tracing is disabled. Tracing can be enabled and disabled without recycling the application server.

Note: Tracing all components can create enormous trace log files with little system activity. Performance might also be affected. Enable the minimum necessary tracing to collect the required information in relation to the problem that you are investigating.

Tracing is controlled via the IBM Administration Console for Content Platform Engine (ACCE). Tracing can be enabled at the P8 domain level and at the site level. Any setting at the site level takes precedence over the setting at the object store level, including the location of the trace logs. If you use different settings at the P8 domain and site levels, ensure that you track the various settings. Figure 10-12 shows the trace control settings at the P8 domain level.

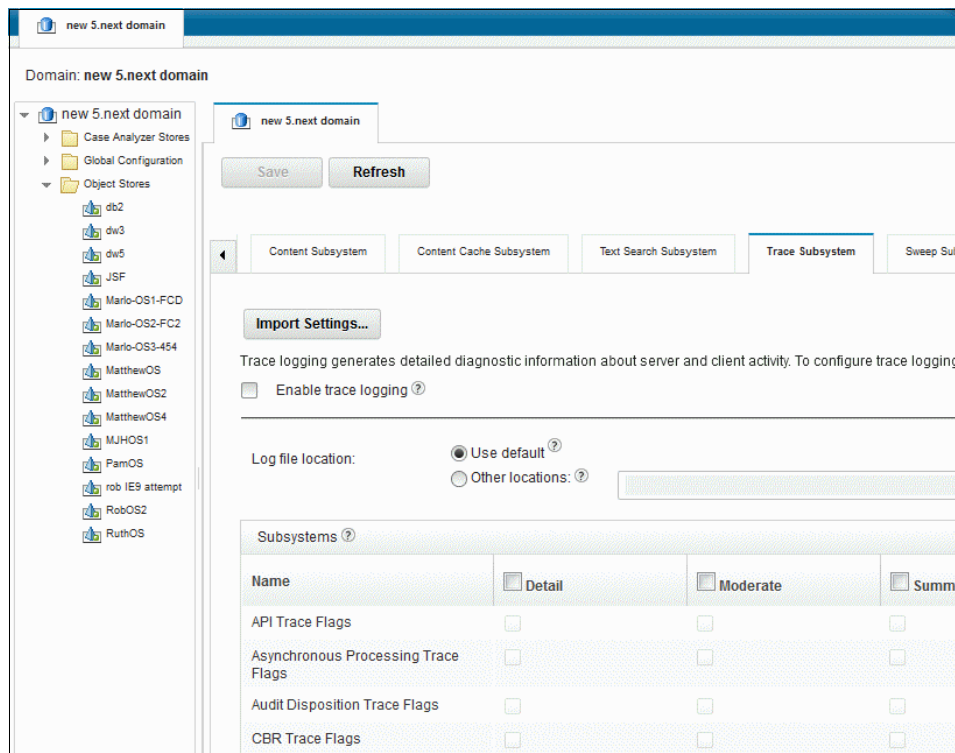


Figure 10-12 Setting tracing in ACCE

Process-related tracing is controlled by the command-line utility, **vwtool**. For more information about **vwtool**, see the following page in the information center:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.pe.vw.doc%2Fbpfv1049.htm>

10.6 Auditing

Auditing is the recording of events that occur on objects. For each recorded event, a row is added to the event table in the object store's database. From the event object, you can get information about the audited event, including the creation date, originating user, result status, and source object of the event.

All out-of-the-box events, such as create and check-in, can be audited, and the auditing capability can be extended to custom events. The IBM Enterprise Records (IER) product, for example, takes advantage of this extensibility to provide audit events that are specific to a records management environment. However, auditing can affect both performance and the database space usage, so it is important to configure auditing judiciously.

For more information about auditing concepts, see the following topic in the information center:

http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.dev.ce.doc%2Faudit_concepts.htm

In addition to storing information about the type of change made to the object and who made the change, you can also choose to store copies of the object before and after the audited event. The `ObjectStateRecordingLevels` property defines whether to keep copies of the object in the audit record from before and after the audited event. The `ObjectStateRecordingLevels` property takes the following values:

- ▶ `ORIGINAL_AND_MODIFIED_OBJECTS`
Records a copy of both the original, pre-event object and the modified, post-event object
- ▶ `MODIFIED_OBJECT`
Records a copy of the modified, post-event object
- ▶ `NONE`
Does not store a copy of the object being audited

Recommendations: Set `ObjectStateRecordingLevels` property to `NONE` because persisting audited source objects in a database can result in substantial consumption of large object (LOB) storage. If you have a regulatory requirement for keeping copies of the object in the audit entry, set it to the appropriate auditing level.

Another way to limit the information stored by an audited event is to use the `AuditAs` meta-property on `PropertyTemplates`. This meta-property enables you to audit specific properties on specific events directly without having to record the object's entire state. Limiting the properties that are audited reduces the required space for the audit information and makes it easier to generate meaningful audit reports.

The following link provides a procedure for configuring property auditing:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.admin.tasks.doc%2Fp8pcc012.htm>

10.7 Managing the logs

In this section, we discuss system message logs, tracing, and log maintenance. Chapter 12, “Troubleshooting” on page 387 provides information about how to interpret the information in the logs.

10.7.1 Log location

IBM FileNet Content Manager is written in Java. Java applications do not log error messages; they log exceptions. Java normal messages and exceptions are written to message log files.

If you are running IBM FileNet Content Manager in a multi-JVM environment, a separate set of logs exists for each JVM.

Content Platform Engine has two message logs for content-related information:

- ▶ `p8_server_error.log`
- ▶ `p8_server_trace.log`

The process-related content is written to the following logs:

- ▶ `pesvr_system.log`
- ▶ `pesvr_trace.log`

Use the Ping page to find the log file location. On the Ping page, search for “Log File Directory”:

`http://<Content Platform Engine server>:<port>/FileNet/Engine`

These files are at the following default locations:

- ▶ WebSphere Application Server
`install_root/profiles/profile_name/FileNet/server_Instance_name`
- ▶ WebLogic Server
`bea/user_projects/domains/my_domain/FileNet/AdminServer`
- ▶ JBoss
`jboss_install/jboss-as/bin/FileNet/server_instance_name`

In addition, you might need to review the application server message logs. These files are at the following default locations:

- ▶ For IBM WebSphere:
 - `WAS_install_path/AppServer/profiles/profile_name/logs/server_name/SystemOut.log`
 - `WAS_install_path/AppServer/profiles/profile_name/logs/server_name/SystemErr.log`
- ▶ For Oracle WebLogic
`WLS_install_path/user_projects/domains/domain_name/servers/server_name/logs/server_name.log`
- ▶ For JBoss
`JBOSS_DIST/server/server_name/log/server_name.log`

10.7.2 Log file size

Typically, you only refer to the logs when you troubleshoot errors, so keep the logs to a reasonable size to make it easier to find the information of interest.

If an issue can be reproduced, either roll to a new set of logs or prune the logs before you reproduce the issue.

Recommendations: Rename system logs to a date format name to keep them for a brief period, and then delete them. The log maintenance timing depends on how busy your system is and how large the logs grow over time.

10.7.3 Trace logs

Tracing all components can create enormous trace log files with little system activity. Performance might be affected. Enable the minimum trace logs to collect the required information in relation to the problem that you are investigating.

10.7.4 Audit logs

Content Platform Engine provides object store audit logging capabilities. When auditing is enabled, the audit log entries are stored in the object store database.

10.8 System administration tools

In this section, we discuss the tools you can use to monitor and manage an IBM FileNet Content Manager environment.

10.8.1 Configuration Manager

Content Platform Engine runs as a Java EE application within an application server. Use the Configuration Manager that is installed with the Content Platform Engine software to configure the software. The tool has a user interface, or you can complete the tasks using a series of command-line scripts.

You can complete the following tasks by using the Configuration Manager:

- ▶ Identifying the directory server to be used with the IBM FileNet Content Manager installation
- ▶ Creating data sources for the GCD, object stores, and workflow systems
- ▶ Identifying the bootstrap user
- ▶ Building and deploying the Content Platform Engine WAR file

The information that is supplied via the Configuration Manager is saved in a profile file. You can use this profile file later to add more resources, such as Java Database Connectivity (JDBC) data sources, to the environment, as well as when you upgrade the Content Platform Engine software to a new release.

Tip: You can perform all the required configuration tasks, such as configuring the directory server and creating JDBC data sources, manually by using the administrative tools provided by the Java EE application server. However, completing the tasks this way is error prone and likely to result in an installation that is configured incorrectly. Instead, use the Configuration Manager.

10.8.2 IBM Administration Console for Content Platform Engine

ACCE is a web-based administrative tool for managing a P8 domain, including the GCD, object stores, workflow systems, and sites. This tool is a replacement for the Windows Microsoft Management Console snap-in tool, IBM FileNet Enterprise Manager (FEM).

Both tools are provided with IBM FileNet Content Manager. ACCE is deployed as part of the Content Platform Engine WAR file. FEM is one of the tools that can be installed when the Content Platform Engine server is installed on a Microsoft Windows platform.

If a task cannot be completed in ACCE, use FEM instead.

A single installation of FEM can be used to manage multiple P8 domains. An ACCE deployment is for a specific P8 installation because it is deployed as part of the Content Platform Engine WAR file.

See the P8 Information Center for detailed information about completing tasks using ACCE and FEM. In this Redbooks publication, the focus is primarily on using ACCE to complete Content Platform Engine administrative tasks.

System administrators need access to ACCE. It is used to configure object stores and workflow systems, define content properties, assign content security, and administer your IBM FileNet Content Manager system.

In development and test environments, it can be useful to expand the usage of ACCE to additional resources so that they can easily build required artifacts and validate behavior seen elsewhere in the IBM FileNet Content Manager environment. However, it is important to establish protocols for using these tools and for building artifacts.

Tips when using ACCE

There are four main artifacts that are managed by ACCE:

- ▶ Domain
- ▶ Case analyzer stores
Case analyzer stores are used with the IBM Case Foundation and IBM Case Manager products, and they are not covered in this publication.
- ▶ Global configuration
- ▶ Object stores

A typical tree view enables you to browse through and act on all the artifacts in the domain. Consider these tips:

- ▶ Tip 1: The information is often displayed as a series of tabs.
There can be many tabs; sometimes, there are too many tabs to display all of them at the same time in the browser window, so scroll buttons are provided.
- ▶ Tip 2: If you cancel a wizard, you are prompted to confirm the cancellation request. Click **OK** to confirm the cancellation request. Clicking Cancel takes you back to the wizard.
- ▶ Tip 3: Right-click items in the tree view to see the context menu.
- ▶ Tip 4: Tabs do not close unless you specifically click the “x” to close them. Although this feature can result in a crowded window, it also makes navigating between items quicker, because the information is already loaded in the browser cache.
- ▶ Tip 5: A tab can have subtabs. Ensure that you know in what context you are working.

Figure 10-13 on page 343 illustrates some of the features that are mentioned in the list of tips.

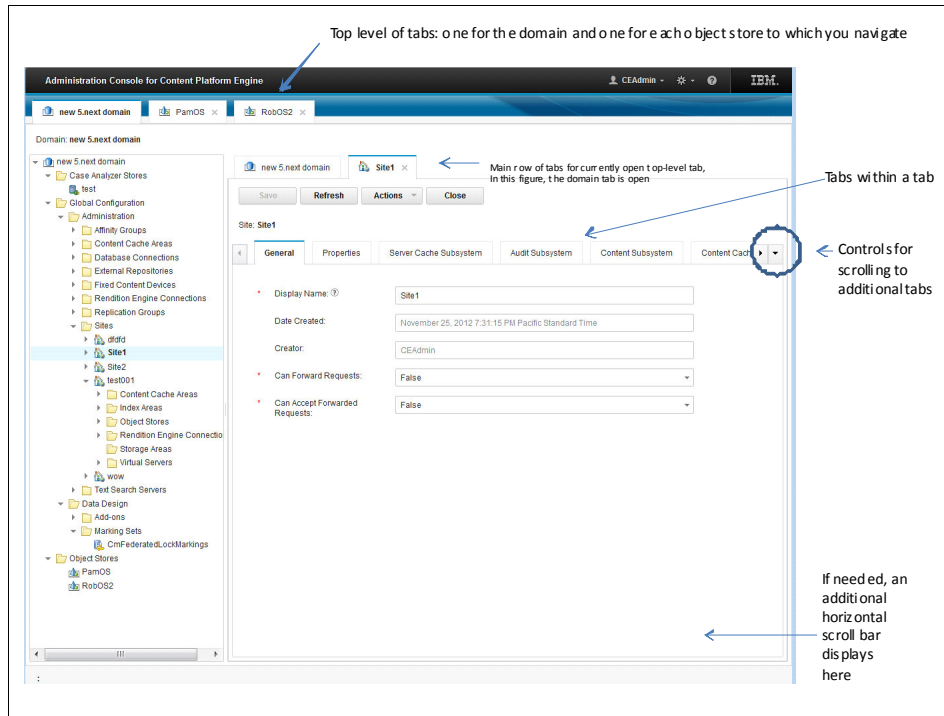


Figure 10-13 ACCE layout

Domain-level settings

The following tabs are available at the domain level. Many settings can be altered. However, unless you want to enable a capability that is by default disabled, or you are trying to resolve a particular issue, we highly advise that you *leave the settings unchanged*, especially because many of the settings can be overwritten at the object-store level. The following settings are domain-level settings:

- ▶ **General**
Use to set the URL to the P8 Information Center to make it easier to get information when you use ACCE.
- ▶ **Properties**
Provides a concise listing of the domain-level artifacts, such as the number of external repositories, object stores, and content cache areas.
- ▶ **Security**
Lists the default security settings for the P8 domain. Whenever a new object is created in the domain, these settings will be applied by default. However, they can be manually overwritten if needed.

- ▶ Directory configuration
Provides the details of the currently configured LDAP environment. The initial set of values is defined when the Content Platform Engine software is configured by using the Content Manager configuration tool. After the initial installation and configuration are verified, use the Directory configuration tab to add information about additional LDAP environments or to modify the existing configuration.
- ▶ Server cache subsystem
Use to define the refresh time for and maximum size of the different types of cache, including the user cache, and also to set the default and maximum number of objects that can be returned by a search.
- ▶ Audit subsystem
Use to configure the pruning of the audit log.
- ▶ Content subsystem
Use to maximize the throughput of content to clients and to configure thumbnail generation.
- ▶ Content cache subsystem
Use to define the size and location of the content cache.
- ▶ Text search subsystem
Use to enable the text search capability that is provided by Content Search Services (CSS) and to optimize the settings for the text extract, indexing, and searching capabilities.

Tip: Enabling the text search capability does not cause text to be indexed. It just makes the feature available. You must also navigate to the specific object stores and document classes that have the content to be indexed and enable text searching at those levels as appropriate.
- ▶ Trace subsystem
Use to configure trace log options. The trace logs are usually required when more information is needed on an error that has been logged in the content error log or when determining the cause of slow performance. But, be selective about the components to trace and the amount of time that tracing is enabled because the tracing will affect system performance and generate a large amount of log information.

- ▶ Sweep subsystem
Use to configure how often to run the sweep processes and what resources to allocate to the sweep processes. The actions of the sweep runs are defined at the object store level and enable you to ensure that old content is removed from the system in a timely fashion, and perform bulk operations that are related to setting retention times and thumbnail generation.
- ▶ Replication subsystem
Used with FileNet Content Federation Services for Image Services to define the available resources for replicating information from Image Services to Content Platform Engine, and from Content Platform Engine to Image Services.
Use the replication subsystem options to stop the Content Platform Engine from processing any federation requests. This capability can be useful when you need to perform maintenance work on the Content Platform Engine or Image Services server.
- ▶ Publishing subsystem
Use to configure the rendition processing that is available with Rendition Engine, an optional add-on to the IBM FileNet Content Manager suite.
- ▶ Asynchronous processing
Use to enable and disable event processing, and to optimize the event processing by the wait time, timeout setting, number of workers, and how often failed events need to be tried again.
- ▶ FileNet Content Federation Services import agent subsystem
Use to enable and disable the processing of FileNet Content Federation Services for Image Services and FileNet Content Federation Services-Content Manager OnDemand federation requests.
- ▶ Workflow subsystem
Use to configure the available resources for workflow and case analyzer processing.

Recommendations: Ensure that dispatchers that are not being used are disabled. For example, disable the asynchronous processing if events are not being processed. Each dispatcher issues regular queries to look for work, so if there is no work to look for, you can save system resources by disabling the dispatcher.

Site-level settings

Every P8 domain has at least one site. In a geographically distributed environment in which you have configured multiple sites, it might be appropriate to overwrite P8 domain-level configuration settings with site-specific settings. Some site-level settings can also be further refined at the object store level.

To access the site-level information, in ACCE, navigate to **Global Configuration** → **Administration** → **Sites**.

The following tabs are available at the site level:

- ▶ **General**
Besides providing general information about the site, such as its name, use this tab to specify whether requests can be forwarded to or from this site. This option is not available at the domain level.
- ▶ **Properties**
- ▶ **Server cache subsystem**
Use to configure the cache for various objects, including user tokens, marking sets, and the GCD.
- ▶ **Audit subsystem**
Use to configure pruning of the audit logs.
- ▶ **Content subsystem**
Along with the content cache subsystem, use this tab to optimize the upload and download of content by clients.
- ▶ **Content cache subsystem**
Use to define the location of the content cache areas and the number of elements that can be stored in the cache.
- ▶ **Text search subsystem**
Use to optimize the indexing of and the searching for content.
- ▶ **Trace subsystem**
Use to configure trace logging. Trace logging is often needed when you are troubleshooting issues with the environment and with custom applications.
- ▶ **Sweep subsystem**
Use to enable the sweep capability and to build sweep schedules. The sweep processes are defined at the object store level and can be used to perform bulk updates, move content, and manage queues.

- ▶ Replication subsystem
Use with FileNet Content Federation Services for Image Services to manage the frequency with which updates to the Image Services repository are replicated to the object stores, and vice versa.
- ▶ Publishing subsystem
Use with Rendition Engine to manage publishing processes.
- ▶ Asynchronous processing subsystem
Use to enable and disable asynchronous event processing and to optimize the processing of the events. When events are generated, a row is entered into the queueitem table. After an event is successfully processed, the row is removed from the table. By default, if an event fails to process successfully, it will be tried again up to seven times. Two columns, retry count and next retry date, in the queueitem table track the number of retry attempts and the next time an attempt will be made to retry processing an event that previously failed.

Tip: Avoid large backlogs in the queueitem table. Set up regular queries against the queueitem table to track the backlog, event processing throughput, and event processing failure rate.

- ▶ FileNet Content Federation Services import agent subsystem
Use with FileNet Content Federation Services for Image Services to manage the initial federation of documents from Image Services to object stores.
- ▶ Workflow subsystem
Use to manage the processing throughput of workflows and Case Analyzer.

GCD level artifacts

The following artifacts are defined at the GCD level so that they can be used with any of the object stores in a P8 domain:

- ▶ Affinity groups, which are groups of index servers and index areas for use with CSS.
- ▶ Content cache areas, which can improve the speed at which content is delivered to clients.

Typically, *content cache areas* are storage areas that are more local to the clients than the object store storage areas. Content can be loaded into a cache storage area when it is first added to the object store, or when it is first accessed by a client. You also define settings, such as how large the cache can grow, the number of elements that can be in the cache, and the rules for pruning the cache.

▶ Database connections

A Java Naming and Directory Interface (JNDI) XA and non-XA data source pair defines a JDBC connection to a database available to the IBM FileNet Content Manager software. You define the data source pairs by using the Configuration Manager.

In ACCE, you define database connections as “*labels*” to a data source pair. And then, as you define object stores and workflow systems, you identify which database connection (and therefore database) to use.

Object stores and workflow systems can share databases, which can simplify the maintenance of the P8 environment. Ensure that if you combine databases that it does not adversely affect any of these areas:

- Application requirements
- Backup and restore schedules
- Data independence requirements

▶ External repositories

These repositories exist outside of the current P8 domain whose content can be made available by using FileNet Content Federation Services.

For more information about FileNet Content Federation Services, see *Federated Content Management: Accessing Content from Disparate Repositories with IBM Content Federation Services and IBM Content Integrator*, SG24-7742.

▶ Fixed content devices

These storage devices, such as IBM Tivoli Storage Manager, EMC Centera, and Network Appliance SnapLock, can be used to store object store content. A full list of the supported devices is provided in the IBM FileNet P8 Hardware and Software Requirements guide, which can be downloaded from the following page:

<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27013654>

▶ Rendition Engine connections

Used to configure Darwin Information Typing Architecture (DITA) and P8 Rendition Engine connections.

▶ Replication groups

A *replication group* is used to connect an external repository with an object store.

- ▶ Sites

A *site* is a logical grouping of P8 domain resources. This feature is used primarily with geographically dispersed clients and multiple data centers. You can use this feature to allocate local resources to clients and limit the amount of WAN traffic.

- ▶ Text search servers

These servers are for use with CSS. Each server can be used to perform text indexing, text searching, or both.

- ▶ Add-ons

Add-ons are modules that can be added to any object store to support specific functionality. Several add-ons are supplied with the IBM FileNet Content Manager product to support functionality provided by Content Platform Engine and FileNet Workplace XT. Other products in the P8 Suite, such as IBM Enterprise Records, as well as custom applications, can also provide (and require) additional add-ons.

Tip: When creating object stores, use only the add-ons that you are sure will be needed. Additional add-ons can be added later, but after they are added to an object store, they cannot be removed.

- ▶ Marking sets

Marking sets are special properties that help control access to objects.

Object store management

ACCE is also used to define and manage object stores. In this section, we focus on the administration and maintenance of object stores. For object store design guidance, see Chapter 4, “Repository design” on page 81. Object store management includes these functions:

- ▶ Searching

Search capabilities are provided in IBM Content Navigator, FileNet Workplace XT, and ACCE. The facilities in IBM Content Navigator and FileNet Workplace XT allow you to search for documents in the object stores. The search capability in ACCE enables you to locate other objects in addition to documents. For example, you can use the search capability to determine how many events are waiting to be processed or how many events are waiting to be tried again because a previous attempt at processing the event failed.

When objects have been found via search, you can then select one or more of the objects and update them if needed.

► Making bulk updates

You can use these types of bulk updates:

- Updating metadata or security settings on a set of documents

Sometimes, you need to make a similar update to a number of objects. For example, if 100 documents were filed into the wrong folder, instead of refiling the documents individually, you can search for the objects and then use a bulk update to refile them all at once.

- Moving content to a different type of storage or a different storage location

This type of move can be necessary for a number of reasons, including wanting to replace a storage device or to move older, less frequently accessed content to cheaper storage.

Moving content is accomplished via the bulk sweep process. You define what the sweep will do at the object store level, and set a default schedule at the domain level.

► Configuring and disabling dispatchers

Dispatchers are used to manage work that is initiated via processes, such as federation and asynchronous event processing. The dispatchers poll the appropriate queues for work and then pass the work along to the threads or workers who then perform the required task.

By default, the dispatchers are enabled. If system performance is an issue and the dispatchers are not needed, consider disabling them by using the check boxes on the following pages. Also, use these pages to optimize the number of dispatchers and worker threads for each type of process:

- Text search
- Replication for federation
- Asynchronous processing

Configuring and optimizing asynchronous processing is handled at the domain level.

► Using recovery bins

By default, when a user or application deletes an object, the object cannot be recovered. However, Content Platform Engine also provides a “soft delete” method. With a soft delete, the object is placed into a recovery bin. The security settings on the recovery bin determine who can restore documents from the recovery bin and who can empty it.

If recovery bins and soft deletes are used in applications, work with the designers of the applications to answer the following questions:

- How many recovery bins are needed?
- Who can restore content from recovery bins?
- Who can empty recovery bins?
- When is it appropriate for the object store administrator to empty and remove recovery bins?

▶ Defining sweep processes

Sweep processes can be defined for bulk updates, retention management, and queue management. Build the default schedules for these processes at the domain level. But, build the sweep definitions, which the individual sweep process will accomplish, at the object store level.

▶ Configuring connection points

Connection points provide a link to a workflow system region. Applications that employ workflow functionality use a connection point to identify the workflow system region that will be used to process the workflow.

10.8.3 Consistency checker

Consistency checker is one of the tools that can be installed when the Content Platform Engine server is installed on a Microsoft Windows platform. Use the tool to perform the following tasks:

- ▶ Detect inconsistencies between the content in file storage areas and the metadata in the associated object store database.
- ▶ Update the storage area statistics after an upgrade.

10.8.4 Database tools

The tools that are provided by your database vendor play an important part in managing the IBM FileNet Content Manager environment.

Backups

Ensure that regular database backups are taken. As object store content is likely stored in file storage or on fixed content devices, the database backups need to be coordinated with backups of the data on the storage devices and also cover any temporary storage areas. In addition, if content search indexes and workflow systems are part of your environment, the backup strategy must include the content that they generate, too. If data has to be restored, everything must be restored to the same point in time.

You can use both hot (or online) and offline backups with IBM FileNet Content Manager environments as long as you ensure that the backups of all the components are synchronized and can be restored to the same point in time.

You must test restoring from backups *regularly*, both as part of general environment maintenance and for disaster recovery.

For more information about backing up and restoring IBM FileNet Content Manager environments, see 10.13, “Backup and restore” on page 364.

Tuning

When you first deploy a new P8 solution, ensuring that the databases are tuned appropriately is a key element of applications that have good response times. Look for these items:

- ▶ Adequate number of available database connections
- ▶ Appropriate indexes to improve search performance

Create indexes on individual object store properties via FEM. Follow these steps to identify a property as an index item:

- a. Navigate to the class that uses the property.
- b. Display the properties of the class.
- c. Click the **Property Definitions** tab.
- d. Select the property from the list, and then click **Edit**.
- e. On the General tab, use the set/remove index option.

Complex indexes must be created by using the database vendor tools.

- ▶ Cache “read ahead”

Maintenance

Monitor the database for these conditions:

- ▶ Available space for items, such as tables, indexes, logs, and journal files
- ▶ Structures that need reorganizing or that have space that needs reclaiming

If many objects, for example, documents or events, are regularly being added and deleted, regularly reorganizing the appropriate tables or rebuilding indexes can have a positive effect on performance and throughput.

Counting database objects

Normally, all access to an object store database needs to be via the Content Platform Engine APIs, but there are circumstances when querying the database tables can be appropriate. For example, because there is no count mechanism in the Content Platform Engine APIs, you might need to track the following information:

- ▶ Total number of objects in the object store as a part of a plan for rolling to a new object store
- ▶ Number of rows in the queueitem table to ensure that event processing is occurring as expected
- ▶ Number of rows in the DocVersion table to ensure that content is being added to the object store in the expected volumes

10.8.5 Application server administration tools

Although you perform the initial setup of the Content Platform Engine by using the Configuration Manager, you need to use the application server administration tools for performance tuning. Update these items by using the application server administration tools:

- ▶ JVM memory settings
- ▶ Garbage collection
- ▶ Thread allocation
- ▶ Connection pools for the database and LDAP

10.8.6 Workflow system tools

Various tools for designing workflows, configuring process regions, and monitoring running workflows are provided as Java applets in FileNet Workplace XT. In addition, a series of administrative command-line tools, such as **vwtool**, are installed as part of Content Platform Engine. For more information about these tools, see the information center and *Introducing IBM FileNet Business Process Manager*, SG24-7509.

10.8.7 IBM Content Navigator tools

Tools are provided with Content Navigator for performing tasks such as configuring desktops. For more information about Content Navigator, see *Customizing and Extending IBM Content Navigator*, SG24-8055.

10.9 Reducing storage costs

It used to be that storage costs were a small component of an IT budget, but with the increase in the use of electronic information and the plethora of documents that must be stored for legal and compliance reasons, storage costs are an issue.

Before you can reduce storage costs, you need to know the following information:

- ▶ What you are storing
- ▶ Why you are storing it
- ▶ Where you are storing it
- ▶ How long do you need to store it
- ▶ What access is needed

With this information, you can design a storage plan that enables you to set up the following rules:

- ▶ Set retention rules when content is added to a repository
- ▶ Update retention rules when requirements change
- ▶ Develop sweep rules and schedules that enable you to perform these tasks:
 - Delete content that is no longer needed
 - Move content to lower-cost storage when it is accessed less frequently

10.9.1 Retention rules

Retention rules identify the minimum length of time that an object, such as a document, annotation, folder, or custom object, must be kept. After the retention date has passed, from an IBM FileNet Content Manager perspective, the object is eligible for deletion. However, other applications, such as IBM Enterprise Records, might have placed holds that will continue to prevent the objects from being deleted.

Fixed content devices

For documents that are stored on a fixed content device, IBM FileNet Content Manager supports both static and dynamic retention models. The date at which the document is eligible for deletion is set at ingestion time. Then, at some future point, the retention can be altered.

When using this retention model, the deletion happens in two stages:

1. A delete request must be initiated from the IBM FileNet Content Manager.
This request “removes” the document from the object store, and the document is no longer visible to IBM FileNet Content Manager applications.
2. In the background, IBM FileNet Content Manager calls the fixed content device to delete the content. Since IBM FileNet Content Manager previously determined that the content is eligible for deletion, deleting the content from the fixed storage device is allowed and successful.

Best Practice: Allow IBM FileNet Content Manager to control the retention on fixed content devices. Do not define default retention periods directly on the fixed content device.

File and database storage

Documents and annotations stored on file and database storage areas can take advantage of the IBM FileNet Content Manager retention capabilities that support setting and updating retention rules throughout the document lifecycle. This retention capability enables documents to be ingested with an “*I know I want to keep this, but I do not know how long I want to keep it*” rule, as well as more specific rules. The content stored on a file or database storage area is not under device retention. This content is controlled by IBM FileNet Content Manager retention only.

As with documents that are stored on fixed content devices, documents are not automatically deleted when they reach their “expiration date”. Instead, you must run regular sweep processes to look for documents that are ready to be deleted.

Folders and custom objects

Retention rules can also be set on folders and custom objects. These objects are stored in the object store database. And, as with documents, you must run regular sweep processes to delete the objects that have met their expiration date.

10.9.2 Using the sweep framework

IBM FileNet Content Manager provides a sweep framework that is an efficient way of acting on all the rows or a subset of rows in a single table.

Use the sweep process to perform any of the following tasks:

- ▶ Dispose of objects that have met their retention requirements
- ▶ Update how long an object must be kept
- ▶ Move content to different storage

The sweep framework also supports thumbnail generation and batch printing.

There are three forms of sweep:

- ▶ Single sweep
Use this form of sweep to complete a one-time batch task, such as moving documents that have been incorrectly filed.
- ▶ Policy-controlled
Use this form of sweep to automate regular maintenance tasks, such as deleting documents that have passed their required retention dates and moving documents to lower-cost storage.
- ▶ Queue sweep
A special form of sweep that is used by IBM FileNet Content Manager for queue operations, such as thumbnail generation.

10.9.3 Monitoring storage and cache usage

Statistics on storage and cache usage are available within ACCE. Monitoring this information enables you to determine whether these conditions exist:

- ▶ Under-utilized storage
- ▶ A need to add additional storage

This information can also help when you need to bill organizations for their storage usage.

Storage statistics

The following information is available for file storage areas. Monitor this information to ensure that the file storage is configured appropriately for your environment:

- ▶ Number of files currently stored
- ▶ Number of bytes of information currently stored in the file storage area
- ▶ Date on which the file storage was last modified
- ▶ Number of files added to the storage area
- ▶ Number of files removed from the storage area
- ▶ Maximum number of files that can be added to the storage area
- ▶ Maximum size to which the storage area can grow

Restriction: ACCE does not provide similar information for fixed content devices. Instead, use the tools provided by the fixed content device provider.

Figure 10-14 on page 357 illustrates the storage statistics information provided in ACCE.

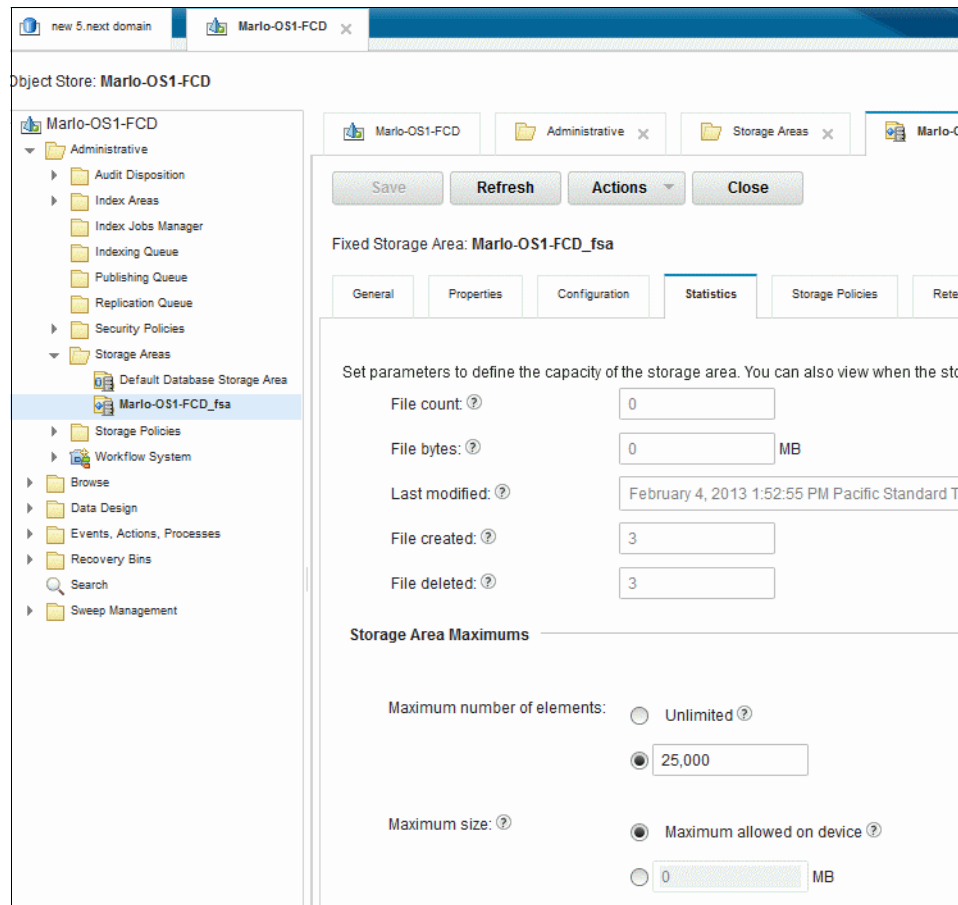


Figure 10-14 File storage statistics

Cache subsystems

There are multiple cache subsystems that can be tuned. You can configure the subsystems at the P8 domain level and at the site level.

Consider these best practices for cache subsystems:

- ▶ Use the default settings and only make changes if specific use cases require them.
- ▶ Only make changes at the site level if there are specific characteristics of that site that require different settings.

- ▶ Track the changes you make, especially if you are making them at the site level.

Figure 10-15 displays the ACCE page for modifying the site-level cache subsystems. In addition to the subsystems shown, there are also subject and metadata merged scope subsystems.

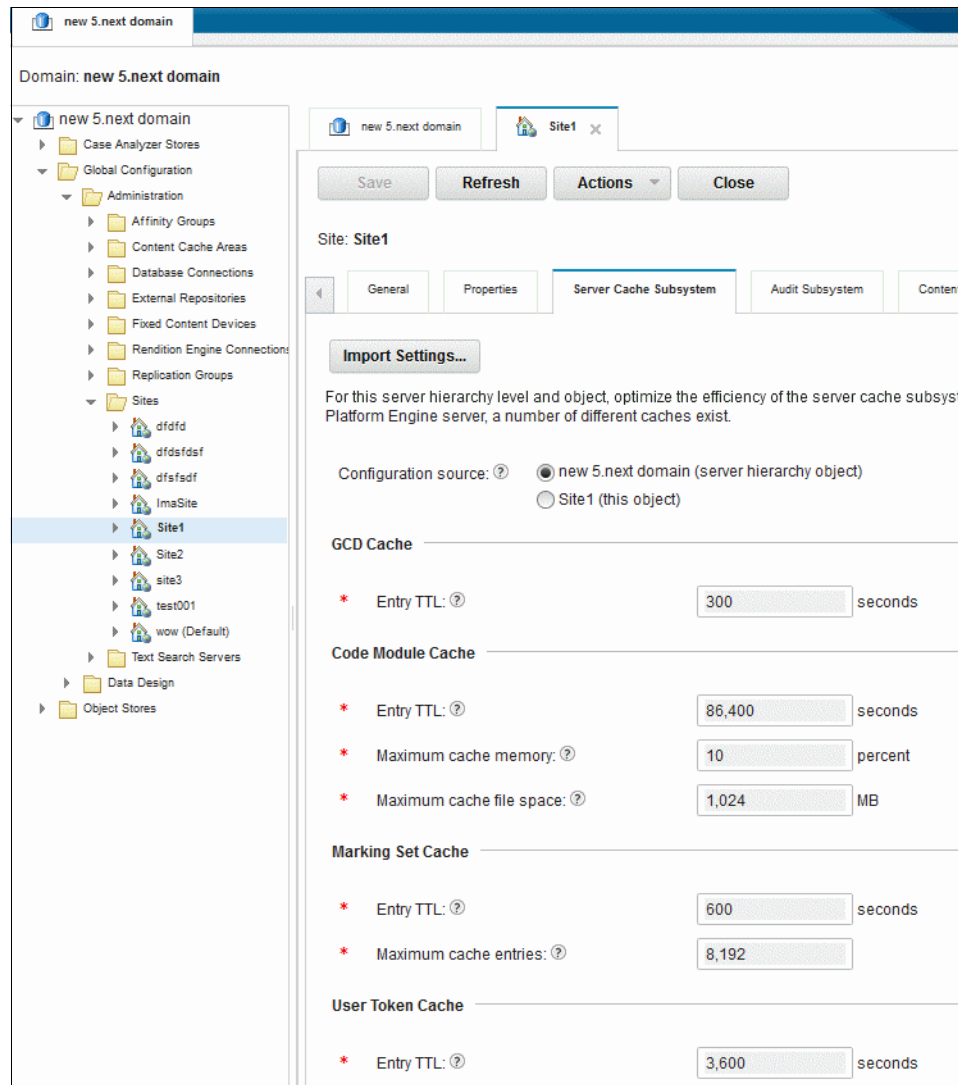


Figure 10-15 Server cache subsystems

Content cache statistics

Content caches are configured at the object store level and determine when content is added to the cache, how long the content stays in the cache, and how large the cache can grow.

Using a content cache effectively improves the speed at which clients gain access to content. By default, content is added to the cache when a client accesses a document, so that subsequent accesses will be quicker. You can also add new content to the cache automatically.

The content cache information needs to be used as a guideline of activity. The following information is provided:

- ▶ Number of files in the cache
- ▶ Current size of the cache
- ▶ Number of files added to the cache since the cache was last cleared
- ▶ Number of files removed from the cache since the cache was last cleared

10.10 Using virus scan software

If the servers that host IBM FileNet Content Manager run virus scan software, be aware that the following situations might occur:

- ▶ Slow installation
Consider disabling the virus scan software when you install or upgrade IBM FileNet Content Manager components.
- ▶ Slow data uploads
If uploading large files or many files, consider disabling the real-time scanning feature of the virus scan software during the upload.
- ▶ File size corruption
Do not use virus scan software on the IBM FileNet Content Manager file storage areas. Virus scan software can alter the physical size of a file. When a client attempts to download a file, IBM FileNet Content Manager checks the physical size of the file against the size of the file that was uploaded. If the two file sizes are different, the download fails.
- ▶ Access-denied errors
Some virus scan software locks files while they are being scanned, which can cause operations that require access to these files to fail.

Recommendations: Check with your database and cluster software vendors to determine whether these components can be adversely affected by any virus scanning software.

10.11 Applying fixes

IBM FileNet Content Manager environments are never, and must never be, static. IBM regularly releases fix packs and new releases, so it is important to have a plan for picking up the latest versions of the IBM FileNet Content Manager software regularly. Keeping the software current makes troubleshooting issues, as well as obtaining fixes for issues that are specific to your environment, easier. It also ensures that you are running software that is still supported by IBM as part of a regular support contract.

The maintenance plan must cover updating the following software:

- ▶ Custom applications
- ▶ Infrastructure updates, including application server, database server, and directory server updates
- ▶ IBM FileNet Content Manager component updates

The plan needs to cover the procedures for when and how to make these updates, as well as the regression testing that is required to ensure that the changes work as expected.

The more frequently updates are applied to the environment, the easier it is to apply the updates:

- ▶ Avoid having to change multiple components at the same time
- ▶ Reduce the risk of a regression
- ▶ Keep your team up-to-date on the upgrade procedures

10.11.1 Tracking fixes

Track the level of software that is installed in all your IBM FileNet Content Manager environments (development, test, production, and so on).

Tip: Ensure that there is one environment running the exact same software as production, so that if an issue occurs in production, you can test the fix prior to applying it to the production environment.

You need to track the software levels for these reasons:

- ▶ Ensure that you use the correct procedures when you update software. Depending on the software level currently in place, the upgrade procedures might vary.
- ▶ If issues arise, you can provide the appropriate details to IBM Support so that they can help resolve the issue as quickly as possible.
- ▶ When you plan an upgrade, you can ensure that any fixes that you have applied to your current environment have been rolled into the release to which you want to move.

Important: Just because a release is made available *after* you install a fix in your environment, do not assume that your fix is included in the release.

10.11.2 Checking compatibility and build numbers

If you have a build number but you are not sure what release it matches, or if you need to check that a fix you want to install is compatible with other FileNet P8 components in your environment, see the FileNet P8 Fix Pack Compatibility Matrices.

The matrices can be downloaded from the following URL:

<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27014734>

There is a separate matrix for each major IBM FileNet Content Manager release. Column B provides the build numbers.

10.11.3 Reporting issues and downloading fixes

To report an issue, use the IBM Support portal (you must be registered):

<https://www.ibm.com/support/entry/myportal/overview>

All IBM FileNet Content Manager fixes are made available via the IBM website called Fix Central:

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

Tip: All FileNet Workplace XT fix packs are full installations of a complete FileNet Workplace XT package and not merely incremental updates.

Important: Review the list of fixes included in the package to determine whether any of them might negatively affect the functionality that is used in your environment. If so, ensure that any regression testing includes these changes.

Prior to installing a software update, ensure that you review the readme file and verify that the fix is compatible with the software in your environment and that it resolves your specific issues.

Also, consider signing up to receive automatic alerts for critical updates and issues:

<http://www.ibm.com/support/electronicssupport/preventproblem.html>

10.12 Updating security

As your system grows, you might find it necessary to add users and groups to create or access content. Although content security can be added for specific users, a best practice is to use security groups. Users can easily be added to the group to gain the security roles that they need, as well as easily removed from a group when their role changes and they no longer need access to the content. Securing content to a specific user requires maintenance to find content and add security for them as user roles tend to change over time.

To update an object store with new users or groups, use IBM FileNet Enterprise Manager's Security Script Wizard to run the **0SecurityUpdate.xml** script. IBM FileNet Enterprise runs on Windows operating systems only and is installed by using the Tools option in the Content Platform Engine installer.

Recommendations: Although there are ways to apply security to individual objects, using the Security Script Wizard is the only way to ensure that security is set correctly for the entire object store. Failure to use the wizard can result in users not having the correct permissions to access or create content.

To update an object store with new users or groups, follow these steps:

1. In the IBM FileNet Enterprise Manager, right-click the object store node, select **All Tasks**, and run the **Security Script Wizard**.
2. When prompted to select an XML security script information file, browse to and select **0SecurityUpdate.xml**. It is installed in the installation base directory:

FileNet\ContentEngine\Scripts\Component Library\

3. When prompted to define security roles, you see two roles under Security Role: Object Store Administrators and Object Store Users.

Click **Add** to add security participants for the selected role. The Select Users and Groups dialog box opens. Click **OK** when you have added the participants for that particular role. See Figure 10-16 on page 363, which shows the Security Script Wizard.

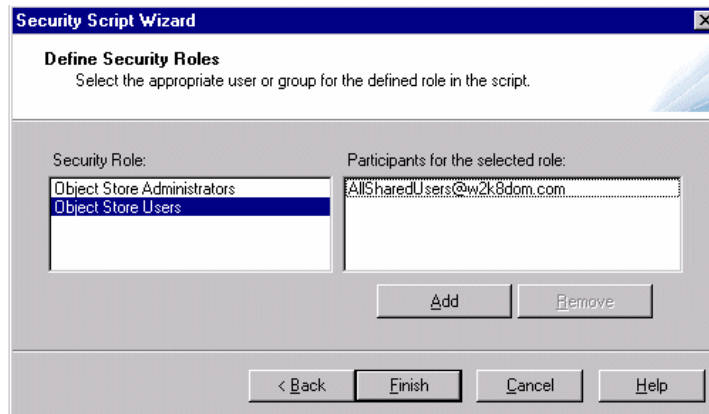


Figure 10-16 Security Script Wizard interface

4. Click **Finish** when you are done. The wizard generates a prompt informing you where its log file will be located. The wizard proceeds to apply the security permissions to the objects in the object store. This process can take time, depending on the number of objects that need to be updated. The wizard reports when the process of applying security is complete.
5. If you added groups to only one Security Role, a notice appears (see Figure 10-17). Click **OK** to continue. This notice appears because no current Security Roles will be deleted; only the new roles will be added by the wizard.

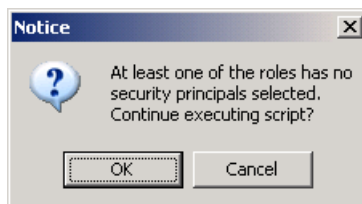


Figure 10-17 Security wizard notice

The Security Script Wizard sets permissions on the root folder in the object store, but it does not directly modify the security assigned to individual documents and custom objects. Depending on how inheritance has been configured, the document and custom object permissions might be inherited from the root folder.

You can read more detailed information by selecting **ecm_help** → **FileNet P8 Administration** → **Content Platform Engine Administration** → **Managing Security** → **Security Script Wizard**.

10.13 Backup and restore

In this section, we discuss IBM FileNet Content Manager backup and restore.

Chapter 7, “Business continuity” on page 217, discusses types of events that can require a system restoration. It focuses on building a highly available environment with protection against catastrophic system or site loss to ensure that your system is always available. If you are responsible for system recovery, familiarize yourself with business continuity methods whether your budget permits a hot site or not. You might be able to use some business continuity methods to reduce backup and restore times in your data center. If your budget permits a hot site, you still need a backup and restore mechanism to recover from human errors, such as deleted or modified files. A mirrored hot site mirrors all activity; it lacks a means to differentiate an intentional or accidental change.

Recommendations: Store your backup media off-site away from your primary servers. Make sure that the media is moved to the off-site location as soon as possible after the backup completes.

The longer your backup media is stored near your primary servers, the greater the chance that a catastrophic event can destroy both your servers and your ability to restore your systems to operational condition.

IBM FileNet Content Manager does not provide backup software. You must use backup utilities that are supplied with your operating system or database or by third parties.

10.13.1 System components requiring backup

This list shows the system components that require backup:

- ▶ Databases (all tables or table spaces and schema for your system)
- ▶ File storage areas if used or configured, including fixed content storage areas

- ▶ Content search files and indexes
- ▶ Server operating system, Java EE environment, and all IBM FileNet Content Manager installed software

You can choose to omit the operating system and software backup. In the event of a failure that requires a restore operation, this choice requires a reinstallation of all components on a server, which increases the time required to return the server to normal operations.

- ▶ Lightweight Directory Access Protocol (LDAP) security system

If you are using the user ID's security identifier (SID) as the unique identifier for the user, ensure that the SIDs are maintained during a restore. IBM FileNet Content Manager uses a unique identifier for security, which by default is the SID. Simply re-creating deleted users or groups does not work because re-creating deleted users or groups typically creates a new unique identifier. If your system is configured to use a different type of unique identifier, for example, email address or employee ID, these attributes can be re-created when a user account is added back to the LDAP.

- ▶ Any external systems with which your IBM FileNet Content Manager application operates

Typical IBM FileNet Content Manager installations operate in concert with existing applications. Examples are Customer Relations Management systems, database applications, and mainframe applications. Their data needs to be backed up at the same time that your IBM FileNet Content Manager system is backed up to ensure full data consistency.

Tip: If your system uses Fixed File Storage areas for compliance or Image Manager applications, you need a normal file storage area for temporary staging of content. If your application performs content reservations or uses annotations, that metadata is stored in the “temporary” file store. This file storage area must be included in your backup and recovery strategy.

10.13.2 Offline backup

An offline backup is the preferred method for IBM FileNet Content Manager. An offline backup ensures that all application data is in a consistent state. When a restore becomes necessary, all data must be recovered to the same point in time.

A *backup window* is the amount of time that your system can be down for backup. If your system has users running from 6:00 a.m. to 11:00 p.m., you have a seven hour backup window. A best practice is to allot time before and after users require the system to accommodate late workers or a backup that runs longer than usual. We advise allotting 1/2 to one hour before and after users

expect the system to be operational. In this example, allotting one hour before and after gives you a five hour total backup window to stop the servers, perform the backup, and start the servers.

Typical installations store content in a file system, metadata in a database, work items in a process database, and pointers to the content in external systems. The amount of time that is necessary to back up the individual system components can vary by minutes or hours.

The amount of time required for the longest component's backup must fit within your backup window. Your content storage area usually consumes the greatest amount of backup time.

There are a few steps that you can take to decrease backup time to fit your window:

- ▶ Use a combination of full and incremental backups. *Incremental backups* simply capture information that has changed since the last backup. This can greatly reduce time spent backing up data. During a restore, you must restore from your last full backup and apply the incremental backups before starting your system, which increases the amount of time necessary to restore your system. A best practice is to perform full backups weekly when a larger backup window is available and perform incremental backups during the week when your backup window is smaller.
- ▶ If you use tape as your backup media, a faster alternative is to back up your data to disk files. When the backup to disk completes, transfer the backup files to tape, which allows your IBM FileNet Content Manager system to run while the transfer to tape occurs.
- ▶ The next section describes potential methods to run online backups. Those techniques can safely be used for offline backups. Simply stop your IBM FileNet Content Manager servers, run the copy, and restart your system. This approach provides the fastest possible offline backup.
- ▶ If your backups cannot be completed within your backup window, you need to look at the online backup methods discussed next.

10.13.3 Online backup

You need to investigate online backup alternatives if your system must run 24x7, your backup time exceeds your backup window, or your service-level agreements (SLAs) require a higher frequency than a nightly backup. Online backups are also referred to as *hot backups*.

The issue with online backups is ensuring consistency in your backups. As mentioned in 10.13.2, “Offline backup” on page 365, backup times can vary between different components. If your IBM FileNet Content Manager database backup completes in 30 minutes, but your file store backup runs three hours, it is highly possible that when a restore is performed, your database might not have metadata pointers to all the files in your file store. The result is an inconsistent system. IBM FileNet Content Manager provides a Consistency Checker utility (see 10.8.3, “Consistency checker” on page 351) that you can use to find inconsistent objects.

There are options on the market that can help resolve this situation. IBM FlashCopy, NetApp, disk, volume, or storage area network (SAN) mirroring techniques are available that permit point-in-time backups or snapshots of your data. These options typically work similarly to the disk mirroring that has been used for many years. Where they differ is that they mirror several disks or volumes in groups and permit adding point-in-time details. Restoring involves copying the mirror back to the last good point in time. Several techniques offer offline tape backup of the mirror and point-in-time copies. Ideally, the utilities provide a means of capturing consistent slices across all disk drives and servers used by your application.

You can use the IBM Lab Services offering to help you create an online backup strategy.

Section 7.4.1, “Disaster recovery concepts” on page 235 discusses methods that use these techniques to copy your data to a remote facility. The same techniques can provide copies in your primary data center. Most storage vendors offer local and remote mirroring capabilities for this copying. It might be called a *point-in-time, snapshot, or flash backup* capability. Most storage vendors also provide tape backup solutions to move the data off-site.

Check whether your database vendor has any special requirements for using these techniques for system backups; most vendors have special requirements. Consider also using an offline database backup for additional safety.

10.13.4 System restore

There is no particular required component order when a system restore becomes necessary. Your LDAP security and databases need to be operational before you start IBM FileNet Content Manager after a restore. Typically, you restore information in this sequence:

- ▶ LDAP system
- ▶ Database server
- ▶ IBM FileNet Content Manager server operating system

- ▶ Application servers
- ▶ Content Platform Engine
- ▶ File stores if used or configured
- ▶ Other IBM FileNet Content Manager components
- ▶ Any external systems with which your IBM FileNet Content Manager application operates

Your IBM FileNet Content Manager system needs to be down during the restore process. If you used incremental backups, restore all incremental backups before you start your IBM FileNet Content Manager system. After all restores are completed, start your IBM FileNet Content Manager system normally.

Recommendations: Consistency checks can run for a long time depending on the amount of content in your system. Limit the amount of time that the consistency check runs. Set the check to start a few hours before the major event that requires its use.

10.13.5 Application consistency check

If your application uses external systems, your application developers must consider creating tools to allow validating the consistency between your IBM FileNet Content Manager system and the external systems. There might be an event where you need to restore your IBM FileNet Content Manager system, and external systems cannot be restored to the same point in time. In those cases, you need a means to validate that content references in the external systems are on the IBM FileNet Content Manager system.

10.14 Task schedule

Table 10-2 lists the recommendations for the frequency of performing IBM FileNet Content Manager system administration tasks.

Table 10-2 Task schedule recommendations

| Task | Frequency | Comments |
|----------------------------|---------------|---|
| Monitor system | Daily | Processes, performance, and logs |
| Back up system | Daily | Databases, file stores, and LDAP service |
| Log maintenance | Weekly | See footnote ¹ |
| Check free space | Weekly | All file systems and databases |
| Check performance | Weekly | See footnote ² |
| Check for latest fix packs | Monthly | See footnote ³ |
| Database maintenance | Periodically | Consult your database vendor for periodic maintenance functions to keep the database optimized. Ensure that you meet their recommendations. |
| Backup software | Monthly | Operating systems, Java EE server, and installed software |
| Apply patches | Semi-annually | See footnote ³ |
| Test restore | Annually | A full system restore must be performed at least once per year on DR hardware. |

¹ Log maintenance must include all operating system, application server, and IBM FileNet Content Manager product error and trace log files. Log maintenance must also include the Content Platform Engine audit log and the Content Platform Engine log database tables, if used. All log files can grow quite large over time; on busy systems, you might need to increase the maintenance frequency. Low use systems might be able to reduce the frequency.

² “System Dashboard performance archiver” on page 332 describes how to archive performance logs. You can generate reports from the archived log files. If you use IBM FileNet System Monitor, you can configure it to keep archived performance data and generate reports, as well.

³ IBM FileNet fix packs are produced at regular intervals. Fix packs are available on Fix Central: <http://www.ibm.com/support/fixcentral/>

10.15 Conclusion

This list summarizes our recommendations in this chapter:

- ▶ Run the **archiver.jar** to capture performance data during peak hours of activity.
- ▶ Maintain message logs by renaming them and then deleting them after a period of time.
- ▶ Manage (clean up) audit and statistics logs weekly when used.
- ▶ Keep auditing as minimal as possible.
- ▶ Use security groups to secure content.
- ▶ Store your backup media off-site.
- ▶ Allot free time before and after the backup as part of a backup window.
- ▶ If you use incremental backups, perform full backups weekly.
- ▶ Run the Consistency Checker utility after you restore a system.

In Chapter 11, “Upgrade and migration” on page 371, we address upgrade and migration topics. In Chapter 12, “Troubleshooting” on page 387, we discuss troubleshooting techniques.



Upgrade and migration

In this chapter, we discuss the upgrade and migration of FileNet Content Manager environments. You learn about the best practices for planning, performing, and documenting upgrades and migrations.

We discuss the following topics:

- ▶ Terminology
- ▶ Planning for updates
- ▶ Upgrading to a new software release
- ▶ Migration best practices
- ▶ Special considerations for upgrade

11.1 Terminology

Specific terminology is used in the P8 Information Center and by the IBM FileNet Support teams for updating a FileNet Content Manager environment.

Understanding the terminology enables you to make informed decisions about your update process choices and the effort involved with each type of change.

11.1.1 Packages

There are several package types that you can apply to your FileNet Content Manager environment:

- ▶ New software release
- ▶ Modification (mod) release
- ▶ Fix pack
- ▶ Interim release
- ▶ Test fix

Software release

A new software release introduces new features and functionality, adds and removes support for infrastructure elements, resolves client-reported issues, and adds and removes support for components that interact with FileNet Content Manager.

Although new functionality might be added and existing functionality changed or removed, the new software release maintains compatibility with an earlier version for any application programming interfaces (APIs). However, APIs can be marked as “*deprecated*”. Deprecated APIs eventually are removed.

Mod release

A mod release, or *service pack*, provides a small set of new features, as well as resolutions to client-reported issues. The mod release provides a roll-up of fixes that are available in previous update packages, as well as fixes that are released for the first time.

Fix pack

A *fix pack* provides a roll-up of authorized program analysis report (APAR) resolutions that were previously provided as interim fixes, test fixes, or in a previous fix pack, as well as fixes not previously released. Each fix pack contains fixes from previous releases.

Interim fix

An interim fix provides the resolution to a few APARs, usually one, that are likely to be needed by multiple clients.

Test fix

A test fix, or limited availability fix, provides the resolution to a small number of APARs, usually one, that are required by a specific client. Test fixes are also known as *limited availability fixes*. If you need this type of fix, IBM provides you with specific download information and a password for extracting the fix package.

Interim fix and test fix packages are similar. The primary difference is the size of the target audience for the package.

11.1.2 Package naming conventions

P8 Content Manager software releases use a four-digit identifier:

<major release>.<minor release>.<mod release>.<fix pack>

The major release, minor release, mod release, and fix pack numbers are incremented based on the software package.

Software release

Software releases can be considered major or minor releases. The designation of major or minor is arbitrary, but in general, it tries to convey the quantity of changes introduced by the software release.

Examples:

- ▶ 5.0.0.0 indicates that the package is major release 5 of the software.
- ▶ 4.5.0.0 indicates that the package is minor release 4.5 of the software.

Mod release

Mod releases are identified by the third digit in the four-digit identifier.

Examples:

- ▶ 5.1.1.0 indicates that this package is the first mod release for the 5.1 software level.
- ▶ 4.0.2.0 indicates that this package is the second mod release for the 4.0 software level.

Fix pack

Fix packs are identified by the fourth digit in the release level and the notation FPxyz, where xyz identifies the number of the fix pack.

Examples:

- ▶ 5.1.0.2-P8CE-FP002 is the first fix pack that is on top of the Content Engine 5.1 release.
- ▶ 1.1.5.2-WPXT-FP012 is the 12th fix pack on top of the FileNet Workplace XT 1.1.5 release.

Interim fix

Interim fixes are identified by the notation IFxyz at the end of the package name, where xyz identifies the number of the interim fix.

Examples:

- ▶ 5.1.0.0-P8CE-IF004 is the fourth interim fix on top of the Content Engine 5.1 release.
- ▶ 5.1.1.2-P8CE-IF001 is the first interim fix on top of Content Engine 5.1.1 fix pack 2.

Test fixes are identified by the notation LAxyz at the end of the package name, where xyz identifies the number of the test fix.

Examples:

- ▶ 4.0.2.4-P8eF-LA001 is the first test fix on top of fix pack 4 for the eForms 4.0.2 release.
- ▶ 1.1.4.6-WPXT-LA003 is the third test fix on top of fix pack 6 for the FileNet Workplace XT 1.1.4 release.

11.1.3 Installation rules

The P8 Information Center provides details of the supported upgrade paths:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.planprepare.doc%2Fp8ppu097.htm>

Always refer to the product release notes, the readme file, and the IBM FileNet P8 Hardware and Software Requirements guide to determine whether there are specific prerequisites that must be met before installing the software.

Tip: The readme file might also direct you to the appropriate installation and upgrade sections in the P8 Information Center for some of the procedural information.

Major, minor, and mod releases contain full installers. Therefore, they can be installed on a system that has no previous version of the software installed, as well as on top of an earlier version of the software.

In some cases, such as FileNet Workplace XT, fix packs are also full installations. In other cases, such as with IBM Content Navigator, the fix packs must be installed on top of the product release that is indicated in the fix pack name. All fix packs are *cumulative*, which means that you can skip fix pack levels.

Interim fixes and test fixes are not usually cumulative; instead, they contain files that must be applied, often manually, to a specific level of software. To verify the content of the package and to determine any prerequisites, see the readme file that is supplied with the interim fix or test fix.

Caution: Before upgrading an environment that is updated with a test fix, check the readme file of the newer software update package to ensure that the package contains the fix that was originally provided by the test fix. If the test fix is not explicitly listed, check with your IBM representative for information about when the test fix will be made generally available. If necessary, request that a new version of the test fix is generated that is compatible with the newer software update package.

Software updates cannot be uninstalled independently or separately from uninstalling the associated product component. The exception to this rule is the packages that are installed by manually copying files. These updates can be “uninstalled” by copying the older versions of the files over the newer versions of the files, and by reversing any other steps described in the readme file that is associated with the update package.

11.1.4 Update types

Three terms are used to describe updating a FileNet Content Manager environment:

- Update

Refers to applying a small change, such as a fix pack, interim fix, or test fix to the environment. And, update is also used as a general term to cover any changes made to the environment, including upgrade and migration.

► Upgrade

Refers to applying a new release to the environment, such as moving from Content Engine 5.1 to Content Platform Engine 5.2. In an upgrade, there is no change in the hardware that is used by the components.

► Migration

Refers to applying a new release to the environment and redeploying the software on new hardware or in a new software configuration. Software configuration changes can include changing application server, moving components to new operating systems, and moving to a new database management system.

Updating, upgrading, and migrating FileNet Content Manager are standard maintenance tasks and need to be scheduled on an on-going basis. These changes need to be coordinated with other system maintenance tasks, such as updating application server levels and applying operating system fixes.

Another term that is frequently used when discussing upgrade and migration is *staging*. Staging can be used in two ways:

- Breaking the upgrade or migration into multiple steps that will be performed at different times so that, for example, the upgrade is completed over multiple maintenance windows.
- Using an interim set of servers on which to perform some of the upgrade and migration steps. The staging servers are frequently used to test portions of an upgrade or migration by using production data.

Figure 11-1 illustrates the process flow for a migration upgrade.

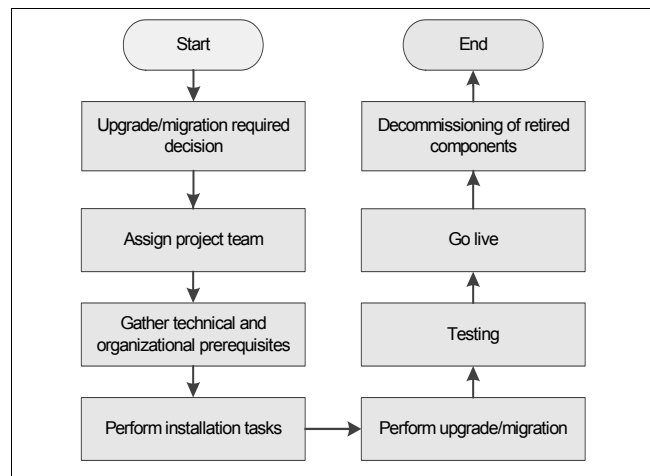


Figure 11-1 FileNet Content Manager upgrade migration flow

11.2 Planning for updates

The goal for any update is to make the change as transparent as possible to your users.

Updates usually must occur in a predefined maintenance window. The length of the maintenance window might vary between corporations, by type of change, and by the environment in which the update is being made. But in all cases, you want the update to complete in the prescribed maintenance window and with minimal impact to your customers. Planning and practice are the keys to every successful update project.

The planning effort includes these steps:

- ▶ Know your starting point.
- ▶ Know your desired endpoint.
- ▶ Understand what might be affected because of the desired endpoint.
- ▶ Detail the steps in the update, including the answers to these questions:
 - Who will perform each step?
 - How long will each step take?
 - What kinds of issues might occur?
 - What are the remedial steps for any potential issues?
 - What tests must be performed to validate that the update was successful?

Practicing the steps of the update provides the data for building the detailed plan and expanding the expertise of the team that is responsible for the update tasks. When rolling out an update, the learning and the practice happen on non-production systems. If the update includes moving to new hardware, that is, performing a migration, the practice can also occur on the new hardware with copies of production data.

11.2.1 Getting started

Before you perform any type of update in a FileNet Content Manager environment, ensure that you are prepared:

- ▶ Know the current levels of all the software in the environment, including the operating system, database, application server, and directory server so that you can validate that these software components are at a level that is compatible with the FileNet Content Manager software that you are planning to install. Your review needs to also include all third-party applications and other IBM components, such as IBM Content Collector, that interact with the P8 Content Manager environment.
- ▶ Obtain the latest fix pack readme file and release notes for the FileNet Content Manager components that are being updated.

Readme files can be downloaded from IBM Fix Central. The release notes are published in the P8 Information Center under **Troubleshooting and support** → **Release notes and what's new**.

- ▶ Review the FileNet P8 Fix Pack Compatibility Matrices to determine whether the package that you want to install is compatible with the other FileNet P8 components in the environment. Download the matrices from this website:

<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27014734>

- ▶ Review the IBM FileNet P8 Hardware and Software Requirements guide to determine whether the package you want to install is compatible with the underlying technologies in use in your environment. The guide can be downloaded from this website:

<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27013654>

If you are planning an upgrade or migration, also review the “Planning and Preparing” section of the P8 Information Center and take the time to complete the installation worksheet and to build a customized version of the upgrade instructions. The worksheet and instructions for building a customized upgrade guide can be accessed from the following page:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8toc.doc%2Fplanning.htm>

11.2.2 Practicing the update

Most clients have several FileNet Content Manager installations so that development and testing can occur in non-production environments. Updating these environments serially reduces the risk of introducing a change that has a negative impact on your production environment, and it also allows you to practice the update process.

The order in which you update the environments can be different depending on the update being installed. Table 11-1 illustrates the order in which you might perform an update, assuming you have the following five environments:

- ▶ Development
- ▶ User acceptance test
- ▶ Performance test
- ▶ Preproduction
- ▶ Production

Table 11-1 Update order

| Type of update | Software release | Mod release | Fix pack | Interim fix | Test fix |
|-----------------|------------------|-------------|----------|-------------|----------|
| Development | 1 | 1 | 1 | 3 | 3 |
| User Acceptance | 2 | 2 | 2 | 4 | 4 |
| Performance | 3 | 3 | 3 | 5 | 5 |
| Preproduction | 4 | 4 | 4 | 1 | 1 |
| Production | 5 | 5 | 5 | 2 | 2 |

Because interim fixes and test fixes are often installed to address an issue encountered in the production environment, it is important to get the change installed into the production environment as quickly as possible, but also as safely as possible. We advise that you install the change initially into an environment running the exact same software as the production environment, and then promote the change to production.

Practicing the update in this fashion has the following advantages:

- ▶ Refines the installation process that is needed at your site.
- ▶ Trains the resource staff who will ultimately perform the update in the production environment.
- ▶ Allows the project manager to develop a realistic schedule for the update.
- ▶ Identifies issues that might occur during the installation and any associated workarounds.
- ▶ Provides an opportunity to build an appropriate test suite to ensure that your applications work correctly after the update.

Recommendations: Test suites need to cover functional, stress, and performance testing.

11.2.3 Documenting the process

Using the customized upgrade documents generated from the P8 Information Center or the readme files provided with the software update packages, build an update document for your site that captures the following information:

- ▶ The process used
- ▶ Issues encountered and their workarounds
- ▶ Timelines
- ▶ Resources used; that is, the people involved in the update, their roles, and the time required

Recommendations: Ensure that at least two people know how to perform each task.

- ▶ Test process followed

The document will form the basis for future update projects.

11.3 Upgrading to a new software release

Upgrading to a new software release is a major project that usually takes several months because in addition to installing the new software, you must validate the following information:

- ▶ Existing applications work as expected; this task includes checking for deprecated APIs and reworking the code as needed.
- ▶ Performance and throughput are not adversely affected by the upgrade.

An upgrade is often the time to introduce new functionalities in the FileNet Content Manager components and custom applications, new hardware, and new underlying technologies.

Depending on the starting point and endpoint of the upgrade, you can choose how to install it:

- ▶ Stage the upgrade; that is, complete the upgrade over more than one maintenance window.
- ▶ Perform a “*big-bang*” upgrade; that is, complete the upgrade in a single maintenance window.

Whatever plan you choose to adopt, leave enough time in the plan for these activities:

- ▶ Backup and restore activities

At a minimum, take backups before you start the upgrade activity and after the upgrade completes.

- ▶ Testing

The test activities include functional, performance, and stress testing.

Tip: If you have a disaster recovery site, ensure that your plan includes breaking any synchronization links at appropriate points in the upgrade process, upgrading the software on the disaster recovery hardware, and re-establishing the synchronization links.

11.3.1 Staging the upgrade

To determine whether staging an upgrade will work for your environment, evaluate the following areas:

- ▶ What is the different mix and match of software components that are supported?

For example, if you move to the FileNet Content Manager 5.2 release, are the versions of FileNet Workplace XT, IBM Content Navigator, or IBM Enterprise Records currently in use in your environment compatible with the FileNet Content Manager 5.2 release? If they are, you can choose to upgrade to the new Content Platform Engine, but leave the other applications at their current levels.

- ▶ Are you moving to new hardware as part of the upgrade?

If so, you can install the new P8 Content Manager software on the new hardware and after the installation is validated, initially test the upgrade process by using copies of the production object stores, workflow systems, and file storage areas.

After the initial validation is complete, point the environment to the actual production databases and file storage areas and allow the update to complete.

- ▶ Are there clear delineations in the resources that are used by your applications?

If you are replacing all the hardware in your current production environment and there is a clear delineation in the resources used by your applications, you can choose to run with two production environments in parallel but on different release levels. In this model, you copy over the object stores, workflow systems, and file storage areas that are used by each application in stages.

For more information about taking this type of approach, see the following technote:

<http://www.ibm.com/support/docview.wss?uid=swg21428407>

11.3.2 Big-bang upgrade

With a big-bang upgrade, the goal is to update the existing environment to the latest release levels all in a single maintenance window. Often, this upgrade process is followed because the upgrade is taking place on the existing hardware.

To be successful with this type of upgrade in addition to practicing the upgrade to ensure that it can be completed in the required window, you must also have a roll-back plan. If the upgrade runs into an issue that cannot be resolved in the maintenance window, you can bring the old environment back online and limit the impact to your clients.

11.4 Migration best practices

If the upgrade involves moving to new hardware or changing the type of application server, database server, or LDAP server, the upgrade is called a *migration*. If you are moving to new hardware, the new hardware can be running the same or a different operating system as the existing hardware.

When you perform a migration, use the following best practices:

- ▶ Ensure that all servers are referenced by a domain name server (DNS) alias rather than an IP address.

This enables the change in server to be invisible to the clients. However, ensure that you consider the amount of time that is required for the change to replicate across the network.

- ▶ If moving databases from Oracle or SQL Server to DB2, work with IBM Lab Services on the data migration.

- ▶ If moving to a new LDAP server, ensure that all user and group references are updated correctly.

Consider testing the change by setting up a new P8 domain (at the same software level as the existing P8 domain) with the LDAP server and using the FileNet Deployment Manager to map users and groups. FileNet Deployment Manager will attempt to map the users and groups automatically, and flags any users or groups that do not appear to have a match in the destination environment.

Tips:

- ▶ If moving to a new type of LDAP server, consider engaging IBM Lab Services to help with the migration because they have expertise and special tools to help with this type of migration.
- ▶ If you perform the upgrade on new hardware, copy the Content Manager configuration file from the existing production system to the new hardware, and use it when running the upgrade of the Content Engine or Content Platform Engine.

11.5 Special considerations for upgrade

Two specific changes are introduced in the FileNet Content Manager 5.2 release that require special consideration.

Support for Legacy Content Search Engine is dropped

Content Search Services (CSS) was first introduced in the FileNet Content Manager 5.0 release as a replacement for the Legacy Content Search Engine. CSS provides similar functionality to Legacy Content Search Engine but is based on Lucene technology and uses a different query language.

Content indexes created by using Legacy Content Search Engine cannot be migrated to CSS indexes, instead the content must be reindexed.

Content Engine 5.1 supports running both Legacy Content Search Engine and CSS. Therefore, the indexes that are required by CSS can be generated as a background task. When the reindexing is complete, applications can start using the new indexes without affecting clients who are performing content-based retrievals.

Note: Applications that use content-based retrieval (CBR) must be updated to use the query language syntax that is compatible with CSS.

Content Platform Engine 5.2 supports CSS only. If content has not already been reindexed by using Content Platform Engine 5.2, clients will be unable to perform content-based retrievals until the reindexing is complete.

If there is a period of time where it is acceptable that content is not indexed, upgrade directly to Content Platform Engine 5.2. Otherwise, upgrade to Content Engine 5.1 before you upgrade to Content Platform Engine 5.2.

Content Engine and Process Engine servers are combined into a single engine called Content Platform Engine

In the FileNet Content Manager 5.2 release, the Content Engine and Process Engine capabilities are merged into a single engine called Content Platform Engine.

If you are performing an in-place upgrade, Content Engine and Process Engine must be upgraded together.

One side benefit of merging the two engines is that the two client installers (one for Content Engine and one for Process Engine) also are merged. Therefore, the number of installation programs that need to be run is further reduced.

11.5.1 Reference information

The following documents provide useful information about upgrades and migrations. Some of the documents reference specific release levels of the P8 Content Manager software, but they can often be generalized to work with all levels:

- ▶ White paper: *IBM FileNet P8 Platform Installation and Upgrade Best Practices*
<http://www.ibm.com/support/docview.wss?uid=swg27010422>
- ▶ Simultaneously Upgrading and Migrating Content Engine
<http://www.ibm.com/support/docview.wss?uid=swg21455046>
- ▶ Best Practices for FileNet P8 3.5.2 to P8 4.5.1.2 Upgrade with Replatforming (Windows to UNIX)
<http://www.ibm.com/support/docview.wss?uid=swg21428743>
- ▶ What adjustments can I make to FileNet Content Engine (CE) Automatic Upgrade?
<http://www.ibm.com/support/docview.wss?uid=swg21428407>

11.6 Conclusion

In this chapter, we discussed the update types and the best practices to follow to ensure a timely and successful update.



Troubleshooting

In this chapter, we discuss the methods that are used to troubleshoot IBM FileNet P8 Content Manager issues. P8 Content Manager implementations range from small departmental systems running one application using one or two servers to large enterprise systems running many applications on many servers.

We discuss the following topics:

- ▶ A typical P8 Content Manager system
- ▶ Different types of troubleshooting
- ▶ Creating customized best practice guides
- ▶ General troubleshooting
- ▶ Troubleshooting installation or upgrade
- ▶ Troubleshooting during application development
- ▶ Troubleshooting functional issues
- ▶ Troubleshooting production issues
- ▶ Troubleshooting performance issues
- ▶ Opening PMRs

12.1 A typical P8 Content Manager system

Before we focus on problem isolation, let us review what a typical P8 Content Manager system consists of and how it works. Figure 12-1 shows a basic P8 Content Manager system.

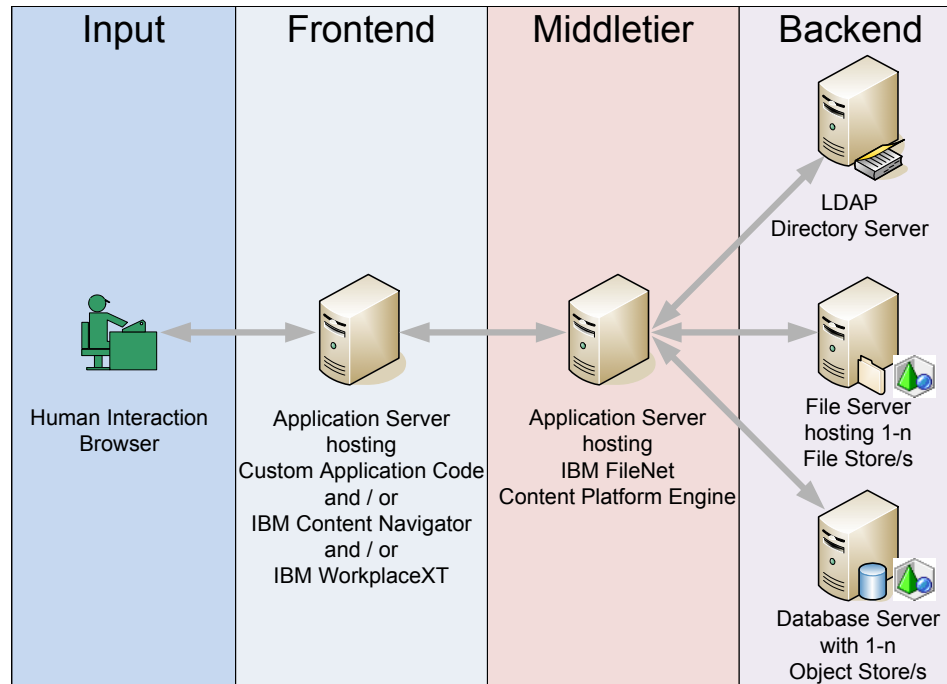


Figure 12-1 Pictorial view of a basic P8 Content Manager system

The basic P8 Content Manager system uses the following components:

- ▶ Client web browser or a desktop, thick client
- ▶ Front-end Java Platform, Enterprise Edition (Java EE) application server
Hosts the front-end application, such as IBM Content Navigator, FileNet Workplace XT, or a custom application.
- ▶ Middle-tier Java EE application server
Hosts the Content Platform Engine instances.
- ▶ Lightweight Directory Access Protocol (LDAP) directory service for security
- ▶ Storage system for document content
- ▶ Database for storing metadata and content

In the basic P8 Content Manager environment, the user points their browser or desktop to the chosen front-end application running on a Java EE application server:

1. The user receives a logon window and enters their user ID and password.
2. The credentials are validated against the directory server.
3. The Content Platform Engine caches user and group membership information.
4. The chosen front-end GUI then displays the appropriate artifacts to the user.

At this point, the user can view, change, or create new content. The benefit of this architecture is that as your user base or transaction activity grows, you can quickly and easily increase the resources allocated to the environment. You can scale vertically by adding more CPU and memory to your existing servers, or horizontally by adding more servers. This approach allows your P8 Content Manager system to grow to support hundreds of applications with thousands of concurrent users working on an enormous amount of content. The *N*-Tier Java EE architecture (server/client) enables P8 Content Manager to scale from small systems to large enterprise systems with minimal effort.

You can simplify the logon sequence by looking at it from a client/server perspective. The components used are essentially several client and several server components working together:

- ▶ The user's web browser is a client to the chosen front-end GUI.
- ▶ The chosen front-end GUI is a client to both the LDAP and Content Platform Engine.
- ▶ The Content Platform Engine is a client to the database and file storage areas.
- ▶ The Java EE application server is the server on which the chosen GUI application and Content Platform Engine run.

This perspective might be oversimplified, but as you approach a problem, think about it in client/server terms. Finding the failing client/server section enables you to quickly rule out what is working and focus on the component that is not working.

In this chapter, we look at common problems and different types of troubleshooting by breaking them down into a client/server approach.

12.2 Different types of troubleshooting

During the implementation and administration of a P8 Content Manager solution, situations arise that require troubleshooting. We list the main categories of troubleshooting:

- ▶ Troubleshooting installation and upgrade issues
Initial, basic problems that occur during the installation or upgrade of a P8 Content Manager system.
- ▶ Troubleshooting during application development or with how the code interacts with the P8 Content Manager configuration
Problems with the functionality of custom code.
- ▶ Troubleshooting performance issues
Response times are less than desired, or the system does not scale for the expected user and workload.
- ▶ Creating customized best practice guides
Problems that occur in the production environment and must be investigated and resolved with minimal impact to the clients.

12.3 Creating customized best practice guides

When discussing techniques and tools for troubleshooting and problem determination, typically most of the discussion is about what to do after a problem has been discovered. Ideally, however, the prudent system administrator or troubleshooter starts thinking about the job long before a problem occurs. Prepare the environment so that troubleshooting can be performed more quickly and effectively if and when problems occur.

One of the first and most important steps in being prepared to troubleshoot issues is to create a customized best practice guide that defines conventions for your environments. The guide preserves thoughts, knowledge, ideas, and information in one place. Start creating the guide during implementation, and continue to update it throughout the lifecycle of the environment. Capture information that you think you will never forget. Despite your best intentions, you will forget all the nuances around why a decision was made, or you might change roles and need to share your knowledge with new staff members.

Being consistent within an environment and across different environments reduces issues and makes troubleshooting easier.

We summarize many of the recommendations to cover in the customized best practice guides. But the list is neither definitive nor absolute. Each customer and every solution have their own special requirements. Use the list to get started and then continue to expand what you capture based on the specifics of your environment.

At a minimum, cover the following points in your best practice guide:

- ▶ A system architecture diagram for each environment, which is updated regularly. The diagram shows the physical layout and identifies what software is installed on each component.
- ▶ List software packages in use and the release level, including the patch level of each package, including any test fixes. The information covers items, such as the operating system, database, application server, Content Platform Engine, and IBM Content Navigator.
- ▶ Any dependencies in the environment, for example:
 - Startup and shutdown of the components.
 - Third-party applications that rely on P8 Content Manager or vice versa.
 - Data dependencies between the systems. This area is especially important if there are batch processes running for importing or exporting data. If a system stops during batch processing, you want to minimize any data loss.
- ▶ Guidelines for the following tasks:
 - Application development, including conventions, logging, multithreading, and designing for growth (for example, how to use multiple object stores, design for farming requirements, and if needed, design considerations for high availability)
 - Software installation, for example, user accounts, installation paths, fix packs
 - Application server configuration, including details about parameters that are using non-default values, or additional Java virtual machine (JVM) parameters
 - Database creation and configuration, including permissions, naming conventions, and the use of indexes
 - LDAP user and group naming convention creation
 - Monitoring configuration, for example, adding counters for use with IBM System Dashboard for Enterprise Content Management or IBM ECM System Monitor

- ▶ Content Platform Engine configuration:
 - Names of the databases used for the global configuration database (GCD) and object stores
 - LDAP server search filters
 - Accounts used for GCD and object store administration
 - Object store naming conventions for document classes, property templates and folder structures, storage policy conventions, and security requirements
 - Workflow system information, including security requirements, user queues, exposed fields, and the location of workflow definitions
- ▶ Housekeeping tasks:
 - Cleanup of log files: When backups need to occur, where old files are kept, and naming conventions
 - Database maintenance jobs, such as performance analysis, implementation of new indexes, and table space maintenance
- ▶ Regularly scheduled maintenance so that the environment remains current:
 - Plan for the installation of fix packs both for P8 Content Manager components and third-party software
 - Plan for upgrading to newer software releases
 - Plan for upgrading to new infrastructure levels, such as new operating systems, application servers, and database server software
 - Plan for upgrading to new hardware
- ▶ Log of problems and resolutions: Document issues that occurred and how they were resolved
- ▶ Add customized worksheets that can be used as reference documents when performing installations and upgrades

A customized best practice guide is a living document and must be updated all the time to track changes in the environment. Do not just write it once and then put it on the shelf. Keep it current, and make your organization aware of it.

A good reference resource is the following IBM developerWorks article that describes actions that you can take with your production environment now for a quicker and more effective way of troubleshooting:

http://www.ibm.com/developerworks/websphere/techjournal/0708_supauth/0708_supauth.html

12.4 General troubleshooting

The IBM FileNet P8 Information Center contains the documentation for the P8 product family. It is updated periodically, so always check for new information when you are troubleshooting:

http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.ecm.common.ic.doc/troubleshoot_main.htm

Recommendations: When an error occurs for the first time, always make it a high priority to analyze the issue, determine the root cause, and take the appropriate mitigating actions. Following this best practice minimizes the likelihood of the problem recurring and additional downtime.

Ideally, you will have automated system monitoring in place. Automated monitoring helps you quickly identify major component problems:

- ▶ IBM ECM System Monitor (see 10.4.3, “IBM ECM System Monitor” on page 334) allows you to quickly identify a major component failure, possibly before your first user calls. The software includes a knowledge base with potential problem solutions so corrective action can occur quickly.
- ▶ Dashboard (see “Dashboard” on page 328) can also help identify problem areas. You must manually check logs and functionality, because the Dashboard is meant primarily as a tool for gathering performance data.

Note: There is a dedicated dashboard provided for IBM Content Collector for Email.

- ▶ Third-party standard monitoring products, such as IBM Tivoli, HP OpenView, and Microsoft System Center Operations Manager (SCOM). These products support monitoring items, such as CPU, memory, network, and storage usage.

Automated monitoring tools can greatly reduce troubleshooting time, because they alert you to major component failures and problems, such as a disk or file system that is full.

Verify that each component is working as expected, as well as any infrastructure components, such as the database, application server, LDAP server, and network components.

Applications and tools that ship with P8 Content Manager, such as IBM Administration Console for Content Platform Engine (ACCE), IBM FileNet Enterprise Manager, IBM Content Navigator, and FileNet Workplace XT, are also helpful for problem determination. They can be used to perform similar actions, such as adding folders, creating documents, and viewing content. If none of the applications can perform a certain function, the problem you are trying to troubleshoot is likely caused by an infrastructure component, such as the network, or the Content Platform Engine server. However, if these applications function correctly but a custom application does not, the likely culprit is the custom application.

If an issue arises just after a component in the environment is updated, start the troubleshooting by ensuring all the component software levels are compatible with P8 Content Manager and with each other.

For P8 Content Manager compatibility requirements, see the following documents:

- ▶ Hardware and software requirements for the P8 suite of products:

<http://www.ibm.com/support/docview.wss?uid=swg27013654>

- ▶ FileNet P8 Fix Pack Compatibility Matrices:

<http://www.ibm.com/support/docview.wss?uid=swg27014734>

- ▶ MustGather: Read first for the Content Platform Engine:

<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg21308231>

The “MustGather” documents the information that the IBM Support team needs to start troubleshooting issues. Use this information as a starting point for your own troubleshooting efforts. If your individual troubleshooting efforts are unsuccessful, attach the information identified in the “MustGather” documents to the problem management record (PMR) or trouble ticket.

12.5 Troubleshooting the installation or upgrade

When implementing a new P8 Content Manager system, a need to troubleshoot issues can arise, for example:

- ▶ Underlying components might be configured in an unexpected fashion
- ▶ Network connectivity
- ▶ Lack of permissions to required resources
- ▶ Unexpected interaction with third-party applications or tools, for example, load balancers

Often, the issues arise because the guidance provided in the following documentation was not followed or validated prior to starting the installation:

- ▶ Planning sections of the P8 Information Center
- ▶ Third-party software-level requirements documented in the P8 Hardware and Software Requirements guide
- ▶ Information in the readme files or release notes

Avoiding common issues

To avoid the most common issues, perform the following tasks:

- ▶ Check the prerequisites of each software package and ensure that these requirements are met.
- ▶ Read the release notes of each software package carefully and ensure that you follow the notes.
- ▶ Check that the appropriate fix packs are installed in your environment.
- ▶ Install each software component one after the other in the following order and validate that each component is working as expected before moving on to the next component:

- Ports

Verify that the ports needed by the P8 Content Manager components are open (firewall) and not in use by other applications. A list of the required ports is available at this website:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.planprepare.doc/p8pap058.htm>

- Directory service

Typically, this directory service is the Corporate directory service, but you must ensure that the directory service is configured in a way that is compatible with the P8 Content Manager environment. Ensure that you have identified the users and groups to use during the installation process, as well as the user and group search filters that identify the users who can access the P8 Content Manager system.

For more information about directory service requirements, see the following link:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.security.doc%2Fp8psd000.htm>

- Operating system

Refer to the P8 Hardware and Software guide to determine the supported operating systems and operating system prerequisites.
- Database

Although new database servers can be installed specifically for use with the P8 Content Manager software, you can also use existing database servers if they meet the documented prerequisites. In addition, you need to create the databases and tablespaces required by the P8 components that will be installed. Some components, such as the Content Platform Engine, have a minimum requirement of one database for the GCD and one for an object store and a workflow system. However, components might require more databases depending on your specific use cases.
- Application server

Assuming an application server meets the documented prerequisite requirements and has adequate capacity, the P8 Content Manager components can be deployed to existing application servers. When deciding to use existing application servers or to install new application servers, consider the expected load on the system and the ability to accommodate more load than initially expected.
- Content Platform Engine
- FileNet Workplace XT

Required for managing workflow systems.
- IBM Content Navigator
- ▶ Check each of the following components to ensure that they are functioning correctly:
 - Network components:
 - Switches and load balancers are configured and working as expected.
 - The appropriate ports are open. For the ports used by P8 components, see the following link:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.planprepare.doc%2Fp8pap057.htm>
 - Operating system:
 - No error messages in the system logs after booting the system.
 - Network performance between components meets expectations.
 - Domain name server (DNS) lookup and reverse lookup between the servers works.

Recommendations: Avoid using IP addresses as references to P8 components. Instead, use DNS server names whenever possible to make it easier to replace or add servers to the environment.

- Database:
 - No error messages when starting the database server or any of the databases.
 - No error messages logged by any databases.
 - Databases can be accessed via telnet on the Java database. Connectivity (JDBC) port from other servers
- Directory service:
 - Directory service starts without an issue.
This is only a concern if you are not using the Corporate directory service.
 - No error messages logged by the directory service during any login attempts.
 - Directory server can be reached by telnet or by using an LDAP client on the defined port.
For information about the ports used for directory service operations, see the following link:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.planprepare.doc%2Fp8pap058.htm>
- Application server:
 - Application server starts without an issue, and no errors are reported by the application server software.
 - Application server administrative login works.
- ▶ Check integration between software components:
 - Content Platform Engine:
 - The installation and configuration steps completed successfully.
 - Ear or war startup is successful.
 - No errors are reported in the application server logs or the Content Platform Engine installation log.
 - Ping pages are reachable: Content Platform Engine System Health page (see Figure 12-2), Content Platform Engine Ping Page (see

Figure 12-3), and workflow system-related Ping Page (see Figure 12-4).

- Access the IBM Administration Console for Content Platform Engine (ACCE) and check the connections:
 - LDAP connection to the directory server
Configuration was done by the Configuration Manager during the Content Platform Engine configuration. If the application server was already configured for LDAP because of other hosted applications, ensure that the settings required by all applications are compatible.
 - All JDBC connections (both XA and non-XA) can connect to the expected databases.
Configuration was done by Configuration Manager during the Content Platform Engine configuration. This test ensures that the network routing is working correctly between the application server and any database server.
- Perform a quick check to validate the proper functionality of Content Platform Engine
Use ACCE to create an object store, add a folder, and then add a document.
- FileNet Workplace XT and IBM Content Navigator:
 - Deployment was successful and no errors were reported.
 - Ear or war startup is successful.
 - No errors are reported in the application server logs, the FileNet Workplace XT installation log, or the IBM Content Navigator installation log.
 - Login page displays.
 - Login is successful.
 - Manual configuration steps, such as defining security for accessing the FileNet Workplace XT administration tools, setting the site preferences, and check whether initiating the Process Configuration Console for initializing regions is successful.
 - Navigate to the folder and document that were created by using ACCE and open the document.
 - Add additional documents and then check that they can be opened by using ACCE.

If you follow this best practice and issues occur, start Root Cause Analysis (RCA) immediately. The localization of the root cause is much easier by using this

incremental approach as opposed to installing a whole system and verifying the system afterward.

For additional installation and upgrade troubleshooting information, see this website:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.install1.doc/p8pti000.htm>

Quick checks via server ping and health pages

Use the server ping and health pages to verify whether the Content Platform Engine, object stores, and workflow systems are online. The pages indicate whether a server is online and provide details about the software versions that are installed. IBM Support asks for this information if you open a problem management record (PMR).

To check whether the Content Platform Engine server is running, use its ping page by pointing your browser to this URL:

`http://<Content Platform Engine server>:<port>/FileNet/Engine`

This URL is an example:

`http://hqdemo1:9080/FileNet/Engine`

Your browser opens a window similar to Figure 12-2.

| Content Engine Startup Context (Ping Page) | |
|--|---|
| Key | Value |
| Product Name | P8 Content Platform Engine - 5.2.0 |
| Build Version | dap511.470 |
| Operating System | Windows Server 2008 R2 6.1 |
| JVM | java.vm.vendor IBM Corporation java.vm.name IBM J9 VM java.runtime.version pwa6460_26sr2ifx-20120419_02 (SR2) java.runtime.name Java(TM) SE Runtime Environment java.vm.version 2.6 java.vm.info JRE 1.6.0 Windows Server 2008 R2 amd64-64 Compressed References 20120322_106210 (J R26_Java626_SR2_20120322_1722_B106210 JIT - r11_20120322_22976 GC - R26_Java626 java.fullversion JRE 1.6.0 IBM J9 2.6 Windows Server 2008 R2 amd64-64 Compressed References 20120322_ R26_Java626_SR2_20120322_1722_B106210 JIT - r11_20120322_22976 GC - R26_Java626 |
| Start Time | Tue Mar 05 11:53:45 PST 2013 |
| Classpath | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\properties;C:\Program Files\IBM\WebSphere\AppSe \Program Files\IBM\WebSphere\AppServer\lib\bootstrap.jar;C:\Program Files\IBM\WebSphere\AppServer\lib\jsf-nls.j Files\IBM\WebSphere\AppServer\lib\urlprotocols.jar;C:\Program Files\IBM\WebSphere\AppServer\deploytool\itp/bat /itp/batch2.jar;C:\Program Files\IBM\WebSphere\AppServer\java\lib\tools.jar |
| Log File Location | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\FileNet\server1 |
| Local Host | cm-terranovavm21 |
| Available Processors | 1 |
| Working Directory | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01 |
| J2EEUtil | com.filenet.apimpl.util.J2EEUtilWS |
| Startup Message | P8 Content Platform Engine Startup: 5.2.0 dap511.470 Copyright IBM Corp. 2003, 2013 All rights reserved on serv |

Figure 12-2 Content Platform Engine ping page

After you confirm that the Content Platform Engine is running, use the Content Platform Engine System Health page to check the content-related items in the environment:

`http://<Content Platform Engine server>:<port>/P8CE/Health`

This URL is an example:

`http://hqdemo1:9080/P8CE/Health`

If your Content Platform Engine is running, you see a window similar to Figure 12-3.

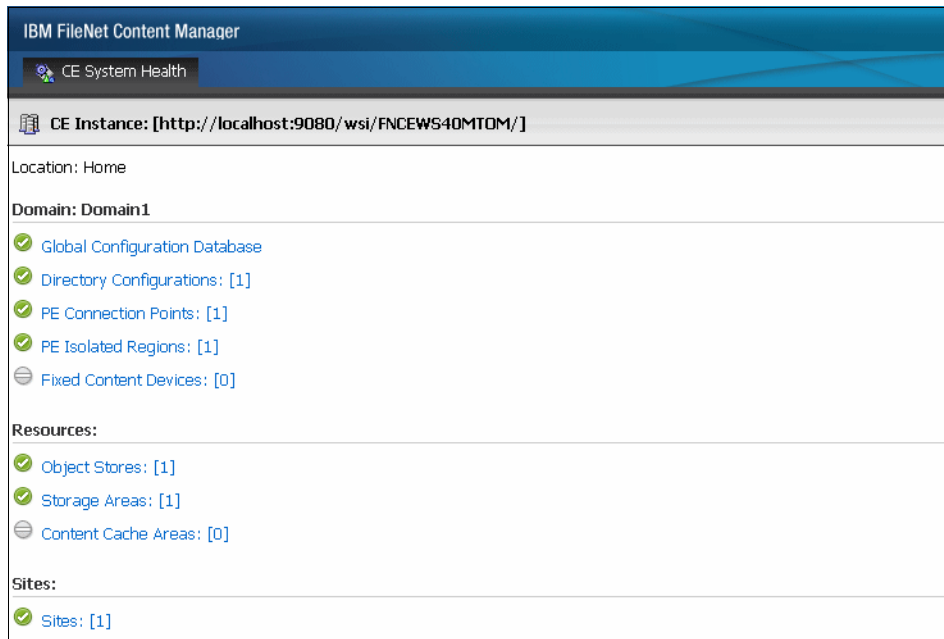


Figure 12-3 Content Platform Engine System Health page

To check on the workflow system-related components, use the following ping page:

`http://<Content Platform Engine server>:<port>/peengine/IOR/ping`

This URL is an example:

`http://hqdemo1:9080/peengine/IOR/ping`

The window is similar to Figure 12-4.

| Process Engine Server Information (Ping Page) | |
|---|--|
| Key | Value |
| Product Name | P8 - 5.2.0.0 |
| Build Version | dap511.470 pe.jar:dap511.470, 02/26/2013 15:33:20 |
| Operating System | Windows Server 2008 R2 6.1 amd64 |
| JVM | JRE 1.6.0 Windows Server 2008 R2 amd64-64 Compressed References 20120322_106210 (JIT enabled, AOT enabled R26_Java626_SR2_20120322_1722_B106210_CMPRSS J9CL - 20120322_106210 |
| Start Time | 2013/03/05 11:54:42.829-0800 |
| ClassPath | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\properties;C:\Program Files\IBM\WebSphere\AppServer\lib\bootstrap.jar;C:\Program Files\IBM\WebSphere\AppServer\lib\jstl.jar;C:\Program Files\IBM\WebSphere\AppServer\lib\urlprotocols.jar;C:\Program Files\IBM\WebSphere\AppServer\deploytool\ftp\batchboot.jar;C:\Program Files\IBM\WebSphere\AppServer\java\lib\tools.jar |
| Log File Location | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\FileNet\server1 |
| Local Host | cm-terranovavm21 |
| Data Directory | C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\FileNet\server1\pdata |
| Server Instance | cm-terranovavm21/server1 |
| Ping time | 2013/03/05 12:44:01.563-0800 |
| Database | PEPrimary POOL: PESecondary POOL: EventExporterTaskManager DELAYED until another 6969ms , Region=0 TaskManager DELAYED until another 110859ms , Region=0 PEHeartBeat POOL: EventExporter-DataCollector POOL: EventExporter-DataPublisher POOL: PESERVER:49733 @Tue Mar 05 12:10:51 PST 2013 PECIMsg @Tue Mar 05 12:10:51 PST 2013 |
| Active RPC Threads: | |

Figure 12-4 Workflow system-related ping page

Note: For farmed environments, verify both ways:

- ▶ Verify the Content Platform Engine access through the load balancer:
http://ce_load_balancer:<port>/FileNet/Engine
 The *ce_load_balancer* is the name of the load balancer, and *port* is the port used to access Content Platform Engine on the load balancer.
- ▶ Verify the Content Platform Engine access directly to each farmed Content Platform Engine instance, because the ping pages as well as the health page only display the status of the accessed instance and not the whole farm.

12.6 Troubleshooting during application development

Application development always needs a form of troubleshooting because of errors in the developed solution or a misuse of the P8 Content Manager APIs. The best practice is to define a developer style guide. This style guide covers the following topics:

- ▶ Which logging framework to use.
- ▶ How to configure the logs and traces.
- ▶ The log format.
- ▶ Identifying the details and information to log, which log file to use, and what gets logged at the informational, debug, and error levels.
- ▶ It also identifies bad practices, such as logging passwords or embedding passwords in the code.

With your custom application log or trace file, you have a starting point for troubleshooting. Also, look at any errors in the Content Platform Engine server logs. Make a note of the timestamps associated with the errors. If the error alone is not sufficient for solving the issue, the time stamp of the error message can also help you look for additional information in system and database logs. For this reason, all servers and, if possible, the clients, need to be time synchronized. Use a central Network Time Protocol (NTP) server and Coordinated Universal Time (UTC) for the server components for the time synchronization. Content Platform Engine indicates the time zone that is used in the `p8_server_error.log`. See Example 12-1.

Example 12-1 Time zone

```
P8 Content Platform Engine Version: 5.2.0 Build: dap511.470 on
rbacalzopc2
All times are local; the time zone is Pacific Standard Time(UTC -08:00)
VirtualServer: rbacalzopc2Node01 ServerInstance: server1
Date (UTC -08:00)      Thread  Sub  Category  Sev  Message
```

On the client side, the local time zone is used.

If custom client applications are accessing Content Platform Engine, the session ID of the client needs to also be written into the log and trace files. This information helps to connect the client and server log information, as well as make it easier to follow the application logic.

Additional information about exception and logging concepts is available at these websites:

- ▶ http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.ce.de.v.ce.doc/exception_concepts.htm
- ▶ http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.ce.de.v.ce.doc/logging_concepts.htm

12.7 Troubleshooting functional issues

The P8 Content Manager provides logging capabilities for issue tracking, error tracking, and troubleshooting, as well as for auditing and process tracking. Third-party software typically also provides logging capabilities. Next, we describe who, how, and why these logs are essential for troubleshooting functional issues.

Recommendations: When enabling trace logging for troubleshooting, only enable the subsystems that are necessary to diagnose the issue. Unconditionally enabling all levels of all subsystems negatively affects the performance and might rapidly fill up the file system.

12.7.1 Review the logs

During troubleshooting, check the relevant log files for errors with an unknown source.

The best way to troubleshoot an issue is to start at the bottom of the Content Platform Engine log files to look for error messages. Because there might be more than one type of error or multiple occurrences of the same message, keep working up through the file until you find the first error. If you find any error messages, review the information provided in the P8 Information Center under **Troubleshooting and support** → **IBM FileNet P8 messages**.

Tip: Use your favorite Internet search engine to look for additional information about any errors.

If an object Globally Unique Identifier (GUID) is provided in the error message, use ACCE or FileNet Enterprise Manager to determine which object is referenced by the GUID because this often provides additional clues about the problem.

If the information provided in the Content Platform Engine logs is not enough to identify and resolve the issue, the information points you to the next logical step in your troubleshooting efforts. Look at these files next:

- ▶ Other application logs
- ▶ Application server logs
- ▶ Database logs
- ▶ Operating system logs

Application logs

Java applications do not log error messages; they log exceptions. Java messages and exceptions are written to message log files. P8 Content Manager has two major engine components: IBM Content Navigator or FileNet Workplace XT as the front end, and the Content Platform Engine as the back end. Content Platform Engine writes four message logs:

- ▶ p8_server_error.log
- ▶ p8_server_trace.log
- ▶ pesvr_system.log
- ▶ pesvr_trace.log

For IBM WebSphere, by default, the files are in this directory:

AppServer\profiles\default\FileNet*<serverInstanceName>*

For BEA WebLogic, the default location is this directory:

\bea\user_projects\domains\mydomain\FileNet*<serverInstanceName>*

If the information in the logs does not provide enough information, it might be appropriate to enable trace logging for the relevant components of the Content Platform Engine.

Application server logs

Check the standard Java EE application server message logs for new, unknown error entries. The Java EE application server message logs for IBM WebSphere or Oracle WebLogic are in the following default directories:

- ▶ For IBM WebSphere, the SystemErr.log and SystemOut.log files are in this directory:

AppServer\profiles\default\logs*<serverInstanceName>*

- ▶ For Oracle WebLogic, the myserver.log file is in this directory:

\bea\user_projects\domains\mydomain*<myserver>*

where *<myserver>* is the web server name

If there are errors that seem related to the functional issues you are debugging, resolve the issues, recycle the environment, clear the logs, and then retest the P8 Content Manager application to see whether the problem is resolved.

Storage device logs

Network-attached storage (NAS) and storage area network (SAN) devices used for file and fixed storage staging areas typically have their own management consoles from which you can view system logs. Likewise, all the fixed content devices supported by Content Platform Engine have their own administration tools that can be used to diagnose the device and view system logs. Look for any warnings or errors on the storage device log, such as “out of space”, “maximum objects exceeded”, or “too many sessions”, that might correlate to the error received in the Content Platform Engine logs.

If there are errors that seem related to the functional issues you are debugging, resolve the issues, recycle the environment, clear the logs, and then retest the P8 Content Manager application to see whether the problem is resolved.

Database logs

If an object store cannot be accessed, response time is slow, or there are poorly performing queries, the next step is to ensure that the database is running as expected. Work with the database administrator to ensure that the databases are running as expected. Common issues that can occur include space allocation issues for the temp, database, and log files, poor query plans resulting in slow searches, and an insufficient number of available database sessions. Also, check for permission errors and that all the database prerequisites documented in the planning and prepare section of the P8 Information Center were implemented correctly:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.planprepare.doc%2Fp8ppi084.htm>

Performance-related issues can often be mitigated by reorganizing tables and reclaiming free space, adding indexes, or reworking searches so that they use indexes that exist.

System logs

Check the relevant system logs. If you find errors, ensure that the errors relate to the problem you are investigating. For example, there might be errors that relate to outdated virus files or an improperly configured mail server that are not related to the issue you are trying to debug.

If there are errors that seem related to the functional issues you are debugging, resolve the issues and then retest the P8 Content Manager application to see whether the problem is resolved.

Enable Trace Logging

To enable trace logging in Content Platform Engine, access the ACCE to check “Enable trace logging” on the Trace Subsystem tab. Then, select and configure the relevant subsystems.

When the trace control configuration is inherited from another object, the Trace Flags list box is disabled.

Table 12-1 lists all the subsystems, and the associated abbreviation that appears in the trace log file along with a short description of what is traced.

Table 12-1 List of traceable subsystems

| Property | Abbreviation | Logs information relative to this area |
|-------------------------|--------------|--|
| API | API | Content Java API operations. Logging is supported only for the Content Java API, not the Content .NET API. |
| Asynchronous Processing | ASYN | The processing portion of an asynchronous event, including document classification and security propagation. |
| Audit Disposition | AUDT | Requests to audit information in the object store. |
| CBR | CBR | Content-based retrieval (CBR), including indexing, searching, and so on. |
| CFS Daemon | CFSD | Content Federation Services (CFS) for Image Services import agent. |
| CFS Import Agent | CFSI | Content Federation Services for IBM Content Integrator import agent. |
| Code Module | CMOD | Code module functionality. Code modules are a special subclass of document class that contains one or more Java components. |
| Content Cache | CCHE | Content caching operations that cache document content in the file system on the local server. |
| Content Storage | CSTG | Content storage operations. |
| Database | DB | Database operations. This option can generate a large amount of information, so only have it enabled when specifically reproducing a database-related issue. |

| Property | Abbreviation | Logs information relative to this area |
|------------------------|---------------------|--|
| EJB | EJB | The Enterprise JavaBeans (EJB) transport layer, which is the component architecture for the development and deployment of object-oriented, distributed, enterprise-level applications. |
| Engine | ENG | The Content Platform Engine server core. |
| Error | ERR | Error handling operations. |
| Events | EVNT | General event processing. |
| Fixed Content Provider | FCPV | Any fixed content providers. |
| GCD | GCD | The global configuration database (GCD) and its operations. |
| Handler | HDLR | Custom server handler code. |
| Metadata | MCHE | Metadata cache operations. |
| Publish | PUBL | Publishing operations. |
| Replication Subsystem | REPL | Replication subsystem operations. |
| Search | SRCH | Search and query operations. |
| Security | SEC | The client and server components used to authenticate (layered over the authentication provider) and authorize user access to Content Platform Engine objects. |
| SSI | SSI | Integration with the Single-document Storage Interface (SSI), which is an interface between the Content Platform Engine and FileNet Image Services. |
| Sweep | SWP | Sweep operations. |
| Thumbnail Generation | THMG | Thumbnail generation operations. |
| WSI | WSI | The Web Services Interface (WSI) transport layer to Content Platform Engine. |

The following trace levels can be set for each subsystem:

▶ SUMMARY

Enables minimal high level logging by providing summary information for all operations. This setting does not significantly affect system performance.

▶ MODERATE

Enables more detailed high level logging than the SUMMARY option for all operations (includes all SUMMARY level information). This setting has some impact on system performance.

▶ DETAIL

Enables the most detailed logging by providing detailed information for all operations and is primarily used to aid in debugging issues (includes all SUMMARY and MODERATE level information). This setting significantly affects system performance.

▶ TIMER

Provides the duration (in milliseconds) that Content Platform Engine requires to complete an operation, such as uploading a file. This setting does not significantly affect system performance.

Depending on the issue reported by the clients, select which subsystems to trace and an appropriate trace level. There is no need to restart the system to start or stop gathering trace logs.

Recommendations: Disable the trace logging as soon as possible, because it can fill up the file system rapidly because of the amount of generated data.

12.7.2 Review additional sources for information about issues

If reviewing the logs does not lead you to a problem resolution, perform the following actions:

- ▶ Look at the portlets “Featured troubleshooting links”, “Flashes and alerts”, and “Problem resolutions” on the IBM Support home page:

http://www.ibm.com/support/entry/portal/troubleshooting/software/enterprise_content_management/filenet_product_family/filenet_content_manger#

- ▶ Check the public P8 forums on the Internet.
- ▶ Use your favorite search engine to look for your issue and a possible solution.

- ▶ Post a question on IBM developerWorks:

<http://www.ibm.com/developerworks/forums/forum.jspa?forumID=1165&start=0>

12.8 Troubleshooting production issues

When discussing production issues and how to troubleshoot them, remember that this environment is the production environment and users are affected by all changes. As long as the production system is running and only some users are affected by an issue, keep calm.

Recommendations: Restarting the servers and applications that are having an issue might get the system back to “normal” operating conditions but usually also camouflages the root cause of the issue. And in a worst case scenario, losing information about the root cause of the problem can worsen the overall situation.

Finding an appropriate balance between error analysis and limiting downtime or slow response times is critical, so perform these steps:

1. Read (and store) log files and error messages.
2. Consider your options before reacting.
3. Execute your remedial steps.
4. Document your activities for future reference.

Try to reproduce the issue and behavior of the production system in a test environment¹. Then, analyze and try to fix the problem. If you are unable to fix the problem on your own, contact the IBM Support team and ask for help. When you receive a fix (software or a configuration change), install the fix into an integration or test environment. Perform some general regression tests in addition to validating that the main issue is fixed before moving the fix into the production environment. Depending on the severity of the issue, you might also choose to move the fix into all your other environments before moving it into production, in much the same way as you handle a new application or an enhancement to an existing application.

Follow these steps before you put the change into the production environment:

1. Back up any log files.
2. Clear the log files so any new errors are easy to detect.

¹ This action presupposes a similar reference system is available with similar physical and software characteristics as the production environment.

3. Enhance the log and trace level in the production environment.
Be conservative when changing logging levels in production because more detailed logging causes an increase in the system load and therefore a decrease in performance.
4. Note all original and changed values in an administration log file for the production environment.
This makes it easier both to revert the system back to its initial state and to update other environments to match the new state of the production environment.
5. Validate that the fix is working as expected.
6. Reduce the level of logging back to normal levels using the reference information generated in step 4.

12.9 Troubleshooting performance issues

Performance problems in P8 Content Manager can come from a wide range of causes. Some issues can be eliminated by performing load and stress tests before rolling out new applications, new software releases, and incremental updates to production. Other issues can occur intermittently and might be caused by unusual situations, such as network bandwidth issues or transferring large files.

During system load and performance testing, it is likely that the required throughput will not be achieved without removing one or more bottlenecks. Performance issues can also occur after applications go live during normal, production operations of a P8 Content Manager solution. The following examples show when issues might occur:

- ▶ New software releases are deployed into production and some function response times increase, and the existing load and performance tests do not cover these particular functions.
- ▶ The user load increases to the point where the current system configuration can no longer perform as required.
- ▶ The data load increases to the point where the current system configuration can no longer perform as required.
- ▶ Security patches applied to the operating system have a negative impact on system performance.
- ▶ New users and groups are created in the directory server, or a change is made in the directory server configuration in such a way that the change negatively affects P8 Content Manager performance.

- ▶ Storage device configuration changes or hardware and software failures occur with the storage device.

Troubleshooting performance issues can cross organizational boundaries and require the cooperation of database administrators, directory server administrators, application server administrators, IT staff, application developers, and the P8 Content Manager administrator.

12.9.1 Performance tuning guide

A first entry point for tuning the system for optimal performance is the performance tuning information provided in the P8 Information Center:

<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.performance.doc/p8ppt000.htm>

The web page provides several links to topics with detailed information for tuning:

- ▶ Operating systems
- ▶ Databases
- ▶ Application servers
- ▶ Directory servers
- ▶ Content Platform Engine
- ▶ FileNet Workplace XT and IBM Content Navigator

The performance tuning topics in the FileNet P8 information center do not include suggested parameter settings, because the correct values are dependent on your environment and application.

12.9.2 Gathering performance data

The measurement of key performance indicator (KPI) values and KPIs of the system during load and stress tests and during normal working hours is important. This data generates the boundary and baseline performance information. The gathered information is used to define monitoring software thresholds and for discovering and troubleshooting performance issues. KPIs reflect how fast the system reacts during a defined workload and the high watermarks that cause the system response times to increase dramatically.

Running load and stress tests requires significant work. In addition to setting up an environment that mimics production, you need to perform these tasks:

- ▶ Populate the environment with data that is representative of production data in terms of type and volume of data.

- ▶ Define and build tests that mimic realistic production workloads.
The tests need to mimic both the type of work clients will perform, the number of clients that will use the system at any one time, and the length of time clients will access the system. The term “*client*” includes users who access the system manually as well as any batch processes.
- ▶ Have a way to refresh the environment so that you can rerun tests to validate that changes affect system performance in a positive manner.

Recommendations: Automate as much of the system setup, performance testing, and data gathering as possible.

The following tools can help you collect and analyze performance data:

- ▶ IBM System Dashboard for ECM
The Dashboard collects the following information from the P8 Content Manager environment:
 - Environmental information
 - Central processing unit counters
 - Disk counters
 - Network inbound/outbound counters
 - User counters and response times of operations
- ▶ IBM Thread and Monitor Dump Analyzer for Java
JVM thread dumps can be used to analyze the detailed behavior of the application server and the hosted JVMs. The tool is available at this website:
<http://www.ibm.com/developerworks/mydeveloperworks/groups/service/html/communityview?communityUid=2245aa39-fa5c-4475-b891-14c205f7333c>
This tool identifies the following information:
 - Hangs
 - Deadlocks
 - Resource contention
 - Bottlenecks in Java threads
- ▶ Windows Reliability and Performance Monitor
The Windows Reliability and Performance Monitor is a Microsoft tool. This tool provides a quick overview of the processes that use a large amount of CPU, memory, disk, and network resources.
If you need a longer history of how processes and applications consume server resources, use the Microsoft Data Collector Sets. Start by using the preset templates, then modify the templates as needed and save them as your own templates and presets.

More details are provided by Microsoft Technet:

[http://technet.microsoft.com/en-us/library/cc749154\(v=ws.10\).aspx](http://technet.microsoft.com/en-us/library/cc749154(v=ws.10).aspx)

► Nmon

This free tool provides a huge amount of information all on one window. Even though IBM does not support the tool and you must use it at your own risk, it provides a wealth of performance statistics. The tool is available at this website:

http://www.ibm.com/developerworks/aix/library/au-analyze_aix/

The nmon tool is designed for AIX and Linux performance specialists to use for monitoring and analyzing the following performance data:

- CPU utilization
- Memory use
- Kernel statistics and run queue information
- Disk I/O rates, transfers, and read/write ratios
- Free space on file systems
- Disk adapters
- Network I/O rates, transfers, and read/write ratios
- Paging space and paging rates
- CPU and AIX specification
- Top processors
- IBM HTTP web cache
- User-defined disk groups
- Machine details and resources
- Asynchronous I/O (AIX only)
- Workload Manager (WLM) (AIX only)
- IBM TotalStorage Enterprise Storage Server® (ESS) disks (AIX only)
- Network File System (NFS)
- Dynamic logical partition (LPAR) (DLPAR) changes (only IBM pSeries® p5 and IBM OpenPower® for either AIX or Linux)

12.9.3 Slow logon

A typical performance issue from all web-based applications is the client user complaining about how long it takes to log on (slow logon). The steps for resolving this type of issue vary depending on whether the issue is on a new or recently upgraded installation, or whether it is occurring on a system that has been running for a while without issues.

Note: After starting or restarting the application server or redeploying the Content Platform Engine ear or war file, the first logon will take longer than subsequent logons because the cache has to be refreshed. You can minimize this behavior:

- ▶ Use the precompile options of the application server, although this will increase the time it takes to restart the application server.
- ▶ Use a script whenever the application server is restarted to walk through all commonly used application functions. The script must be run on all application server nodes.

Issue following installation or upgrade

For a new installation or a recently upgraded installation, check the following configurations:

- ▶ The LDAP configuration of the application server:
 - Connection timeout setting for LDAP queries.
 - Size of the LDAP connection pool is sufficient.
- ▶ The LDAP configuration in ACCE:
 - Validate which directory services are configured.
 - Check whether the DNS entry is resolved to a single, dedicated server or is an alias to a number of directory servers.

Ideally, the Content Platform Engine will always access the directory server that is closest physically to the Content Platform Engine server. The greater the distance between the servers, the slower the logon.
 - Size of the Maximum User Token Cache Entries

Ensure that the size is adequate compared to the number of active system users, so that if users frequently log in and out, their information remains cached and does not have to be reacquired from the directory server every time.
 - Time of User Token Cache Entry TTL

Increasing the value of this parameter can enhance system performance, but the trade-off is LDAP changes take longer to take effect within the P8 Content Manager environment.

- ▶ Examine the user, group membership, and organizational unit (OU) configurations in the LDAP server:
 - If there are nested group memberships, check the nesting depth.

The more nested groups and the deeper the nesting depth, the longer it takes the Content Platform Engine to identify all the groups to which a user belongs.
 - Limit the tree that has to be traversed to find users and groups as much as possible.
- ▶ Check the network
 - Number of network hops and the latency of the network between the application server and the LDAP server.

Work with the network administrators to implement special routing for LDAP requests.
 - Bandwidth between the application server and the LDAP server.

The performance can be improved by introducing quality of service (QoS) for the LDAP requests if the bandwidth is limited.

Production issues

In addition to the basic performance issues during the initial implementation of a solution, there is always the possibility of running into performance issues during production.

Implement regression test cases to validate your application after upgrading to new software levels, fixing an issue, or introducing new features. One of the regression tests needs to be a performance baseline test in which a known user logs on to the system. If your clients are reporting slow logons, one of the first troubleshooting steps is to run this baseline test. The results of the test help identify whether there is an issue common to all users or whether the slow logon affects only a subset of your clients.

If all users are affected, review the common infrastructure elements to determine whether a failure occurred.

If the problem is limited to a subset of users, follow these steps:

- ▶ Ask the user if the logon was quicker in the past. If yes, ask when they first noticed the slowdown.
- ▶ Examine the user's group membership to determine whether anything changed recently:
 - Does this user belong to significantly more groups than other users?
 - If there are nested group memberships, determine the nesting depth.

- ▶ Determine which DNS server is processing the user's authentication request, and whether authenticating via a different server has an impact on the logon.

12.9.4 Slow searches

Content Platform Engine supports customizable searches. Searches can be constructed with various operators and combinations of search conditions, joins, subqueries, and ordering, and can include the paging of results.

Because of this powerful and customizable search capability, and because unnecessary indexes can degrade performance when objects are added or updated, a single set of database indexes to support high performance searches for all application designs does not exist.

Therefore, you must analyze the searches run in your environment in order to create the right and minimal set of database indexes to ensure high performance.

IBM provides a detailed technote to deal with this challenge:

<https://www.ibm.com/support/docview.wss?uid=swg21502886>

This list summarizes the best practices when you create searches:

- ▶ Use a wildcard search only at the end of a string search value and use `Starts With` for a `LIKE` query.
- ▶ Create Oracle function-based or DB2 generated-column indexes for string comparisons that are not case sensitive.
- ▶ Join only one or two tables at a time in a query.
- ▶ Avoid unnecessary `Order By` clauses.
- ▶ Use the `Intersects` operator for multiple `List property OR` conditions.
- ▶ Use continuable searches for queries that are run from a graphical user interface.

Continuable searches reduce the time database locks are held, and allow result sets to be displayed more quickly to clients when compared to non-continuable searches. However, the total time taken for a non-continuable search is shorter than for continuable search.

In a non-continuable search, all rows are returned and the database can hold row locks during the query, which can result in timeouts or database contention.

- ▶ Verify that searches are using the expected indexes and are using the best possible query plan.

- ▶ Ensure DBAs regularly monitor database statistics and reorganize frequently updated tables.

12.9.5 Storage performance issues

When you run into storage performance issues, use the IBM System Dashboard for Enterprise Content Management to examine the file and fixed storage performance counters. There are many counters related to storage performance, and these counters can indicate a bottleneck reading or writing to a storage device, or a high failure count related to a storage device.

A good place to start is the Content Upload and Content Download server-based counters. If the performance issue cannot be resolved and IBM Support is needed, collect IBM System Dashboard data during the time that the issue is seen. The interval that the counters are collected needs to be in the 10 - 30 second range. Be sure to supply IBM Support with a system monitor archive file (Archive history option), since an archive file can be used to examine all the detailed information collected by the system monitor. In addition to an archive file, supply IBM Support with information about the system architecture. Include detailed information about the storage devices and usage patterns, for example, storage devices shared by applications other than the Content Platform Engine.

12.9.6 Tuning sweep jobs

For the best performance of policy and job sweeps, create a covering index on the base table that is being swept. In addition, analyze the searches and the query plans for the sweep jobs and add indexes as necessary.

Use the sweep preview mode to determine the rate at which the sweep will traverse the objects, without actually performing the sweep actions on the objects. Follow these steps to run a sweep in preview mode:

1. Define the job or the policies exactly as they will be run in production, including the proper classes, subclasses, and filter expressions.
2. Set the Sweep Mode to Preview Counters Only mode.
3. Let the sweep run and use the IBM System Dashboard to monitor the progress.
4. Review the sweep framework summary tracing to see the overall sweep rate after the sweep finishes.

12.10 Opening PMRs

If you are having difficulty correcting a problem yourself, open a Problem Management Record (PMR) with IBM software support. You can open PMRs by calling IBM Support directly or via the IBM software support portal. To open a PMR, you will need your IBM Customer Number (ICN).

12.10.1 The IBM software support portal

The IBM software support portal details all facets of IBM software support. It provides self-help information, as well as instructions for creating PMRs and escalation procedures. You can obtain the specialized IBM software support portal for FileNet at the following link:

http://www.ibm.com/support/entry/portal/Overview/Software/Enterprise_Content_Management/FileNet_Product_Family/FileNet_Content_Manager

12.10.2 Open a PMR by calling IBM

IBM software support has local call-in numbers for most countries. Local numbers are at this website:

http://www.ibm.com/support/entry/portal/Overview/Software/Enterprise_Content_Management/FileNet_Product_Family/FileNet_Content_Manager

Select **Support & downloads** → **Service requests and PMRs** and find the link. Then, click **Contact support** → **Directory of worldwide contacts**.

Or use this hot link:

<http://www.ibm.com/planetwide>

In the US, call IBM software support at 1-800-IBM-SERV (1-800-426-7378) and select option 2. Ask the dispatcher to open a PMR on your behalf and to connect you with an IBM Support specialist.

12.10.3 Open a PMR via the web

You can open a PMR via the web through the Electronic Service Request (ESR) tool, which is available at this website:

<http://www.ibm.com>

Follow these steps to open a PMR:

1. Select **Support & downloads** → **Service requests and PMRs** → **Go to IBM Service Request**.
2. Log in with your IBM ID.
3. Select **Open a new service request** → **Enter your keyword(s)**. For example, enter Content Platform Engine.
4. Select the product that you are using. Enter the relevant data. Select **Continue**.

Or, use this hot link:

<http://www.ibm.com/software/support/probsub.html>

To submit PMRs via the website, your company's Site Technical Contact must authorize you to submit PMRs electronically to IBM.

Important: If your production system is down, call IBM Support directly and open a severity 1 PMR.

12.10.4 Necessary items when contacting IBM software support

When calling or submitting a problem to IBM software support, have the following information ready. The more information you can supply, the quicker the IBM Support team can start resolving your issue. Provide the following information:

- ▶ Your IBM customer number
- ▶ Your company name
- ▶ Your contact name:
 - Preferred means of contact (voice or email)
 - Telephone number where you can be reached
- ▶ Machine type and model number:
 - Related product and version information
 - Related operating system and database information
- ▶ Prepare the "MustGather" information manually or by using the IBM Support Assistant (ISA) Workbench.

- ▶ Detailed description of the issue.

Being able to articulate the problem and symptoms before contacting software support expedites the problem solving process. It is extremely important that you are as specific as possible in explaining a problem or question to the IBM software specialists. Our IBM Support specialists want to be sure that they provide you with exactly the right solution; therefore, the better they understand your specific problem scenario, the better they are able to resolve it.

Gathering background information

To effectively and efficiently solve a problem, the software specialist needs to have all of the relevant information about the problem. Being able to answer the following questions will help in the efforts to resolve your software problem:

- ▶ Has the problem happened before, or is this a new, isolated problem?
- ▶ What steps led to the failure?
- ▶ Can the problem be re-created? If so, what steps are required?
- ▶ Have any changes been made to the system, such as hardware, network, or software updates?
- ▶ Are any messages or other diagnostic information produced when the error occurs? Attach the information to the PMR and include a screen capture if appropriate. Identifying the time at which the error occurred is helpful.

It is often helpful to have a printout of the message numbers of any messages received when you call IBM Support.

Define your technical question in specific terms and provide the version and release level of the products in question.

Gather relevant diagnostic information, if possible. It is often necessary that the software support specialists analyze specific diagnostic information, such as storage dumps and traces, in order to resolve your problem. Gathering this information is often the most critical step in resolving your problem.

On more difficult problems, you might also need the following items:

- ▶ Application architecture diagram that details how all application components are designed to work
- ▶ Network topology diagram, including servers, routers, firewalls, and network load balancers
- ▶ If your problem is performance-related, performance archive files

Determining the business impact

You need to assign a severity level to the problem when you report it, so you need to understand the business impact of the problem that you are reporting. A description of the severity levels is in Table 12-2.

Table 12-2 Problem severity descriptions and examples

| Severity level | Further definitions | Examples |
|----------------|---|--|
| Severity 1 | Critical situation/system down: Business critical software component is inoperable. As a rule, it applies to the production environment. | The P8 Content Manager system is down and affecting all users. |
| Severity 2 | Severe impact: A software component is severely restricted in its use, causing significant business impact. | The P8 Content Manager system cannot be accessed by one department. Other users are able to access the system. |
| Severity 3 | Moderate impact: A non-critical software component is malfunctioning, causing moderate business impact. | A client cannot connect to a server. |
| Severity 4 | Minimal impact: A non-critical software component is malfunctioning, causing minimal impact, or a non-technical request is being made. | Documentation is incorrect. Additional documentation requested. |

When speaking with the software support specialist, also mention the following items if they apply to your situation:

- ▶ You are under business deadline pressure.
- ▶ Your availability, or when you will be able to work with IBM software support.
- ▶ You can be reached at more than one phone number.
- ▶ You can designate a knowledgeable alternate contact with whom the IBM Support representative can speak.
- ▶ You have other open problems (PMRs) with IBM about this service request.
- ▶ You are participating in an early adoption program.
- ▶ You have researched this situation prior to calling IBM and have detailed information or documentation to provide for the problem.

12.10.5 IBM Support Assistant (ISA) Workbench

The IBM Support Assistant Workbench provides access to several different serviceability tools, which can assist you in many areas of problem diagnosis, such as Java troubleshooting, product configuration analysis, and log analysis. IBM Support Assistant can be customized for over 350 products and over 20 tools.

If you have permission to use this software tool, install and use it for general troubleshooting or for gathering all relevant data for opening a PMR.

IBM Support Assistant Workbench is available at this website:

<http://www.ibm.com/software/support/isa/#isawb>

If the entire tool is too large or you are not allowed to install this type of tool in the production environment, consider using the IBM Support Assistant Lite Data Collector version instead. IBM Support Assistant Lite Data Collector is smaller than the IBM Support Assistant Workbench, because it is a simple utility that collects the specific data that is needed for opening a PMR.

IBM Support Assistant Lite Data Collector is available at this website:

<http://www.ibm.com/software/support/isa/#isadc>

You can also install IBM Support Assistant as a team server that you and your team can access via a browser. An Overview tab provides all the relevant system data. The Symptom tab displays all problems found and ranks them. A Global Knowledge Base Matches tab displays potential solutions for the symptoms identified during the analysis of your case files. In this view, you can quickly identify technical documents, APARs, and fix packs that address the symptoms identified during the automated analysis.

12.10.6 Type of fixes that might be provided

There are three types of fixes that might be provided by the IBM Support Team release:

- ▶ Fix pack
- ▶ Interim fix
- ▶ Test fix

An *Authorized Program Analysis Report* (APAR) is associated with each fix that is contained in a support package.

An APAR is generated as a result of the following actions:

1. A customer reports an issue via a PMR.
2. The Level 2 (L2) support team investigates the issue and determines that they need additional assistance, and open a Change Request Management (CRM).
3. The Level 3 (L3) support team assists with the investigation of the issue.
If a software fix is required, an APAR is opened so that information about the issue can be available to other customers.

All fixes are made available via IBM Support Fix Central:

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

Before you install a fix, review the readme file that is provided with the package. Review the appropriate FileNet P8 fix pack compatibility matrices that can be downloaded from this website:

<http://www.ibm.com/support/docview.wss?uid=swg27014734>

Fix pack

A *fix pack* provides a roll-up of APAR resolutions that were previously provided as interim fixes, test fixes, or in a previous fix pack, as well as fixes not previously released.

Interim fix

An *interim fix* provides the resolution to a few APARs, usually one, that are likely to be needed by multiple customers.

Test fix

A *test fix* provides the resolution to a few APARS, usually one, that are required by a specific customer. Test fixes are password protected.

12.10.7 Rolling up fixes

If you received a test fix or an interim fix from IBM Support, before you install a new fix pack or a later release of a P8 Content Manager component, ensure that the fixes you need are included in the software. The fixes and software releases occur at various times so you cannot assume that your specific issue is resolved in the “newer” software release package.

If you are in any doubt, contact the IBM Support team to ensure that your fixes are included in the software release that you want to install.

12.11 Conclusion

In this chapter, we discussed the basics of troubleshooting P8 Content Manager and associated third-party software.

In the next chapter, we provide an overview of building software solutions that use P8 Content Manager.



IBM FileNet Content Manager solutions

In this chapter, we describe product feature capabilities and solution components to aid designers in preparing Enterprise Content Management (ECM) solutions. The material describes available options for the input, management, storage, process, and presentation phases of ECM solutions.

We introduce these ECM design aids:

- ▶ Solution building blocks:
 - Foundation components
 - Content ingestion tools
 - Process management
 - Presentation features
- ▶ Sample use cases of solution building blocks

13.1 Solution building blocks

At a fundamental level, any ECM solution is composed of five major solution components:

- ▶ Foundation components
- ▶ Content ingestion tools
- ▶ Process management
- ▶ Presentation features

All ECM solutions are essentially a construction of information input, storage, information processing, and presentation and delivery. Figure 13-1 illustrates the major ECM solution components.

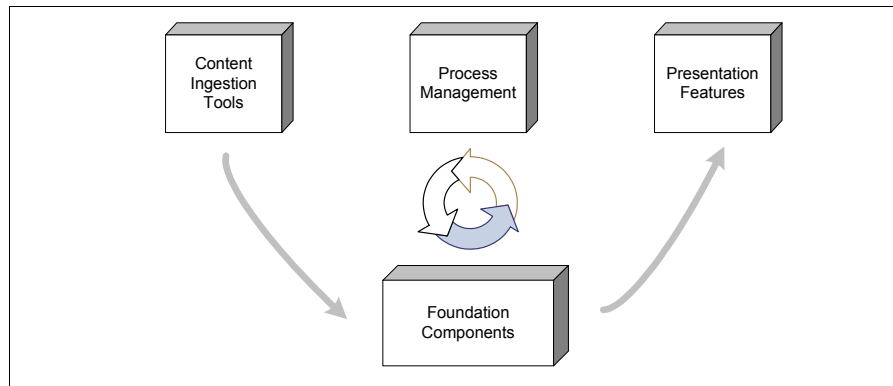


Figure 13-1 Major ECM solution components

These major components form the building blocks of ECM solution design. *Solution building blocks* are the features that ECM solution designers can specify and combine to build out each of the components of an ECM solution: content ingestion, content management, process, and presentation. The IBM FileNet suite of products contains applications and tools that offer designers a wide range of features and functions for the design of each of the major components of an ECM implementation.

Figure 13-2 on page 429 shows several of the IBM tools that are available to ECM designers and the place for these blocks within the four major design phases of an ECM solution.

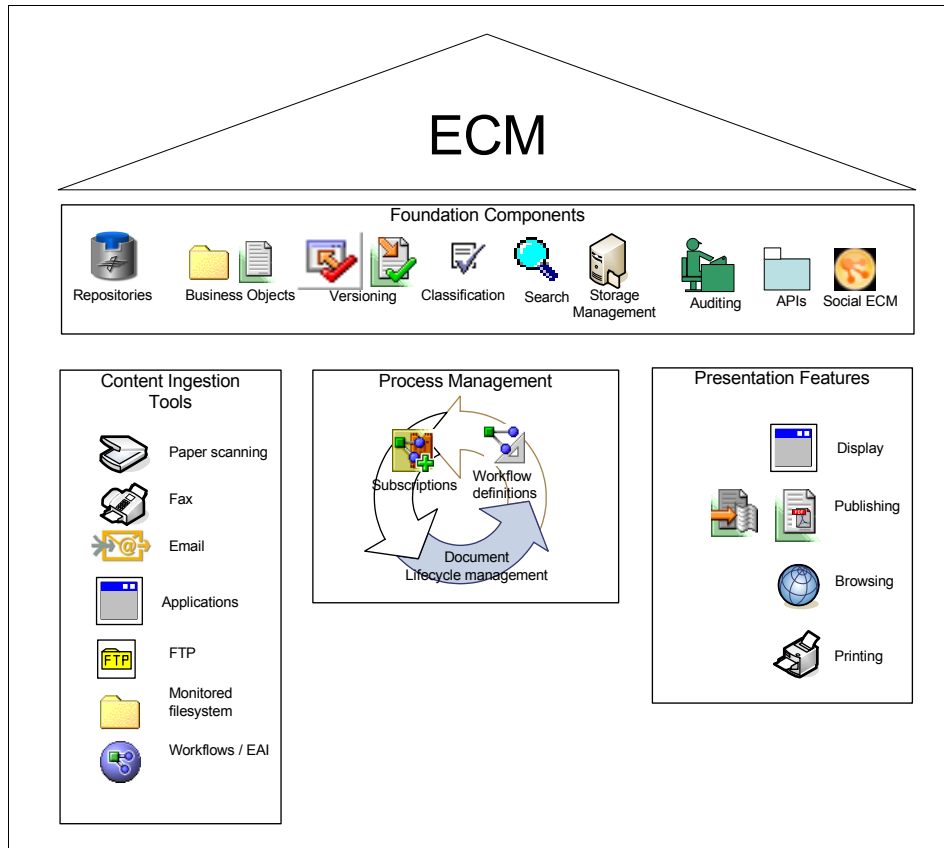


Figure 13-2 ECM design phases

13.1.1 Foundation components

We describe the foundation components of the P8 Content Manager:

- ▶ Repositories
- ▶ Business object management
- ▶ Classification
- ▶ Versioning
- ▶ Security
- ▶ Auditing
- ▶ Search
- ▶ Content Platform Engine application programming interfaces (APIs)
- ▶ Content storage and content caching
- ▶ Lifecycle management and retention management
- ▶ Social ECM capabilities

Repositories

Repositories are the basic components of FileNet P8 Content Manager. Their main purpose is to store the business objects, for example, documents, images, folders, and custom objects, with the respective metadata and provide a centralized information library.

A single FileNet Content Platform Engine can serve several repositories that are also called *object stores*. An object store can store various business data, including structured and unstructured content, such as images, XML documents, Microsoft Office documents, and web pages. It can also be configured to store the content in a database, a file system, a fixed content device, or any combination of these options.

Business benefits: The FileNet content repository provides a standard solution for document creation, versioning, and check-out, or add and check-in. The FileNet content repository can house all documents in a central repository with accessibility for all authorized users.

Recommendations: Use repositories to separate business objects based on functional and logical purposes, enforce object security, and improve the manageability of objects. For example, you can create a repository for sensitive Human Resources objects, a second repository for customer-related objects, and a third repository for vendor-related objects.

Business object management

In addition to managing documents, P8 Content Manager also can manage other types of data, such as folders, custom objects, and annotations.

Folders are special objects that are used to relate other type of objects, such as document and custom objects, and provide a way to browse through other objects. Folders have the following characteristics:

- ▶ Have system properties that the system manages automatically, such as Date Created.
- ▶ Can have custom properties for storing business-related metadata.
- ▶ Are secured.
- ▶ Can be hierarchical (a folder can have subfolders).
- ▶ Can contain documents and custom objects.
- ▶ Can generate server events when they are created, modified, or deleted. These events are then used to customize behavior.
- ▶ Can be annotated.

- ▶ Provide *containment by reference*, which allows any specific object to be filed in multiple folders at the same time.

Business benefits: Folders provide an option to organize and structure documents and custom objects in logical entities. For example, you can create a loan folder that contains all the relevant business objects of a loan application.

Recommendations: When you use folders, be aware of the following design considerations that might affect overall system performance:

- ▶ The number of folders
- ▶ The depth of a folder structure
- ▶ The number of objects in a folder
- ▶ Folder items with the same name in the same folder because this duplication violates a uniqueness constraint.

Custom objects contain only metadata without any content and are used to store business information. Custom objects have the following characteristics:

- ▶ Have system properties that the system manages automatically, such as Date Created.
- ▶ Can have custom properties for storing business-related metadata, such as Account Number.
- ▶ Are secured.
- ▶ Can participate in business processes as workflow attachments.
- ▶ Can generate server events when they are created, modified, or deleted. These events are then used to customize behavior.

Recommendations: Use custom objects to store data that relates to your business requirements, for example, a client can be represented as a custom object.

Document objects represent the electronically stored information that is managed by the ECM system. Documents have the following characteristics:

- ▶ Have system properties that the system manages automatically, such as Date Created.
- ▶ Can have custom properties for storing business-related metadata about the document.
- ▶ Are secured.

- ▶ Can have content that can be indexed for searching.
- ▶ Can point to content that is outside of the object store (external content).
- ▶ Can have no content (metadata only).
- ▶ Can be versioned to maintain a history of the content over time.
- ▶ Can be filed in folders.
- ▶ Can have a lifecycle.
- ▶ Can participate in business processes as workflow attachments.
- ▶ Can generate server events when they are created, modified, or deleted. These events are then used to customize behavior and trigger any custom events or workflows in other systems
- ▶ Can be rendered to different formats, such as PDF and HTML, by using the add-on IBM Rendition Engine.
- ▶ Can be published to a website.
- ▶ Can be annotated.
- ▶ Can be audited.

Annotation objects represent relevant incidental information about objects that can be associated with custom objects, folders, or documents. Annotation objects have the following characteristics:

- ▶ Annotation security is independent from object security. Default security is provided by the class and by the annotated object. An annotation can optionally have a security policy assigned to it.
- ▶ Can have subclasses.
- ▶ Can have zero or more associated content elements, and the content does not need to have the same format as its annotated object.
- ▶ Are uniquely associated with a single document version and, therefore, are not versioned when a document version is updated.
- ▶ Can be modified and deleted independently of the annotated object.
- ▶ Can be searched for and retrieved using the Content Engine API.
- ▶ Can subscribe to server-side events that fire when an action (such as creating an annotation) occurs.
- ▶ Can participate in a link relationship.
- ▶ Can be audited.

Classification

Each *document class* defines the properties and the default security for all the documents that are added under a specific document class. Document classes support design based on organization content or function and can encapsulate a single design aspect. Classification of the documents is performed by selecting document class and property values for each document. Classification can also be performed by adding objects into folders that define classification taxonomies.

Classification can be performed in the following ways:

- ▶ Manually by a user.
- ▶ By an application that uses the P8 Content Engine API.
- ▶ Automatically by using the content-based classification capability that is provided in the P8 Content Platform Engine.

Business benefits: Document classes support transparent business functionality and can automatically file the document in the correct folder and apply the correct retention schedule.

Recommendations: Define a top-level document class with all the properties that are common across all objects in the enterprise. All specific application objects are children of the enterprise document class.

Versioning

Versioning is a base document management capability that is used to maintain the history of the changes of the document content. The set of versions for a single document is called a *version series*. P8 Content Platform Engine supports a minor and major version scheme; a minor version typically represents a “work in progress” document and a major version represents a completed document.

The system can be configured to apply security policies that in turn automatically apply different access rights for major and minor versions, making it easy to enforce a different viewing audience for in-progress documents.

In addition to version numbers, P8 Content Platform Engine maintains a state property that indicates the current state of each document version. The states are listed:

- ▶ In Process: A work-in-progress version. Only one version of a version series can be in process.
- ▶ Reservation: A document currently checked out for modification. Only the latest version of a version series can be reserved.

- ▶ Released: A document released as a major version. Only one major version of a version series can be released.
- ▶ Superseded: A version superseded by another version. Many versions in the version series can be superseded.

Business benefits: A version series retains all history of the content and supports the correct information management.

Recommendations: Define security on versions in order to ensure that only authorized users can access “work in progress” versions or completed versions.

Auditing

Auditing is the recording of events that occur on business objects. All business objects and almost all events can be audited. *Audit definitions* describe how to audit an event. For example, you can configure an audit definition for a document class so that audit entries are automatically created whenever documents of that class are checked in.

Audit entries are stored in a table of the object store database. Those entries can be viewed, exported for reporting reasons, and administered by users with the correct authorization.

Business benefits: Auditing helps you monitor content and process management for the following activities and regulatory compliance:

- ▶ Object creation
- ▶ Updates
- ▶ Deletions

Recommendations: Only enable auditing for selected events and plan for audit log cleanup. Audit logs are stored in a database table and a large audit log table can create performance issues.

Security

In P8 Content Manager, you secure the business objects by defining a directory service that controls who logs on to the Content Platform Engine by setting access rights for those users.

P8 Content Manager has a defined security context. Only those users, groups, and machine accounts that are explicitly given access to the object store can access the resources (business objects).

There are many ways to define the security of a business object:

- ▶ **Default instance security:** The security of the object is defined in the object class and inherited to all instances of that document class.
- ▶ **Version state:** The security of the object is defined by the version level of the document. For example, some users might have access to minor versions of the document, and other users have access only to major versions of the same document.
- ▶ **Document state:** The security of the document is controlled by the document state.
- ▶ **Marking sets:** The security of the document is controlled by a property value and by the code you implement that interprets the meaning of the property.
- ▶ **Directly applied security:** The security of the object is assigned directly to the object by a user or an application.
- ▶ **Inherited security:** Security is placed on the object by a security parent or by setting up a relationship with an object-valued property whose Security Proxy Type is set to `Inherited`.

Business benefits: Using the model where document privileges are assigned from Directory Services functional groups, not individual users, helps reduce the cost of managing the security of the system by reducing security access complexity and by handling the separation of duty requirements.

Marking sets allow access to a document to be controlled based on specific property values to ensure that sensitive information is protected, for example, `Secret/Confidential`. With marking sets, you ensure document security and privacy control, limit access to sensitive data, and control access to documents.

Recommendations: Always assign security to LDAP groups and not to specific LDAP users. Assigning rights to groups gives you more flexibility over object access. (You can easily add or remove users to the specific group, therefore giving or removing access to objects that are already stored.)

Try to avoid the Deny security option on objects, which is an implicit way to remove access from users or groups.

Marking sets are stored in the global configuration database (GCD). If you add too many of them, it might affect the performance of the system.

Search

P8 Content Manager supports property and content-based searches.

With property searches, a user or an application can search multiple object stores for business objects that have a specific value in a property. Therefore, users can search for documents, custom objects, or folders in different object stores based on a property value.

Searches are defined in P8 Content Manager as SQL queries and support many of the standard SQL operators, such as **OR**, **AND**, **LIKE**, **UNION**, and **INTERSECTION**.

Search definitions can be created and then stored in an object store, allowing users easy access to common queries.

Content-based retrieval (CBR) supports searching within the content of a document or the metadata. CBR provides capabilities to search for misspelled words, typographical errors, word stems, synonym expansion, and wildcards. CBR search results can be ranked by relevancy and can display a document summary format.

Bulk operations can be performed on search results. Operations can be scripted or selected from a set of predefined operations, such as delete, cancel checkout, file, unfile, and change security.

Business benefits: Content Platform Engine search capabilities provide an effective means of locating information, improving the ability to share information across the organization, and enhancing record requests, thus improving organizational efficiency.

Recommendations: Always specify maximum limit for returned results. Searches are translated as queries against object store databases. Long running, non-optimized queries negatively affect the overall system performance.

Do not use wildcard searches because they can create problems, such as database table locks, and performance issues.

Search is not a reporting tool and you must not use it that way.

APIs

Content Platform Engine has a collection of available development and integration tools.

Depending on your enterprise strategy and architecture, you can use any of the following APIs for application development:

- ▶ Content Engine Java API: Provides access to the full content capabilities of Content Platform Engine
- ▶ Content Engine .NET API: Is the functional equivalent of Content Engine Java API for .NET application development
- ▶ Content Management Interoperability Services API: Is an open source OASIS standard that enables applications to work with one or more content management systems by defining a standard domain model and a standard set of services
- ▶ Process Java API: Provides classes for all workflow and business process management features
- ▶ Process Engine REST Service: Is used from custom applications to perform fundamental business process management operations
- ▶ Web Services: Provides a service that provides access to most of the same functionality as the Content Engine and Process Java APIs

Business benefits: The P8 Content Manager APIs provide a way to integrate with other line of business applications, improve the user experience by offering functionality, and access multiple systems transparently.

Content storage management

P8 Content Manager supports storing content in a file system, a fixed content storage system, and the object store database. Depending on the requirements of your solution, choose one or more of these options for content storage.

Database storage is useful when you only need to store a few, small documents. There are performance advantages to storing smaller documents (less than 10 MB) in a database storage area when compared to other storage area types. Avoid storing any document that is over 100 MB in a database storage area. The main benefit of database storage is that backups are much simpler because your document content is backed up along with your normal object store database backup.

File and fixed storage areas are the preferred medium when storing large numbers of files with high ingestion rates.

File storage areas use a directory structure on the file system to store a document's content. Documents are stored among the directories at the leaf level by using a hashing algorithm to evenly distribute files among these leaf directories.

Fixed storage areas are used to store documents in external repositories, such as IBM Tivoli Storage Manager, EMC Centera, and Network Appliance SnapLock. There are two scenarios for the integration of P8 Content Manager with those external repositories. In the first scenario, the document content is managed by Content Manager and the external device is used only as a storage device. In the second scenario, fixed storage areas are used with federation when content is stored in an external repository. In this scenario, the document and its associated metadata can be accessed as native P8 documents, in addition to their accessibility via the source repository.

P8 Content Manager also provides the following features for storage management:

- ▶ Bulk content move: With the bulk content move sweep job, you can move content from one storage area to another. There is also a Move Content API method, which can be used from other applications to move content from one storage area to another. Content can be moved from any storage area type (database, file system, or fixed content) as long as content is not under device-level retention.
- ▶ Content caching: Database file store area content and fixed storage area content can be cached on a cache server. For frequently accessed content, content caching provides a faster response time in content retrieval. Content caching also benefits geographically distributed systems and systems with hierarchical storage devices by storing copies of content local to where they are accessed most often.
- ▶ Content de-duplication: P8 Content Manager supports de-duplication of the content that is ingested from various sources. This feature saves storage costs for duplicate content that is saved in the repository.

- ▶ **Content compression:** Content Manager supports the compression of the content that is stored in a storage area. The compression is transparent to the client that is storing or retrieving the content. This feature saves storage costs but it has performance implications because the compression and decompression of the content occurs when the content is stored or retrieved.
- ▶ **Content encryption:** This feature helps protect the documents in the storage areas from unauthorized access directly to the storage area. The encryption and decryption of the content is performed automatically by the Content Platform Engine. The entire process is transparent to the users, but it imposes performance penalties when the content is stored for the encryption and when the content is retrieved for the content decryption.

Business benefits: Content Manager storage management provides many options to satisfy every functional and technical requirement.

Recommendations: Encrypt all the sensitive content of your organization to ensure privacy. Encryption is defined on the storage area level. Enabling the encryption on a storage area does not encrypt the content that is already stored in that storage area.

If your organization is geographically dispersed, use content caching and minimize the network traffic for content retrieval.

If you are using hierarchical storage devices with tape support, such as IBM IBM Tivoli Storage Manager, consider using the content caching capability for slow content retrieval from tapes. Using content caching, the retrieval might time out but the content cache continues to retrieve the content so that the content is available in cache for the next user request.

If you are using records management, be careful with the content deduplication feature of Content Manager.

Retention management

Retention management is an event-based retention infrastructure that can define object-level retention policies. It is supported for documents, annotations, folders, and custom objects.

A retention management automatic deletion and disposal policy defines the rules for when objects are automatically deleted.

The policy has these characteristics:

- ▶ Can apply to any searchable repository object.

- ▶ Allows documents, folders, and custom root classes that have a retention date in the past to be deleted.
- ▶ Allows custom root classes that have a closure date in the past to be deleted.
- ▶ Can delete queue items that have reached the maximum failure count more than one month in the past.
- ▶ Is based on values assigned to the CmRetentionDate system property.
- ▶ Can also be based on system-defined or user-defined properties.

Business benefits: Implementing a retention management scheme helps organizations meet organizational, business, regulatory, and legislative requirements.

Recommendations: Always define a retention period for the content that is considered critical for your organization. Retention periods that are defined on Content Manager need to match the retention period of your records management requirements. Also, by defining a retention period, you can decrease the volume of information on your Content Manager repository and provide only relevant information to the business users of your ECM system.

Social ECM

P8 Content Manager can be used for the enablement of social collaboration, social content management, and integration with IBM Connections.

P8 Content Manager supports the following social ECM features:

- ▶ Ability for users to recommend a business object.
- ▶ Ability for users to comment on managed objects. Comments can only be created by authorized users.
- ▶ Ability for users to “follow” updates to business objects.
- ▶ Social tagging of managed objects.
- ▶ Activity stream generation for business objects. *Activity streams* provide a syndicated view of updates to the content, including notifications and recommendations.
- ▶ Tracking the number of downloads of a document.
- ▶ Large content streaming.
- ▶ Thumbnail generation and storing.
- ▶ User-centric recycle bin for deletion and recovery of documents.

Foundational components provide the strong build blocks for ECM enterprise solution for organizations across multiple industries.

13.1.2 Content ingestion tools

IBM FileNet offers several content ingestion (capture) applications, each of which is designed for capturing a different type of media. The capture applications listed in this section can handle the following media types:

- ▶ Paper documents
- ▶ Faxes
- ▶ Office (electronic) documents, presentations, or spreadsheets
- ▶ PDF, web, .txt files, or multimedia files
- ▶ Email messages
- ▶ Documents stored on network drives or desktops
- ▶ Documents stored in remote repositories, such as IBM FileNet Image Services, IBM Domino®, OpenText, Microsoft SharePoint, or EMC Documentum

IBM Datacap Taskmaster Capture

IBM Datacap Taskmaster Capture is used as a capture portal for all the documents that are managed by an organization. IBM Datacap Taskmaster Capture offers these features:

- ▶ Scans and verifies all paper-based documents
- ▶ Supports multiple recognition engines that are configurable with rules for data extraction from scanned documents
- ▶ Supports bar code recognition for automatic classification and indexing
- ▶ Supports web-based remote scanning and verification by using a browser
- ▶ Supports the import of documents from file systems

Bulk Import tool

Bulk Import is a tool that is designed to help organizations move document images in Content Manager even when those images are created by third-party tools or external processes. With this tool, you can store document images from files at a rate of more than a million documents per day. It also can assemble documents from more than one image, create batches of documents, and assign metadata and indexing properties to documents.

Microsoft Office Integration facility

IBM Content Navigator and FileNet Workplace XT integrate with Microsoft Office applications to help the casual business users manage documents, emails, and attachments. By using the Microsoft Office Integration facility, business users can add new documents to an object store and retrieve and update existing content. This integration provides access to most ECM features, such as hierarchical folder navigation, version control, metadata management, security, and even approval workflows.

Recommendations: For casual users who use only Microsoft Office content, consider this feature of Content Manager as your primary option.

IBM Content Collector for Email

IBM Content Collector for Email supports organizations to gain control of email in order to meet record management obligations, connect email to business processes, and manage storage space.

With Content Collector for Email, you can monitor the incoming and outgoing emails of an organization and, based on rules, archive the messages in Content Manager. It also supports the extraction of message attachments and storage, based on user-defined rules, such as keywords and email addresses, to Content Manager.

It also helps organizations to reduce the storage requirements for their email management system by leveraging the storage management features of Content Manager.

IBM Content Collector for SAP

IBM Content Collector for SAP provides outbound archiving and retrieval of SAP generated business documents, SAP reports, and database data. It provides inbound archiving and retrieval for external documents, such as invoices.

Content Collector for SAP reduces operational costs by managing the growth of SAP application data through archiving. It increases the efficiency of SAP users and business processes by linking relevant content to SAP transactions.

IBM Content Collector for SAP provides two options for integration with the SAP system. One option is the P8 client that can link documents and folders selected by a user to an SAP transaction. The second option is called *Utility Client*. Utility Client can link an archived document on Content Manager by processing bar codes or by creating work items in the SAP workflow.

IBM Content Collector for Files

IBM Content Collector for Files helps organizations manage documents on network file shares and provides tools to help users comply with corporate and regulatory policies.

IBM Content Collector for Files automatically captures documents that are placed on a monitored location of a file share and uses the Content Manager archiving and deduplication capabilities to help reduce infrastructure costs that are associated with the management of file systems. It permits advanced file handling based on rules. It can be configured to leave documents in place or move them and replace them with shortcuts.

IBM Content Collector for SharePoint

IBM Content Collector for SharePoint provides collection and archiving. It also provides extended enterprise content management and business process management capabilities for the SharePoint content.

IBM Content Collector for SharePoint collects and archives content from SharePoint document libraries, wikis, and blogs and automatically classifies it. Collected documents are replaced by a shortcut in SharePoint. It can also be configured to declare collected content as a record by using IBM Enterprise Records.

Content Federation Services

Content Federation Services (CFS) unifies content from different repositories into one or more object stores so that the content and associated metadata can be accessed by using the P8 Content Manager suite of products.

With CFS, you can put metadata from heterogeneous repositories into object stores so that the metadata is available to all users. You can also manage the full lifecycle of the digital content, enforce records management policies, and provide an enterprise-wide content search mechanism regardless of the repository in which the content is stored.

Business benefits: Content Ingestion tools help make the collection of data easy and quick. They simplify information search and can automatically extract data from documents during the ingestion process. Using the content ingestion tools indexing process can be standardized, simplified, and independent of the organizational changes.

13.1.3 Process management

IBM FileNet offers various solution building blocks for process management.

Events and subscriptions

Events provide a mechanism to initiate actions that are executed when objects are created, modified, and deleted from an object store. A *subscription* is the association of a particular event trigger with an event action. Many subscriptions can be associated with an event trigger.

Subscriptions can be associated with a class so that they apply to the class itself or to all instances of all objects of that class type, or they can be associated with a specific object. Event subscriptions can be executed synchronously or asynchronously. When set to run synchronously, the object action (for example, create or update) and the operations of the event actions are completed as a single transaction; failure in either results in the rollback of both operations. When set to run asynchronously, the object action and the event action operation run as separate transactions; in this case, the object operation can succeed independently of the event action operation.

Business benefits: Subscriptions can streamline the business process and make all integration transparent to the business user.

Recommendations: Subscriptions provide a powerful way to activate your content. You can define a subscription so that a workflow is launched when a document is created or modified. For example, when a loan application document is created, a loan approval workflow is initiated, a loan folder is created, the client information is retrieved from core systems, and a custom object is created with the information.

Change preprocessors

Change preprocessors are action handlers that change new or updated objects before they are saved to the Content Manager. *Change preprocessor handlers* are associated with a class definition. When an object of that class is saved, the action handler is triggered.

Change preprocessors allow object modifications that are difficult or impossible to accomplish by using event action handlers. For example, a change preprocessor can alter a modifiable-only-on-create property because those properties cannot be altered after the object is saved.

Content lifecycle management

The *content lifecycle* is a series of sequential states that a document goes through during its life.

A content lifecycle definition consists of two objects:

- ▶ Lifecycle policy: Identifies the allowed document states. The policy also identifies the lifecycle action that executes in response to state changes.
- ▶ Lifecycle action: Custom action that the system performs when a document moves from one state to another. A lifecycle action is typically coded by a developer, but managed by an administrator. The custom actions handle the following state changes:
 - Promote: Moves the document forward to its next lifecycle state.
 - Demote: Returns the document to its previous lifecycle state.
 - Reset: Returns the document to its first lifecycle state with each new version.
 - Set in exception mode: Prevents the document from changing lifecycle state.
 - Clear from exception mode: Enables the document to change lifecycle state.

The content lifecycle defines simple processes related to managing a document's lifecycle. For more complex document lifecycle actions, use IBM Case Foundation.

Business benefit: Ability for a group of people to work on the same document with proper version control and avoid duplication, providing real-time collaboration and information sharing to improve and streamline business processes.

Recommendations: For simple serial workflows, use content lifecycle management.

Business process management

IBM Case Foundation is a package that includes P8 Content Manager. Case Foundation can create, execute, manage, analyze, and simulate business processes (also referred as *workflows*) that are performed by users or applications.

By creating a workflow definition, you define the activities and resources that are needed to complete a business process. A *workflow definition* is a series of steps connected by a series of routes that defines the sequence that the steps are executed. Workflow definitions can contain several maps and submaps that can group related steps.

Steps in the workflow represent a business task or a system activity. Steps can be executed by a user, a group of users, or by an automated application. Workflow steps can run in parallel to facilitate efficient processes.

Routing defines the order in which the steps are executed. Routing can be based on a specific rule or events. Except for the last step, every step has one or more routes that lead to it. Routes can be defined so that they are always taken or followed only if a condition is met.

You can use deadlines and timers to ensure that work is processed in a timely manner. A *deadline* provides a time-based scheduling constraint, which requires that a step or workflow is completed within a certain amount of time. The deadline can be relative to the time that the step was routed to the participant or to the time that the workflow was launched. A participant with a deadline can receive a reminder of the pending deadline through an email message. When the deadline is passed, a visual reminder displays in the participant's inbox, and an email can be sent to a configurable list, such as one or more supervisors. The distribution list can be specific to each work item. This automatic process escalation has the additional benefit of operatively ensuring that certain functions or processes are completed on time and without tying up resources to continuously monitor system activities.

A *timer* indicates a time during which you want a specified series of steps to process. If the timer expires before this processing completes, processing proceeds to another workflow map that provides alternate processing of the work.

Recommendations: For complex content-centric processes, use the process management capabilities of Content Manager. Examples of content-centric processes are loan origination processes and insurance claim processes. Using Content Manager process management, you can activate the organizational content and take control over processes that involve documents.

13.1.4 Presentation features

Several P8 Content Manager features are available to present content to your users. IBM FileNet P8 presentation features include options for converting active content to the following formats:

- ▶ Native content format
- ▶ PDF
- ▶ HTML
- ▶ Darwin Information Typing Architecture (DITA) documents
- ▶ Annotations

Native content format

FileNet Content Manager stores the content in the native content format (format in which the content is created). When the content is retrieved from Content Manager, the content stream is passed to the client and managed by the associated client application. Content also can be displayed on the requesting client by using the embedded Content Manager Viewer.

Recommendations: Try to use the embedded Content Manager Viewer as much as possible. It provides multiple views of the content according to user preference and IT permission sets and it ensures security. It makes the ECM experience positive and transparent to the users by solving issues such as where and how to store documents.

PDF and HTML presentation

The *Rendition Engine* is a P8 Content Manager add-on that facilitates document publishing. Publishing a document enables converting a document into PDF or HTML format, or generating a replica of the document in either PDF or HTML format. The replica, which is known as the *publication document*, can have its own security and property settings. Publishing can be triggered by event actions or by changes in a document's lifecycle state. When a document reaches the "released" lifecycle state, for example, a PDF version can be automatically created with public-view security rights.

Published documents have these characteristics:

- ▶ Can continue to exist after the source document is deleted based on the assigned retention schedule and business rules
- ▶ Can be automatically deleted when the source document is deleted
- ▶ Are not changed when their source documents are changed
- ▶ Can exist in a different folder than the source documents

- ▶ Can have a different file format than the source documents, for example, the source document can be a word document, and the publication document can be an HTML document. Publishing options are defined by individual templates
- ▶ Can originate as Microsoft Office (for example, Word, Excel, and PowerPoint) documents and be rendered to PDF or HTML

DITA documents

DITA is an XML-based open standard for developing, managing, structuring, and publishing content. IBM originally developed DITA for more efficient reuse of content in product documentation. IBM donated DITA work to the Organization for the Advancement of Structured Information Standards (OASIS) for further development and public release.

The content can be composed based on the DITA model that allows content to be linked to multiple topics. After the content is reviewed and approved, it is published to allow business users to perform searches and to navigate around the content.

The two central units of authoring in DITA are the topic and the map. The *map* combines multiple topics into a structure that has a unique map. The *topic* might appear in different manuals, and in multiple sections of the final document. Maps are XML documents that consist of links to topics and metadata. Maps do not have content themselves. DITA content (topics and maps) is rendered into PDF and HTML.

Storing each piece of content in a separate file allows users to check out, revise, check back in as a new version, and reuse the single source material in multiple locations.

Next, we review the sample use cases from Chapter 2, “Solution examples and design methodology” on page 17. We use the available components of Content Manager and explain how those components are used to address business requirements.

13.2 Sample use cases using solution building blocks

We explore various IBM FileNet Content Manager sample solutions through use case descriptions. These use cases correspond to the use cases that we introduced in Chapter 2, “Solution examples and design methodology” on page 17.

13.2.1 Policy document creation use case

The use case concerns policy or procedures and safety documentation. The departments that are in charge of the policy need to go through a simple lifecycle process to develop and publish a new policy on the web or to update an existing policy. This process includes transforming an idea into policy, implementing the policy actions, and then evaluating and measuring policy performance. This example, which is illustrated in Figure 13-3 on page 450, is a typical first project when introducing P8 Content Manager into a company.

The use case has the following requirements:

- ▶ Documents can be authored, reviewed, approved, or released.
- ▶ There are four roles (author, reviewer, approver, and user).
- ▶ Each role has certain permissions.
- ▶ Users can only see approved documents and always the latest version.
- ▶ Changes to documents need to be audited.
- ▶ Documents must be filed in folders according to their classification.
- ▶ Policy documents exist in a shared folder on the network.
- ▶ Users must be able to search documents based on their properties or words inside the content.
- ▶ A PDF version of the policy is published to the intranet/Internet.

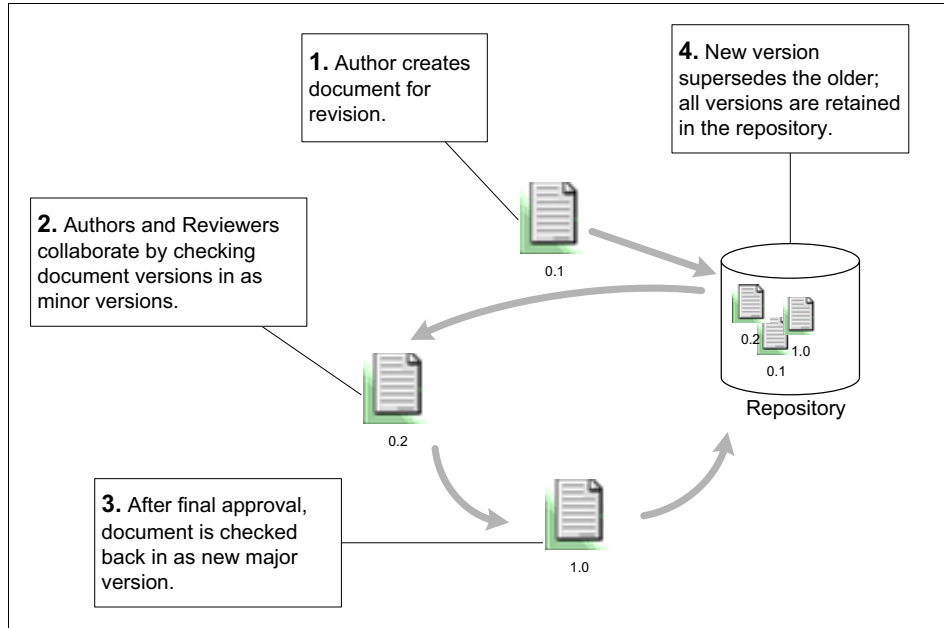


Figure 13-3 Policy document review and approval use case

Based on the requirements, this use case uses these components:

- ▶ Foundation components:
 - Document classes for classification of documents
 - Folders for document classification
 - Minor and major versions
 - Security based on version state
 - Auditing for document changes
 - Property-based search and content-based retrieval for locating documents
- ▶ Content ingestion tools:
 - Microsoft Office integration for storing documents directly from Microsoft Office applications
 - IBM Datacap Task Master for scanning paper-based documents and importing already created policy documents from a shared folder
- ▶ Process management
 - Content lifecycle management for the different states of the policy approval process

- ▶ Presentation features:
 - Native content format
 - Rendition Engine for PDF output

Solution details

As described in the requirements, authorized users must be able to check out policy documents from the repository, change the content, and put it back in the repository as a new version. For this requirement, we use P8 Content Manager checkout, checkin, and versioning capabilities and Microsoft Office integration. By adding a version to the repository, we use the content lifecycle management capabilities of P8 Content Manager, and we assign the lifecycle state Pending approval on the document. Approvers can check out the document, review the content, and change it, creating a version of the document. They can return the policy document to the previous state with comments for the author by adding a new minor version, or they can approve the document and put it in the approved state by adding a new major version.

In the requirements, there is the need for the user community to easily locate the policy documents that are stored in an object store. For that purpose, we use the object store foldering capabilities. We are creating logical folders for the policy documents where users can put them according to their classification (for example, Human Resources policies or procurement policies). We are also providing to users the ability to search for policy documents based on their properties. The P8 Content Manager search capabilities allow you to search for documents by using any combination of document properties (for example, Human Resources policies that were published two years ago).

According to the requirements, general users must have access only to approved documents. Authors and reviewers must also have access to draft documents.

We use object store security to present the draft documents to the special users that create, review, and approve the policy documents.

Also, there is a requirement that an approved policy document must be published to the company's intranet site as a PDF document. For this requirement, we use IBM Rendition Engine for PDF generation and publishing.

Finally, there is the need to import all already created policy documents to Content Manager. For this step, we use the file import capabilities of IBM Datacap task master.

13.2.2 Insurance claim processing use case

This use case concerns the handling of an insurance claim. The claims processing department needs to handle the documents that are associated with a claim. The compliance department needs to declare records after the claim is closed and define the records retention period. Paper-based documents that are sent to the claims department need to be scanned. A tight integration with the core insurance claim application is needed. This example, which is illustrated in Figure 13-4 on page 453, presents a typical scenario of a use case with significant integration with external systems and other content repositories.

This use case has the following requirements:

- ▶ Integration of core insurance application with Content Manager.
- ▶ Folders for document organization.
- ▶ Document scanning and optical character recognition (OCR) of scanned documents.
- ▶ Email capture of claim-related documents.
- ▶ Events for document indexing from the Claim Management System and exception process notification.
- ▶ Records management for claim document declaration as enterprise records and to set retention.
- ▶ Documents must be available for viewing by authorized users during the claims handling process.
- ▶ Insurance policy documents from other repositories are displayed.

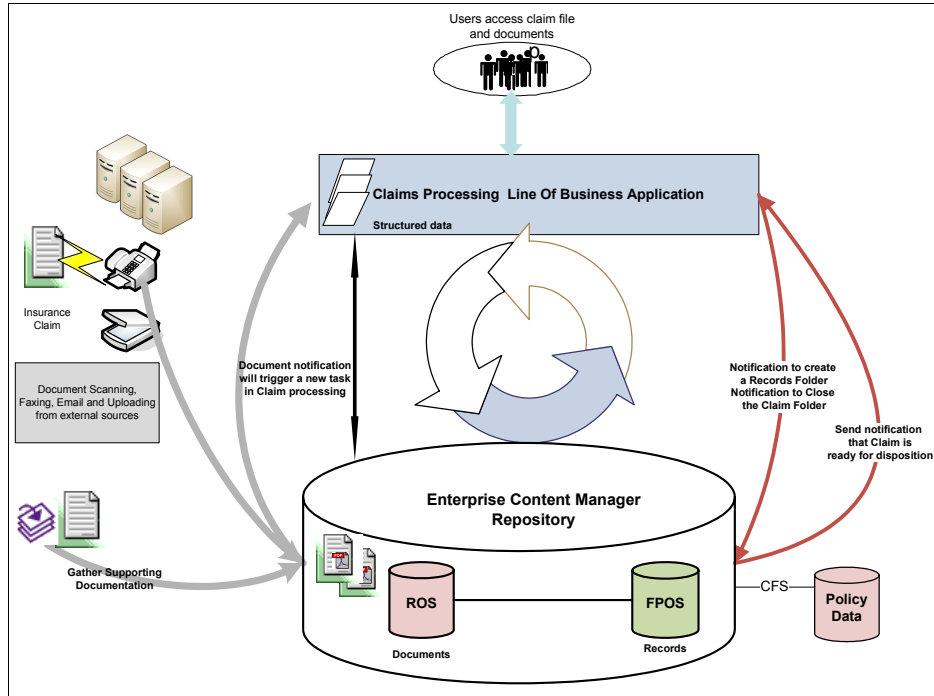


Figure 13-4 Insurance claim processing use case

Based on the requirements, this use case uses these components:

- ▶ Foundation components:
 - Records folders for records filing
 - Records management for insurance claim documents
 - Annotation for marking and highlighting specific parts of the documents
 - Content Manager APIs for integration with the Claim Management application for folder creation and properties update
 - Events for task initiation on the Claim Management System
 - Security for display documents of each claim in authorized user groups
- ▶ Content ingestion tools:
 - IBM Datacap Task Master for document capture and OCR
 - Email archiving
 - Content federation services for content that is stored in other repositories

- ▶ Process management
 - Events for notification of content addition on the Content Manager repository
- ▶ Presentation feature
 - Content display in native format in claim processing department

Solution details

This complex solution uses many features of P8 Content Manager.

As described in the requirements, after claim notification, a new claim must be opened in the core Claim Management System. New claim registration on the core system triggers the creation of a new folder in IBM Enterprise Records. In that folder, all records objects will be created for the claim-related documents that are stored in the Content Platform Engine repository. For that requirement, we use Content Manager APIs for integration with the Claim Management System.

When the paper document arrives, we use IBM Datacap Task Master to scan that document. By using OCR/intelligent document recognition (ICR) capabilities, we retrieve the claim number from the paper documents. Using that claim number, along with other indexing information, such as the document type, the document is stored in the Content Manager repository and filed under the claim folder. A notification is sent to the Claim Management System. For the notification of the Claim Management System, we use Content Platform Engine events to execute certain code when a document is added in the ECM system.

During the claim lifecycle, the Claim Management System updates documents and folders on the ECM system by using the Content Manager APIs with the current claim status.

Due to sensitive personal information in the document, only authorized users must have access to claim documents based on the claim status. For that requirement, we use object store security and marking sets that control the access to the document based on a property value (claim status).

According to the requirements, the Claim Management System users must have access to insurance policy documents that are stored in a different Content Manager repository. For that requirement, we use Content Federation Services to integrate the claim repository with the insurance policy repository.

Users must be able to provide casual information around the documents, such as comments, or highlight a specific portion of a document that contains critical information. For that capability, we use Content Manager annotations over the documents.

When the claim is closed in the Claim Management System, a notification needs to be sent to trigger a retention for the claim documents. For that requirement, we use the IBM Enterprise Records features and APIs.

13.2.3 SAP invoice archiving use case

This use case covers the paper invoice data extraction and archiving. The accounts payable department needs to process many paper invoices, update the Enterprise Resource Planning (ERP) system and control the invoice processing. Paper invoices that are sent to the accounts payable department need to be scanned, the information needs to be extracted, and an update must be made to the SAP system with the information on the paper invoice. Furthermore, the scanned image needs to be associated with the corresponding SAP transaction for auditing reasons. The example is illustrated in Figure 13-5 on page 456.

This use case has the following requirements:

- ▶ Scanning paper documents.
- ▶ Data extraction from scanned documents.
- ▶ Extracted data must be validated against the SAP system.
- ▶ Data must be confirmed by authorized employees of the accounts payable department.
- ▶ A record of the invoice must be created on the SAP system and the scanned document must be associated with the SAP record.
- ▶ The scanned image must be available through the SAP system.
- ▶ Authorized users must be able to search for the invoice document without having access to the SAP system.

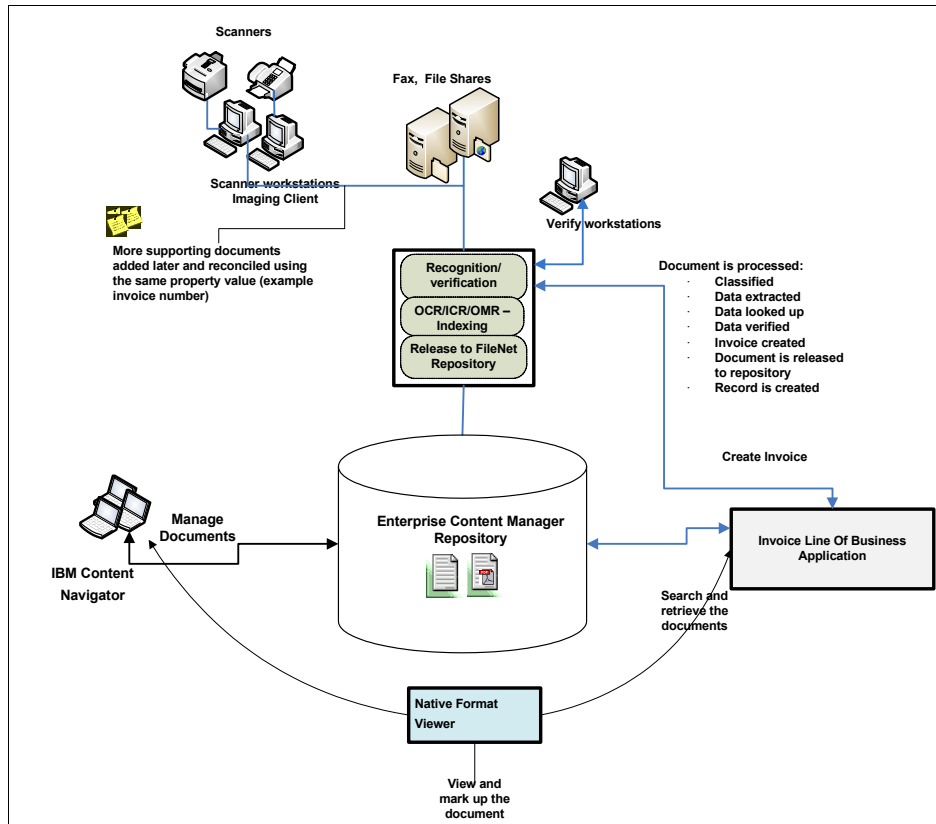


Figure 13-5 SAP invoice archiving use case

Based on the requirements, this use case uses these components:

- ▶ Foundation components:
 - Repositories to store invoice images
 - Content Manager security
 - Property search
- ▶ Content ingestion tools:
 - Datacap Taskmaster Capture
 - IBM Content Collector for SAP

- ▶ Presentation features

Native content format with Image Viewer Pro, which supports the scanned image display

Solution details

Based on requirements, all incoming invoices must be scanned on arrival and data must be exported from the scanned images. For that requirement, we use Datacap Task Master scanning and OCR/ICR capabilities.

Exported data must be validated by authorized users and the data accuracy must be verified. For that requirement, we use Datacap Task Master validation features.

The scanned images of the invoices must be stored in the Content Manager and become available to authorized SAP users to link those documents to SAP transactions. For that requirement, we use IBM Content Collector for SAP, which provides the functionality for linking Content Manager images to SAP transactions.

SAP users must be able to view the scanned image of the invoice on the related SAP transaction. For that requirement, we use IBM Content Collector for SAP and the native content format viewing presentation feature of Content Manager.

Authorized users must have access to scanned images outside of the SAP system and must be able to search for those images based on their properties. For that requirement, we use Content Manager security and search features.

13.2.4 Email capture for compliance use case

This use case covers the email archiving and records management needs of an organization. Laws in several countries require the preservation, archival, and eDiscovery of emails that are addressed to an organization. Furthermore, email archiving helps reduce the storage requirements for the email management system. This example is illustrated in Figure 13-6 on page 458.

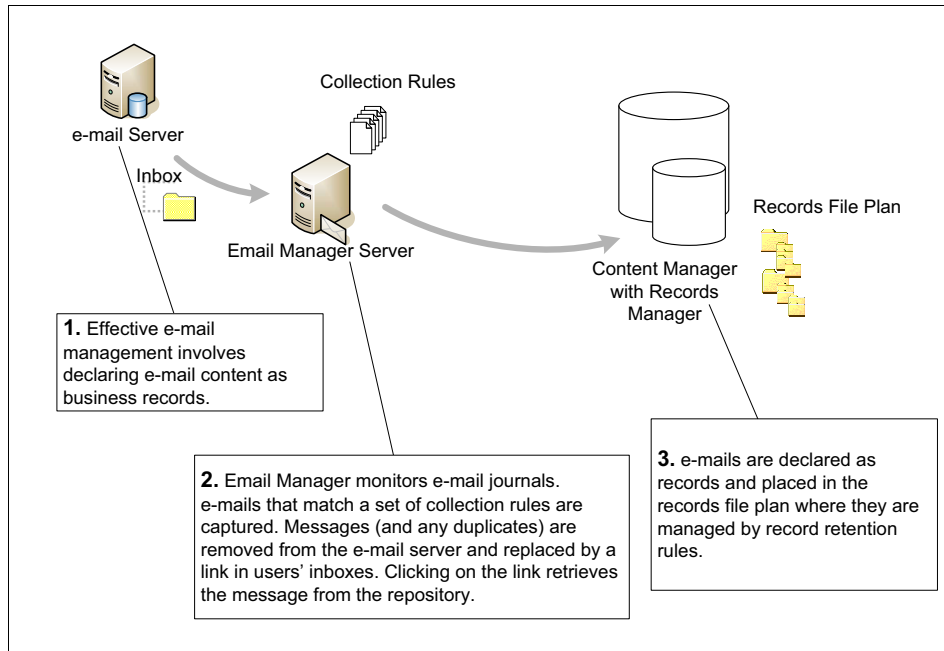


Figure 13-6 Email capture for compliance use case

This use case has the following requirements:

- ▶ Email to various accounts must be monitored and, based on enterprise rules, must be stored in an object store.
- ▶ Based on enterprise rules, a record must be declared, and a retention period must be defined.
- ▶ Employees must have access to their archived mail through the mail client.
- ▶ Authorized employees from other departments, such as the Legal department, must have access to the archived emails.
- ▶ Content-based retrievals must be available for locating emails with specific words in the mail body or on the subject line.

Based on the requirements, this use case uses these components:

- ▶ Foundation components:
 - Object stores
 - Storage management
 - Content-based searches
 - Records management

- ▶ Content ingestion
 - IBM Content Collector for Email

Solution details

According to the requirements, mail to specific accounts or mail that meets a rule (for example, contains the word “proposal”) needs to be stored in an object store. For that requirement, we use IBM Content Collector for Email. It monitors mailboxes, retrieves email based on business rules, and stores it in an object store.

Some of the emails that are considered special based on business rules are declared as records by using IBM Enterprise Records. Emails are associated with a retention period based on legislative requirements.

Users must be able to see the emails and their attachments in their mail clients, but the content must be stored in an object store. For that requirement, we use the stubbing capabilities of IBM Content Collector for Email. With that capability, the original content of the email is removed from the email server and is replaced by a stub that points to the object store where the content is stored.

For the emails that are not declared as records, content deduplication and content compression are needed, specifically when an email with a large attachment is sent to multiple recipients within the organization. For that requirement, we use the content deduplication and content compression feature of P8 Content Manager.

Authorized users must have access to those emails and attachments that are declared as records outside the mail clients of the user. In order to locate specific emails and attachments, searches within the content are required. The searches must be implemented by using the CBR capabilities of P8 Content Manager.

13.2.5 Knowledge management through collaboration use case

This use case covers the collaboration between authors and experts for the development of an organization’s training material and the collaboration between trainees for content recommendation, tagging, and recommendations. Training material has to be prepared by an author and published to a social community of subject matter experts (SMEs). Using the social features of P8 Content Manager, the SME and content authors collaborate and share ideas and comments for the finalization of the training material. When the training material is finalized, it becomes available to the communities in which the trainees participate. In those user communities, trainees can add and view comments and download counts and recommendations from other members of that community. This example is illustrated in Figure 13-7 on page 460.

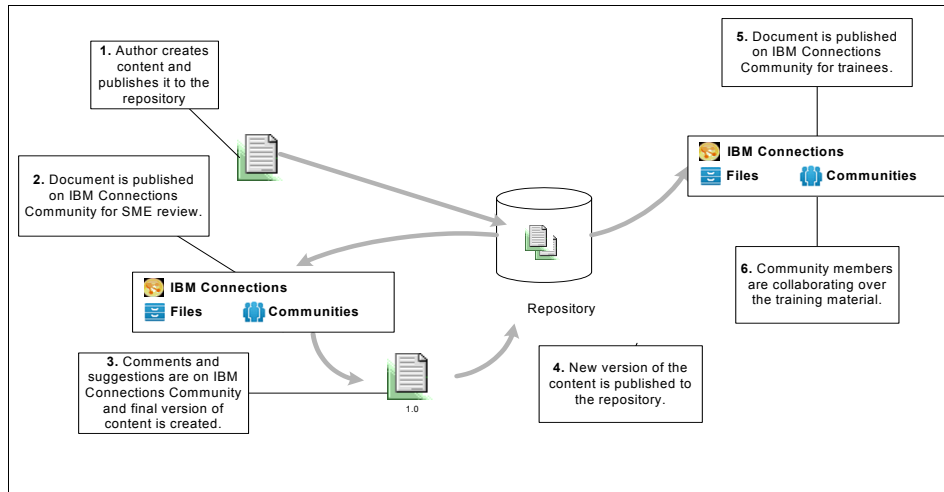


Figure 13-7 Knowledge management through collaboration use case

This use case has the following requirements:

- ▶ Support is needed for multiple versions of the content.
- ▶ Content must be published on the IBM Connections communities.
- ▶ Users must be able to put comments, recommendations, and tags on the content and download it.
- ▶ Users must be able to follow the content and get updated versions or comments over the content.
- ▶ Activity feeds over the content must be available to users.
- ▶ The recommendation counts and download counts must be available to the community users.

Based on the requirements, this use case uses these components:

- ▶ Foundation components:
 - Content repositories
 - Versioning
 - Social ECM features

Solution details

Based on the requirements, users must be able to create different types of training material from Microsoft Word and PDF to video and audio files and store them in Content Manager. For that requirement, we use the versions and folders capabilities.

After storing the content, it must be published in an IBM Connections community with SMEs for review and collaboration. The SMEs can create new versions of the content, write comments on it, and follow the content activity. By using the collaboration capabilities of Content Manager, a final version of the content is produced.

When the final version of the training material is available, that material must be published on an IBM Connections community where users that are members of that community can view and download the content. For that requirement, we use the Content Manager large content streaming capabilities.

Members of the community must be able to tag content and provide comments and recommendations. Also, they need to be able to “follow” the content when it is updated, view activity feeds on that content, and view download counts and recommendation counts. For that requirement, we use the Content Manager Social ECM capabilities.

13.3 Conclusion

In this chapter, we described the main solution building blocks of an ECM system. We described features and characteristics of Content Manager and add-ons that can be used for the implementation of a huge range of applications from small departmental applications to large Enterprise Content Management applications that cross the boundaries of many departments. As a reference, we used these solution building blocks for the implementation of the five use cases that we introduced in Chapter 2, “Solution examples and design methodology” on page 17.

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.

- ▶ *IBM FileNet P8 Platform and Architecture*, SG24-7667
- ▶ *Introducing IBM FileNet Business Process Manager*, SG24-7509 (the product is currently known as IBM Case Foundation)
- ▶ *Advanced Case Management with IBM Case Manager*, SG24-7929
- ▶ *IBM Content Analytics Version 2.2: Discovering Actionable Insight from Your Content*, SG24-7877
- ▶ *Disaster Recovery and Backup Solutions for IBM FileNet P8 Version 4.5.1 Systems*, SG24-7744
- ▶ *Federated Content Management: Accessing Content from Disparate Repositories with IBM Content Federation Services and IBM Content Integrator*, SG24-7742
- ▶ *IBM High Availability Solution for IBM FileNet P8 Systems*, SG24-7700
- ▶ *Understanding IBM FileNet Records Manager*, SG24-7623

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 websites are also relevant as further information sources:

- ▶ IBM FileNet Content Manager support website:
<http://www.ibm.com/software/data/content-management/filenet-content-manager/support.html>
- ▶ IBM FileNet P8 Version 5.2 Information Center:
<http://publib.boulder.ibm.com/infocenter/p8docs/v5r2m0/index.jsp>
- ▶ URL links to Version 5.1 of the IBM FileNet P8 Information Center:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r1m0/index.jsp?topic=/com.ibm.p8toc.doc/ic-homepage.html>
- ▶ URL links to Version 5.2 of the IBM FileNet P8 Information Center:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=/com.ibm.p8toc.doc/ic-homepage.html>
- ▶ Product documentation for IBM FileNet P8 Platform:
<http://www.ibm.com/support/docview.wss?rs=86&uid=swg27036917>
- ▶ *IBM FileNet Hardware and Software Requirements guide*:
<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27013654>
- ▶ FileNet P8 Fix Pack Compatibility Matrices:
<http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27014734>
- ▶ IBM FileNet Content Manager Fix Central - Provides available fixes:
<http://www.ibm.com/support/fixcentral>
- ▶ Information Center - Installation:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.install.doc/p8pti000.htm>
- ▶ Information Center - Supported upgrade paths:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.planprepare.doc%2Fp8ppu097.htm>
- ▶ Information Center - Database administration installation tasks:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.planprepare.doc%2Fp8ppi084.htm>
- ▶ Information Center - Planning and preparation for upgrade and migration:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8toc.doc%2Fplanning.htm>

- ▶ White Paper - *IBM FileNet P8 Platform Installation and Upgrade Best Practices*:
<http://www.ibm.com/support/docview.wss?uid=swg27010422>
- ▶ White Paper - *Simultaneously Upgrading and Migrating Content Engine*:
<http://www.ibm.com/support/docview.wss?uid=swg21455046>
- ▶ White Paper - *Best Practices for FileNet P8 3.5.2 to P8 4.5.1.2 Upgrade with Replatforming (Windows to UNIX)*:
<http://www.ibm.com/support/docview.wss?uid=swg21428743>
- ▶ White Paper - *What adjustments can I make to FileNet Content Engine (CE) Automatic Upgrade*:
<http://www.ibm.com/support/docview.wss?uid=swg21428407>
- ▶ Information Center - General troubleshooting:
http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.ecm.common.ic.doc/troubleshoot_main.htm
- ▶ IBM developerWorks article, “Things you can do now for quicker and more effective way of troubleshooting”:
http://www.ibm.com/developerworks/websphere/techjournal/0708_supauth/0708_supauth.html
- ▶ Information Center - Ports information:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.plannprepare.doc/p8pap058.htm>
- ▶ Information Center - Directory service requirements information:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.security.doc%2Fp8psd000.htm>
- ▶ Information Center - Exception and logging concepts:
 - http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.ce.dev.ce.doc/exception_concepts.htm
 - http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.ce.dev.ce.doc/logging_concepts.htm
- ▶ You can obtain technical notices from the previous product documentation website (in the Technical Notices section):
 - IBM FileNet P8 Performance Tuning Guide
 - IBM FileNet P8 High Availability Technical Notice
 - IBM FileNet Content Engine Query Performance Optimization Guidelines Technical Notice

- IBM FileNet Application Engine Files and Registry Keys Technical Notice
- IBM FileNet P8 Asynchronous Rules Technical Notice
- IBM FileNet Content Engine Component Security Technical Notice
- IBM FileNet P8 Directory Service Migration Guide
- IBM FileNet P8 Disaster Recovery Technical Notice
- IBM FileNet P8 Extensible Authentication Guide
- IBM FileNet P8 Process Task Manager Advanced Usage Technical Notice
- IBM FileNet P8 Recommendations for Handling Large Numbers of Folders and Objects Technical Notice
- IBM FileNet P8 DB2 Large Object (LOB) Data Type Conversion Procedure Technical Notice

Although several technical notices were written for the 3.5 version, much of the content provided is useful for Version 4.0 as well.

- ▶ Administering Content Platform Engine – “Sharing Data Sources” and “Creating a Database Connection” topics:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/index.jsp?topic=%2Fcom.ibm.p8.ce.admin.tasks.doc%2Fp8pcb027.htm>
- ▶ Content storage management and storage farming IBM developerWorks article:
<http://www.ibm.com/developerworks/data/library/techarticle/dm-1003filenetstoragemanagement/index.html>
- ▶ Inheritance proxies and various ways of getting this done:
<http://www.ibm.com/support/docview.wss?uid=swg21425080>
- ▶ *IBM FileNet P8 Performance Tuning Guide* - Provides information about tuning parameters that can help improve the performance of your IBM FileNet P8 system:
ftp://ftp.software.ibm.com/software/data/cm/filenet/docs/p8doc/50x/p850_performance_tuning.pdf
- ▶ *IBM FileNet P8 Performance Tuning* - There are several pages that provide additional information for improving the performance of IBM FileNet P8 components:
<http://pic.dhe.ibm.com/infocenter/p8docs/v5r2m0/topic/com.ibm.p8.performance.doc/p8ppt000.htm>
- ▶ Proven Practice: IBM FileNet Deployment Manager 5.1 Data Migrations:
<http://www.ibm.com/support/docview.wss?uid=swg21609929>

- ▶ Migrating IBM Case Manager solutions using FileNet Deployment Manager and Case Manager Administration Client (is valid also for FileNet Content Manager only):
<http://www.ibm.com/support/docview.wss?uid=swg21612959>
- ▶ Impact of a FileNet Deployment Manager import using the “Update If Newer” option without the “Use Original Create/Update Timestamps” option:
<http://www.ibm.com/support/docview.wss?uid=swg21455363>
- ▶ FileNet Deployment Manager fails to import the documents of a given version series if one of those documents references an object that appears after the document in the deploy dataset (explanation of object hierarchy):
<http://www.ibm.com/support/docview.wss?uid=swg27020038>
- ▶ MustGather: FileNet Deployment Manager (FDM):
<http://www.ibm.com/support/docview.wss?uid=swg21502186>
- ▶ IBM System Dashboard guide:
<http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss?CTY=US&FNC=SRX&PBL=SC19-3084-03>
- ▶ IBM System Dashboard client API information:
<http://www.ibm.com/support/docview.wss?uid=swg21502836>
- ▶ IBM System Dashboard Usage Reporter:
<http://publibfp.dhe.ibm.com/epubs/pdf/c1930850.pdf>
- ▶ IBM ECM System Monitor information:
<http://www.ibm.com/software/products/us/en/ecmsystemmonitor>
<http://www.ibm.com/support/docview.wss?uid=swg27010374>
- ▶ IBM Case Manager product information:
<http://www.ibm.com/software/advanced-case-management/case-manager>
- ▶ IBM Rational product line information:
<http://www.ibm.com/software/rational>
- ▶ IBM Connections for social media information:
<http://www.ibm.com/software/lotus/products/connections>
- ▶ IBM InfoSphere Optim Data Lifecycle Management Solutions information:
<http://www.ibm.com/software/data/optim>

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services



Redbooks

IBM FileNet Content Manager Implementation Best Practices and Recommendations

(1.0" spine)
0.875" <-> 1.498"
460 <-> 788 pages



IBM FileNet Content Manager Implementation Best Practices and Recommendations



**System architecture,
business continuity,
and capacity
planning**

IBM FileNet Content Manager Version 5.2 provides full content lifecycle and extensive document management capabilities for digital content. IBM FileNet Content Manager is tightly integrated with the family of IBM FileNet products based on the IBM FileNet P8 technical platform. IBM FileNet Content Manager serves as the core content management, security management, and storage management engine for the products.

**Repository, security,
application design,
and solution building**

This IBM Redbooks publication covers the implementation best practices and recommendations for solutions that use IBM FileNet Content Manager. It introduces the functions and features of IBM FileNet Content Manager, common use cases of the product, and a design methodology that provides implementation guidance from requirements analysis through production use of the solution. We address administrative topics of an IBM FileNet Content Manager solution, including deployment, system administration and maintenance, and troubleshooting.

**Deployment, system
administration, and
maintenance**

Implementation topics include system architecture design with various options for scaling an IBM FileNet Content Manager system, capacity planning, and design of repository design logical structure, security practices, and application design. An important implementation topic is business continuity. We define business continuity, high availability, and disaster recovery concepts and describe options for those when implementing IBM FileNet Content Manager solutions.

Many solutions are essentially a combination of information input (ingestion), storage, information processing, and presentation and delivery. We discuss some solution building blocks that designers can combine to build an IBM FileNet Content Manager solution.

This book is intended to be used in conjunction with product manuals and online help to provide guidance to architects and designers about implementing IBM FileNet Content Manager solutions. Many of the features and practices described in the book also apply to previous versions of IBM FileNet Content Manager.

**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**