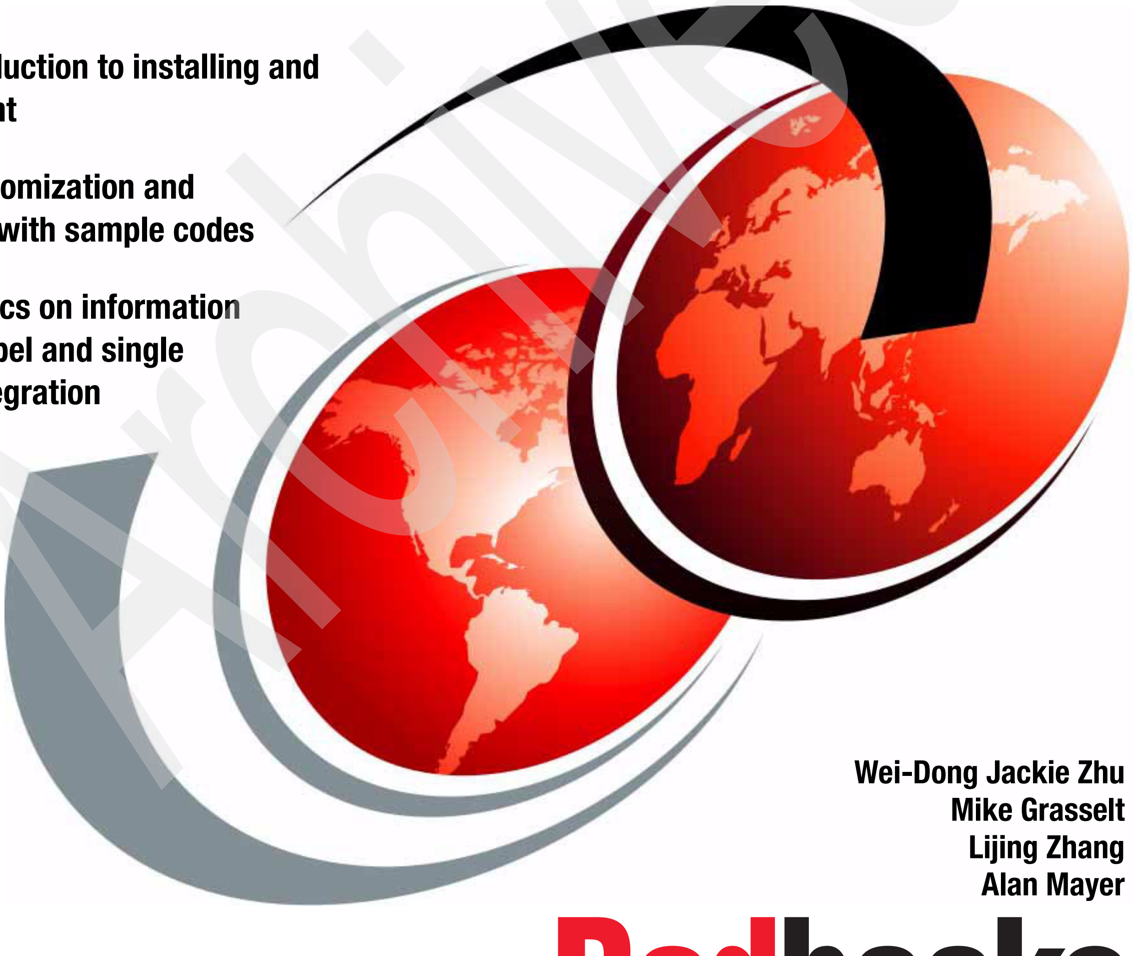
DB2. Information Management Software

IBM

eClient 101
Customization and Integration



Basic introduction to installing and using eClient

eClient customization and integration with sample codes

Special topics on information mining, Siebel and single sign-on integration

Wei-Dong Jackie Zhu
Mike Grasselt
Lijing Zhang
Alan Mayer

Redbooks

ibm.com/redbooks

International Technical Support Organization

eClient 101 Customization and Integration

October 2003

SG24-6964-00

Note: Before using this information and the product it supports, read the information in "Notices" on page xi.

First Edition (October 2003)

This edition applies to Version 8, Release 2 of IBM DB2 Content Manager for Multiplatforms (product number 5724-B19) and Version 8 Release 2 of IBM DB2 Information Integrator for Content for Multiplatforms (product number 5724-B43).

Contents

Notices

Trademarks

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

Redbooks (logo) ™	IBM®	SecureWay®
ibm.com®	Lotus Notes®	SP2®
iSeries™	Lotus®	ThinkPad®
AIX®	Micro Channel®	Tivoli®
ClearCase®	OS/390®	VideoCharger™
DB2 Universal Database™	PAL®	WebSphere®
DB2®	QBIC®	
ImagePlus®	Redbooks™	

The following terms are trademarks of other companies:

Pentium is a trademark of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

Preface

This IBM Redbook provides a basic introduction to IBM DB2® Content Manager Version 8 eClient. By providing helpful, easy-to-understand sample codes and step-by-step instructions, this redbook will help you in your next eClient integration and customization project.

In Part 1, we introduce the Content Manager family of products, which includes Content Manager, Information Integrator for Content (formerly known as EIP - the term is still used throughout this redbook for easy reference), Information Mining Service of EIP, and eClient. We also provide detailed step-by-step instructions on installing eClient, installing eClient in a WebSphere® Network Deployment environment, and using eClient.

In Part 2, to prepare for eClient customization and integration, we introduce J2EE, servlets, and JSPs. We cover the eClient architecture and inspect a basic eClient control flow. We also provide the essential information required to start creating applications with EIP. We include sample codes to demonstrate the usages of APIs. In addition, we provide step-by-step instructions on setting up an eClient development environment and discuss design and implementation considerations.

Part 3 is about customizing eClient. We provide sample codes for changing the look and feel, customizing the edit attributes window, adding customized functions in the search results window, and using EIP privileges for access control. By demonstrating some of the customization with the sample codes, you should have a better understanding of what and how you can customize eClient for your business needs.

Part 4 covers integrating eClient. We cover e-mail integration and special topics on Information Mining Service, Siebel integration, and single sign-on integration. The installation, setup, configuration, and integration are presented with detailed step-by-step instructions. In the Information Mining Service chapter, we provide detailed sample codes for enabling metadata-based data retrieval. Even if you are not using Information Mining Service, we highly recommend that you read the chapter for more in-depth knowledge of eClient customization from the sample codes.

In Part 5, to assist in troubleshooting and debugging eClient applications, we provide tips and recommendations on how to troubleshoot problems, with a list of typical problems and their resolutions. In addition, we give a brief introduction on performance tuning for eClient.

The team that wrote this redbook

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

Wei-Dong Jackie Zhu is a Project Leader in Content Management for the International Technical Support Organization at the Almaden Research Center in San Jose, California. She has more than 10 years of software development experience in document search systems, image workflow processing, and digital media distribution. She holds a Master of Science degree in Computer Science from the University of Southern California.

Mike Grasselt has been with the IBM Germany Development Lab in Boeblingen since 1997 working on OS/390®, Java projects, and education. He is a technical lead for the Information Mining feature of Information Integrator for Content.

Lijing Zhang is a software engineer in IBM US. He has worked for IBM since 2001 in the Content Management field. His areas of expertise include Content Manager, Information Integrator for Content, and VideoCharger™. He is a co-author of *Content Manager V8 Certification Study Guide*.

Alan Mayer is an I/T Architect in IBM Global Services in US. He has 10 years of experience implementing Content Manager for various clients. His areas of expertise include application development and document imaging.

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

Andy Smith
David B Victor
Ken Nelson
Jerald Schoudt
Shailesh Gupta
Charlie Jin
Gordon Campell
Randy Richardt
An Phan
Mel Zimowski
Michael J. Mitchell
Fay Wong
IBM Software Development, Support, and Services

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

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

`ibm.com/redbooks/residencies.html`

Comments welcome

Your comments are important to us!

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

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

  `ibm.com/redbooks`

- Send your comments in an Internet note to:

  `redbook@us.ibm.com`

- Mail your comments to:

  IBM® Corporation, International Technical Support Organization
  Dept. QXXE Building 80-E2
  650 Harry Road
  San Jose, California 95120-6099

Part 1

Getting started with eClient

This part helps you to get started with eClient. We introduce the overall Content Manager family of products, and provide detailed step-by-step instructions on how to install eClient, how to install eClient in a WebSphere Network Deployment environment, and how to use eClient.

1

Introducing Content Manager

This chapter introduces the Content Manager product, which includes the following components:

- IBM DB2 Content Manager Version 8 (Content Manager)
- IBM DB2 Information Integrator for Content Version 8 (also known as EIP)
- Information Mining Service of EIP
- IBM DB2 Content Manager Version 8 eClient

1.1 Introduction

The Content Manager portfolio provides a foundation for managing, accessing and integrating critical business information on demand. It enables you to integrate all types of content - document, image, rich media - across diverse business processes and applications, including Siebel, PeopleSoft and SAP.

Content Manager Version 8 integrates with existing hardware and software investments, enabling customers to leverage a common infrastructure, achieve a lower cost of ownership, and deliver new, powerful information and services to customers, partners, and employees where and when needed.

The IBM DB2 Content Manager Version 8 portfolio includes the following main product categories:

- IBM DB2 Content Manager Version 8 (Content Manager)
- IBM DB2 Information Integrator for Content Version 8 (Information Integrator for Content)
- IBM DB2 Content Manager Version 8 OnDemand
- IBM DB2 Content Manager Version 8 CommonStore
- IBM DB2 Records Manager

In this redbook, we focus on the IBM DB2 Content Manager Version 8 eClient, the Web client of IBM DB2 Content Manager Version 8 for Multiplatforms. Before we go into the details of eClient, we provide a quick overview of Content Manager, EIP, Information Mining Service of EIP, and Content Manager eClient.

1.2 Content Manager Version 8

IBM DB2 Content Manager Version 8 (Content Manager) is the centerpiece of the IBM DB2 Content Manager portfolio for Enterprise Content Management. It is used to store and manage unstructured data.

A Content Manager system mainly consists of a single Library Server and one or more Resource Managers, formerly known as the Object Server.

The Library Server is essentially a database application with stored procedures that manages user profiles, access controls, document routing rules and data model definition. The Library Server stores and manages index information (metadata) for the unstructured data stored in the Resource Managers. There is only one Library Server for a Content Manager system.

There may be one or more Resource Managers in a Content Manager system. This is where the unstructured data objects are physically stored. Resource Manager is essentially a Web application running on IBM WebSphere Application Server. Figure 1-1 shows how the Library Server and Resource Managers build the infrastructure of a Content Manager system.

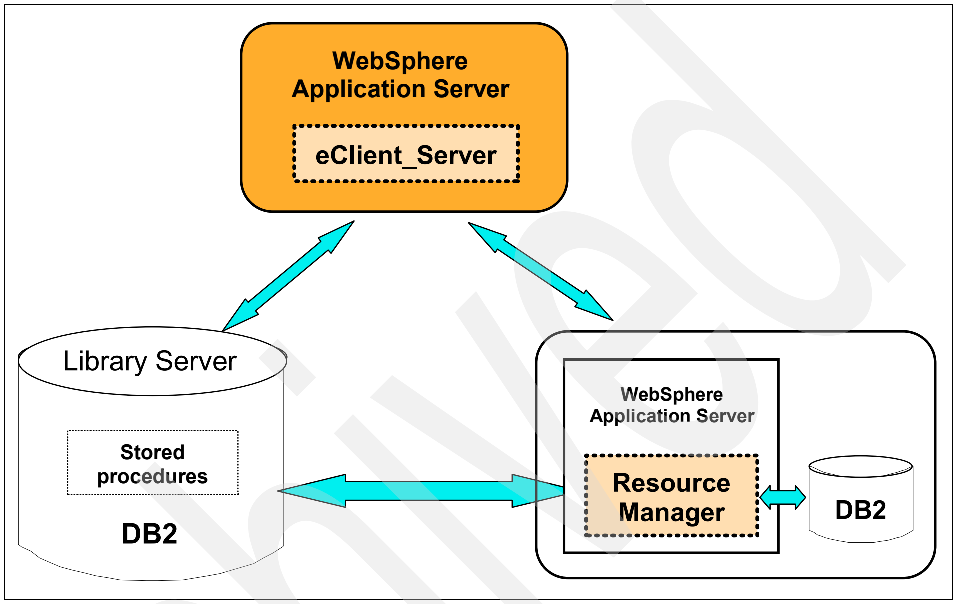


Figure 1-1 Content Manager infrastructure

If you have video and audio files and want to have streaming capability, you may choose to store them in a Content Manager VideoCharger server. While it is integrated with Content Manager server, the VideoCharger server is considered an extension of Resource Manager.

There are three types of Content Manager clients that can be used to manage and retrieve documents stored in Content Manager server: eClient, window client, and customized client. The focus of this redbook is on eClient.

To demonstrate how a Content Manager system works, let us examine the steps that need to be taken by the Library Server, the Resource Manager and the eClient to perform a search for a document and the delivery of the document for an end user.

1. Using a Web browser, a user tries to log on to an eClient server by submitting a user ID and a password.
2. The Library Server receives the logon request and validates the user information.

3. If the user is authorized to access the system, the Library Server grants the right and the user logs on to the eClient server.
4. The user searches for some documents; this translates to a search request to the eClient.
5. The Library Server receives the request and performs a search. The document access control list and user's privilege set are evaluated to determine if the user has permission to access the documents.
6. The Library Server returns a hit list from the search. The hit list only contains the documents that the user has access to.
7. The user clicks one of the result entries in the eClient; this translates to a document retrieval request.
8. The Library Server receives the retrieval request. It returns a security token and the Resource Manager location information (where the document resides) to the eClient. A unique token is dynamically generated by the Library Server for each client request and will remain valid for a set length of time.

   Up to this point, the communications are between the eClient and the Library Server only. Resource Manager has not been in the picture yet.
9. eClient receives the security token from the Library Server and the information about where the document is located. It uses the security token to directly contact the Resource Manager where the document is stored.
10. The Resource Manager receives the retrieval request from the eClient. It validates the security token. If the security token is validated, the Resource Manager directly delivers the documents to the eClient.

    The document does not flow through the Library Server.
11. The document displays in the Web browser; the user views the document.

1.3 Information Integrator for Content Version 8

IBM DB2 Information Integrator for Content Version 8 (Information Integrator for Content), formerly Enterprise Information Portal, provides a single point of access to unstructured and structured content stored on one or more content servers.

Information Integrator for Content is a framework that consists of two parts:

- Information access
- Services

In the following section, we briefly describe these two parts. For a complete list of features, refer to Chapter 1 in *Managing Information Integrator for Content*, SC27-1346-01.

1.3.1 Information access

To access the information stored in content servers, an application can either use one of the content server connectors or a federated connector, as illustrated Figure 1-2 on page 8.

Content server connectors provide the communication interface among the applications, the content servers, and the administration database. A connector can be implemented for arbitrary content servers. Information Integrator for Content Version 8 provides connector implementations for content servers such as DB2 or Content Manager Version 8.

The *federated connector* has the same interface as all other connectors but it does not have a physical store and it is configured with any number of supported connectors to become the single point of access for multiple content servers. You can use it to search, retrieve, and update data objects in the content servers; however, you need to call the content server connector directly to create and delete data objects.

To allow federated access, a *schema mapping* between the federated content server and each participating content server is required. The schema mapping handles the difference between how the data is physically stored and how the user wants to process the data in an application. This applies to attributes as well as user IDs and passwords. In an application, the persistent data objects are represented by the Dynamic Data Objects (DDO). A DDO is a server-neutral and self-describing data object for transferring data into and out of a content server. A DDO has a single persistent ID (PID), an object type, and a set of data items.

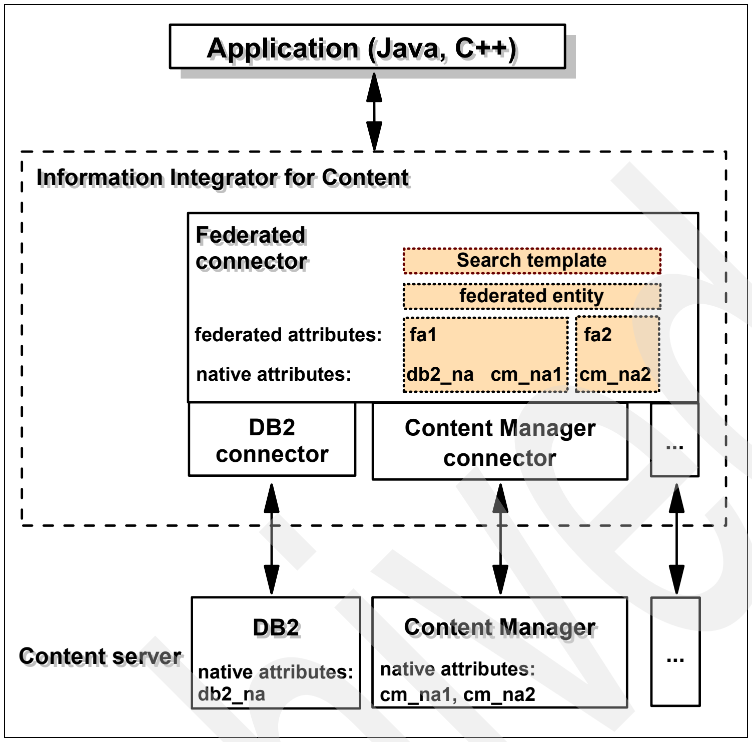


Figure 1-2 Information access overview

Figure 1-2 illustrates the information access using connectors and the mapping of native attributes to the federated attributes. Federated attributes can be grouped to a federated entity and then be used to define a search template. Defining search templates is a convenient way to predefine queries and control the access to those queries. To search, you simply retrieve a search template from the administration database, input the search values for the federated attributes and perform a search. If the template contains federated attributes that are mapped to the native attributes of different content servers, the search runs simultaneously across these servers.

The programming interface is available for C++ and Java. Java applications can also use the client/server implementation, which is based on Remote Method Invocation (RMI).

1.3.2 Services

Information Integrator for Content services provide added values for information access and programming interfaces that are aligned with the information access interfaces.

Currently Information Integrator for Content provides two services:

- **Information Mining Service**

  This service allows you to automatically analyze and organize documents on content servers. Because nearly 80 percent of your business data is unstructured, you cannot do this manually. Information Mining provides tools such as automatic categorization, summarization and information extraction. If the analysis results get stored together with the original document, you can, for example, restrict searches to certain categories and display a summary for each search result. You can find a description of the Information Mining service features in 1.4, "Information Mining Service" on page 9.

- **Workflow**

  You can use the workflow service to control the flow and performance of work in your business. When users work with the results of federated searches, they often must make decisions on what actions to perform. You can determine in advance how you want users to perform the work. The actual documents can reside on any of the supported content servers. You can automate the workflow by setting up profiles and rules.

1.4 Information Mining Service

Industry studies show that employees, in general, spend 30 percent of their time just looking for the information they need to do their jobs. What is more, most of this information is unstructured and is buried within reports, e-mail, mail, or faxes. One way to use these documents efficiently is to organize the information within the documents and create metadata for documents. The metadata of a document includes names, institutions, or places mentioned in a document or category a document belongs to. Users can use the metadata to narrow their searches of documents to certain topics or terms.

The Information Mining Service of Information Integrator for Content provides text analysis components to extract metadata automatically, thus making mining economically viable. The complete interpretation of only factual knowledge stated in unrestricted natural language is still out of reach using current technology. However, tools that apply pattern recognition techniques and heuristics are capable of extracting valuable information from arbitrary free-text. Extracting information ranges from identifying important terms, such as names, institutions, or places mentioned in a document, to summarizing a document.

In the following section, we briefly describe the features, concepts and architecture of the Information Mining Service. For a detailed description, go to the Information Center and select **Enterprise Information Portal -> Administration -> Managing information mining**.

1.4.1 Features

The Information Mining Service provides the following mechanisms for the automatic creation of metadata:

- **Categorization** assigns one or more categories to a document based on a user-defined taxonomy (category tree). Users can store topic information along with documents. This helps users to organize document collection and allow topic-based search and navigation. The categorization component contains a Web application that provides a graphical user interface for the creation and maintenance of taxonomies called the Information Structuring Tool.
- **Summarization** extracts the most important sentences of a document. Users can read this information before deciding whether to read the entire document or not. Users can specify the length of the intended summary and directly influence the balance between the complexity of the extracted metadata and the amount of information in documents.
- **Language identification** determines the language a document is written in. This is useful if the document processing needs to be restricted to documents of a certain language. It is a required pre-processing step before applying other services.
- **Information extraction** recognizes significant vocabulary items, such as names, terms, and expressions, in text documents automatically. The extracted terms can, for example, be used in automatically created queries to find related documents.
- **Clustering** divides up a set of documents into similar groups or clusters. This is another way to organize document collection but unlike categorization, the structure is not predefined. Clusters are derived from the document collection automatically.

With the provided infrastructure, documents returned by a federated search or the IBM Web Crawler can directly be accessed and analyzed. You can also write applications that analyze documents from any arbitrary source.

The contained document filter extracts the textual content from a wide range of document formats, such as PDF and HTML; this is a required pre-processing step before running any of the analyses mentioned earlier.

In addition, the service can store the created metadata in the Information Integrator for Content (EIP) database, but it is always a good practice to store this information along with the document in the content server. This way, you can use the metadata in a federated search, for example to search for documents that belong to a certain category.

1.4.2 Concepts

The library and its catalogs represent the main concepts in the Information Mining Service.

The library is a conceptual view of the Information Mining related content of the EIP database. The library contains a set of catalogs.

A catalog is a container for:

- A taxonomy, which is a hierarchical tree structure of categories.
- A collection of training documents that is assigned to the categories to define typical content for these categories.
- A categorization model based on the document training results that can be used to automatically assign categories to documents. This model is generated using the Information Structuring Tool where a taxonomy can be created and trained. The model serves as input to the categorization service.
- The document metadata if you decide to store it in the EIP database.

1.4.3 Architecture

Figure 1-3 on page 12 provides an overview of the Information Mining Service architecture.

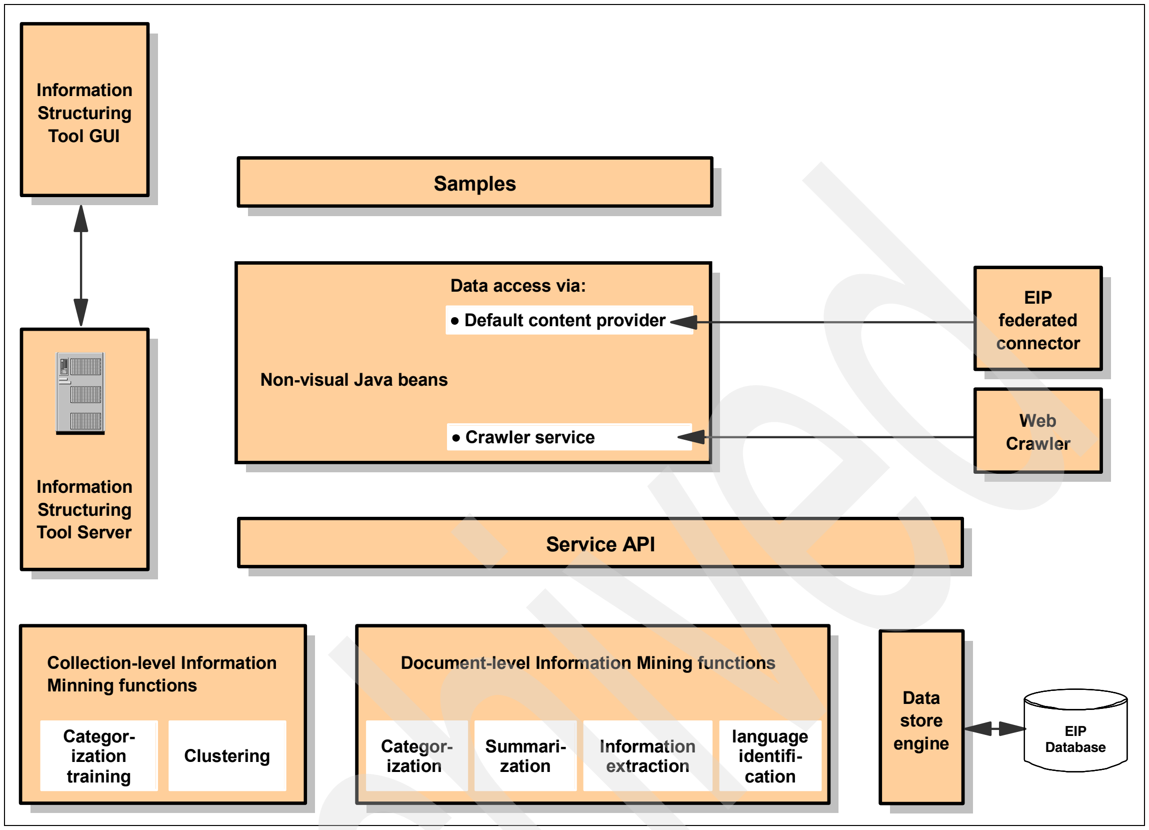


Figure 1-3 Information Mining Service architecture

The following layers make up the Information Mining Service architecture:

- **Java Service API**: This layer exposes the Information Mining functionality and metadata persistency as a consistent Java API. It provides access to the analysis functions such as clustering, categorization and summarization. The categorization training is only available with the Information Structuring Tool. The API also allows you to store metadata in the EIP database.
- **Non-visual JavaBeans**: This layer provides ready-to-use components based on the JavaBeans specification. The beans can be connected to the information access beans in EIP, for example to analyze the documents returned by a search with the federated connector. The beans also support the documents returned by the IBM Web Crawler of EIP.
- **Samples:** This level consists of sample code using the non-visual JavaBeans that illustrates the usage of the JavaBeans API.

- **Information Structuring Tool**: A Web application for creating and maintaining taxonomies.

For detailed scenarios on how to use the Information Mining Service, refer to Chapter 13, “Enabling metadata-based content retrieval” on page 281.

1.5 Content Manager Version 8 eClient

The Content Manager Version 8 eClient (eClient) is a Web application that allows users to manage documents, search for documents and retrieve documents from content servers. The following are a few sample content servers that the eClient can have access to:

- IBM Content Manager for Multiplatforms
- IBM Content Manager OnDemand
- IBM Content Manager ImagePlus® for OS/390

The eClient application server is deployed and runs in WebSphere Application Server. It consists of JavaServer Pages (JSPs), servlets, a viewer applet, Cascading Style Sheets, property files, etc. The eClient is built on top of the EIP Java APIs. You can customize the eClient to meet the needs of your organization. For details on the eClient architecture, see Chapter 5, “eClient architecture” on page 101.

With the eClient, you can connect to the Information Integrator for Content federated server, which allows you to perform searches across a variety of content servers simultaneously. You can also choose to connect directly to individual content server. Figure 1-4 on page 14 shows the eClient infrastructure.

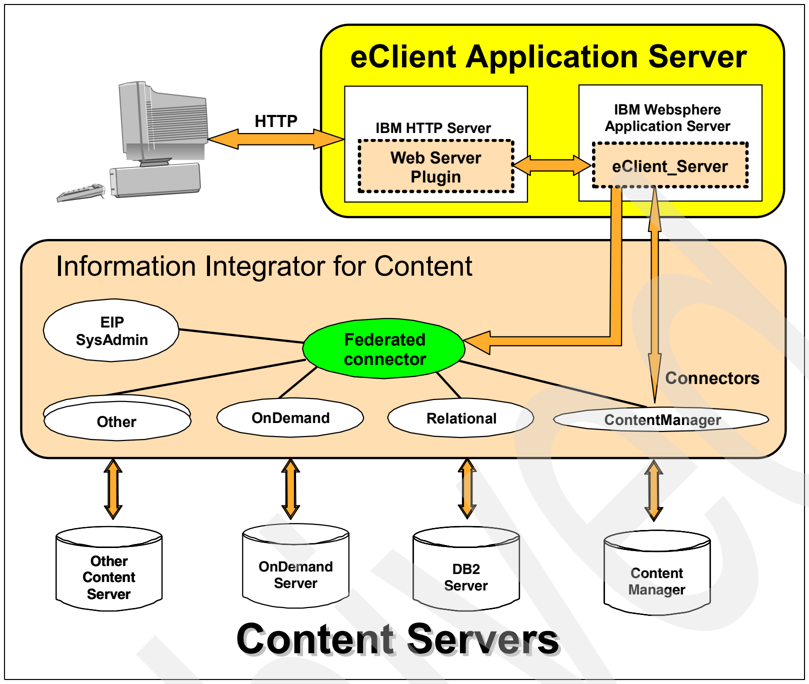


Figure 1-4 eClient infrastructure diagram

The eClient Version 8 provides many new features and significant enhancements over the previous versions. Prior to Version 8.1, users primarily use the eClient to perform document searches and retrievals. Since Version 8.1, eClient offers document manipulation features. These include:

- Importing and deleting documents
- Check-in and check-out support for documents
- Re-indexing documents
- Changing document attributes
- Folder operation (for example, creating folders, adding documents to folders, removing documents from folders)
- e-clipboard to support folder operations
- Versioning support
- Support for Content Manager document and folder note logs

All of the above features are available when connecting to Content Manager Version 8 servers. When connecting to other content servers or previous versions of Content Manager servers, only some of the mentioned features may be available.

Direct object retrieving is another key feature introduced in eClient Version 8. With this retrieving method, the objects are retrieved directly to the end user from Content Manager Version 8 servers. It bypasses the eClient server when retrieving objects and therefore enhances performance. To take advantage of the this great feature, you must:

- Set directRetrieveEnabled=true in the IDM.properties file to enable it.
- Use either **`launch`** or viewer applet for the MIME type in the IDMadminDefaults.properties file.

Viewer applet is another important new feature introduced in eClient Version 8. It is built on top of EIP Java Viewer Toolkit. When you retrieve a document from a content server and choose to display it in a viewer applet, you can manipulate annotations attached to the document.

eClient Version 8 also supports the Advanced Workflow in EIP Version 8 and the document routing feature in Content Manager Version 8. When connecting to Content Manager Version 8 servers, you can also usethe WebSphere Single Sign-on feature.

The federated folder concept is introduced in eClient Version 8.2. A federated folder is similar to a folder in Content Manager. It may contain documents and folders. Entities contained in the federated folders could be from native content servers or from other federated folders.

Other new features and feature enhancements are provided in eClient Version 8.2. One important enhancement is the eClient performance, which has been improved dramatically.

2

Installing eClient

This chapter provides detailed procedures for installing, configuring, and verifying IBM DB2 Content Manager Version 8 eClient, and IBM DB2 Information Integrator for Content Version 8 (EIP) on the Windows platform. Before starting your installation, first read Chapter 1, “Introducing Content Manager” on page 3 and optionally read Chapter 5, “eClient architecture” on page 101. They will help you understand the Content Manager product portfolio and the eClient architecture for a successful installation.

This chapter covers the following topics:

- Installation overview
- Installing Information Integrator for Content Version 8 (EIP)
- Installing and configuring the Information Structuring Tool on WebSphere Application Server
- Installing Content Manager Version 8 eClient

Note: The intent of this chapter is to lead you through a simple, first time installation. For a more complex installation scenario, read Chapter 3, “Installing eClient in a WebSphere Network Deployment environment” on page 51.

2.1 Installation overview

In this installation overview section, we briefly discuss installation options for IBM DB2 Content Manager Version 8 (Content Manager), IBM DB2 Information Integrator for Content Version 8 (Information Integrator for Content, also known as EIP) and IBM DB2 Content Manager Version 8 eClient (eClient), and the eClient prerequisites.

We used the following products while preparing this chapter:

- Windows 2000 Server + Service Pack 3
- DB2 Universal Database™ V8.0 + Fix Pack 1
- WebSphere Application Server V5.0
- IBM HTTP Web Server V3.26
- Content Manager V8.2
- EIP V8.2
- eClient V8.2

2.1.1 Topology

There are many ways to configure Content Manager, EIP, and the eClient system.

For a development environment, you may install all three products on the same machine. Developers can have full control of their own system.

In a production environment, you may consider installing each product on a separate machine. In this scenario, EIP connectors (either local or remote connector) must be installed on the eClient machine.

To achieve the best performance, you may have a cluster of eClient servers. In this setup, multiple eClient application servers run on multiple machines in a WebSphere Application Server Network Deployment environment. This configuration provides workload management that avoids a single point of failure. For information on installing eClient in a WebSphere Application Server Network Deployment environment, read Chapter 3, “Installing eClient in a WebSphere Network Deployment environment” on page 51.

2.1.2 Prerequisites

Since eClient is a Web application, you must install WebSphere Application Server on the same machine as eClient. A Web server is essential to run eClient on the Internet or intranet. It is not necessary to install the Web server on the same machine where eClient is installed.

In order to connect to different back-end servers such as Content Manager, EIP connectors (and minimal EIP components) must be installed.

You can find a complete list of prerequisites for eClient installation in Chapter 2, “Requirements” in *IBM Content Manager for Multiplatforms/IBM Information Integrator for Content: Installing, Configuring, and Managing eClient*, SC27-1350.

For our scenario, we have installed DB2 V8.1, DB2 TIE V8.1, WebSphere Application Server V5.1, HTTP Web Server V3.26 and Content Manager server V8.2 on the same machine.

2.2 Installing Information Integrator for Content Version 8 (EIP)

This section provides instructions for installing, configuring and verifying Information Integrator for Content Version 8 (EIP) on a Windows platform. We assume you have already installed Content Manager Version 8 on your machine. If you have not done so, please refer to *IBM Content Manager for Multiplatforms: Planning and Installing Your Content Manager System*, GC27-1332 for more details.

2.2.1 Hardware and software requirements

When you follow the procedures provided in the following section to install EIP on Windows, your should have the appropriate hardware and software.

Basic hardware requirement

The following is a list of basic hardware requirements for your EIP installation. In order to follow our scenario, we assume you will install everything in one machine.

- Processor: Intel Pentium® 800 MHz or equivalent
- RAM: minimum 512 MB, 1024 MB recommended
- Disk storage: 1 GB swap space, 400 MB install space, 10 MB temporary space

Basic software requirement

The following is a list of basic software requirements for your EIP installation. In order to follow our scenario, we assume you will install everything in one machine.

- Operating system: Microsoft® Windows NT 4.0 Server with Service Pack 6 or later, Windows 2000 Server, or Windows XP
- Network communication: TCP/IP installed with Windows

- IBM DB2 Universal Database Version 8.1 with DB2 Application Development Client
- Microsoft Visual C++ Version 6.0
- Java Development Kit, Version 1.3
- JDBC driver 1.3 (Java only)
- ODBC 3.0 (C++ only)
- Internet Explorer Version 5.0 or later

Software prerequisites verification

Use Table 2-1 as a quick reference to verify that your machine has the prerequisite software and the appropriate versions.

Table 2-1 Basic software prerequisite verification

Prerequisite	How to check	Example value
1) Windows NT Service Pack 6 2) Windows 2000 Server Service Pack 2	**`Winver`**	1) Version 4.0 (Build 1381: Service Pack 6) 2) Version 5.0 (Build 2195: Service Pack 2)
Java Development Kit V1.3	**`java -fullversion`**	Version needs to read 1.3.1 (for example, if you are using the version from WebSphere Application Server, it will read: `java full version “J2RE 1.3.1 IBM Windows 32 build cn131w-20020403 ORB130“`).
UDB EE V7.2 with Fix Pack 7 or higher	From the DB2 command window: **`db2level`**	Level needs to read “SQL07025” or greater with Fix Pack level of “WR21306” or greater.
DB2 UDB Enterprise Server Edition Version 8.1 with Fix Pack 1	From the DB2 command window: **`db2level`**	Level needs to read 1 SQL08010 or DB2 V8.1.1.27. The Fix Pack information needs to read “FixPak 1” and list the Fix Pack level (for example, “s021124” is the Fix Pack that had been available since November 24, 2002).
DB2 Text Information Extender with Fix Pack 1	From the DB2 command prompt: **`db2text start`**	1. CTE0185 2. CTE0001 Operation completed successfully

Prerequisite	How to check	Example value
Net Search Extender (required if you use DB2 Version 8.1)	From the DB2 command window, start the text search program: **`db2text start`** Then type: **`db2textlevel`**	CTE0350 Instance “DB2” uses DB2 Net Search Extender code release “tx9_8” with level identifier “ tx9_26a”
Tivoli® Storage Manager API Client Version 4.2.1	**`c:\tsm\api\samprun\dapismp`**	API Library Version = 4.2.1.0
Tivoli Storage Manager Server Version 4.2.1	Log on to the TSM server administration Web page `http://<hostname>:1580`, Where `<hostname>` is the name of the TSM server.	The version appears on the Web page. It should say Version 4, Release 2, Level 1.0
1) WebSphere Application Server AE 4.0.3 2) WebSphere Application Server AES 4.0.3	Check the product.xml file located in x:\WebSphere \AppServer \propers \com \ibm\websphere.	<version>4.0.3</version>
Microsoft Visual C++ Compiler Version 6.0	Select **Start -> Programs**.	1) Microsoft Visual C++ 6.0 2) Microsoft Visual Studio 6.0
Microsoft Visual Studio .NET Professional	At the command line, type **`cl`**	`Microsoft 32-bit C/C++ Optimizing Compiler Version 13.00.94966 for 80x86 Copyright (C)Microsoft Corporation 1984-2001. All rights reserved.`

For installing and updating prerequisites, refer to Chapter 4, “Installing and updating prerequisite programs for Windows” in *IBM Content Manager for Multiplatforms: Planning and Installing Information Integrator for Content*, GC27-1345.

Important: EIP connectors are prerequisites of eClient. The eClient installation process will fail if they are not installed.

There are two different kind of connectors in EIP, local connector or remote connector. If you choose to install local connectors on the same machine where eClient will be installed, eClient can directly connect to a back-end server such as the Content Manager server. If you choose not to use the local connectors, you must install remote connectors. In this case, an RMI server will be required for connecting eClient to the Content Manager server. However, at the time of writing, RMI server is not available for Content Manager Version 8.

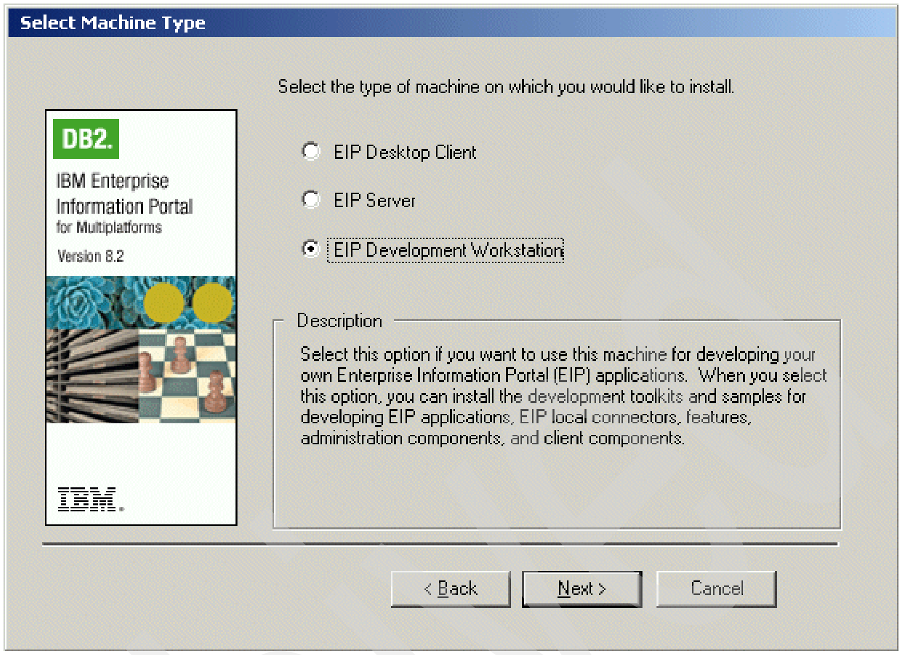
2.2.2 Installing EIP

Complete the following steps to install Information Integrator for Content Version 8 (EIP):

1. Log on to the machine as a system administrator user.

> **Note:** For the best practice, we recommend that you create a user, such as *admin*, and assign it to an administrator group. Then, you should log in as user *admin* to install all products, including DB2, WebSphere Application Server, Content Manager, EIP, and eClient.

2. Insert the EIP V8.2 CD into the CD-ROM drive.
3. On the EIP installation LaunchPad window, click **Install**.
4. Review the IBM software license agreement and click **Accept**.
5. On the next window, click **Next** to continue.
6. On the Select Machine Type window, select **Development Workstation** as shown in Figure 2-1 on page 23 and click **Next**.


Select Machine Type

Select the type of machine on which you would like to install.

( ) EIP Desktop Client
( ) EIP Server
[✓ EIP Development Workstation]

Description

Select this option if you want to use this machine for developing your own Enterprise Information Portal (EIP) applications. When you select this option, you can install the development toolkits and samples for developing EIP applications, EIP local connectors, features, administration components, and client components.

Figure 2-1 Select machine type

When you select the Development Workstation option, you will have all components to choose for installation. If you select the **Server** option, the Connector toolkit and sample components are removed from the Component Selection window. If you choose the **Desktop Client** option, the Administration, Connector toolkit and sample components are removed from Component Selection window. Also, the Information Mining server and IBM Web Crawler subcomponents are removed from Features components.

7. On the Select Destination window, enter the installation directory, in our scenario, c:\cmbroot, click **Next**.
8. On the Component Selection window, select the components and the subcomponents as shown in Table 2-2 and Figure 2-2 on page 24. Click **Next**.

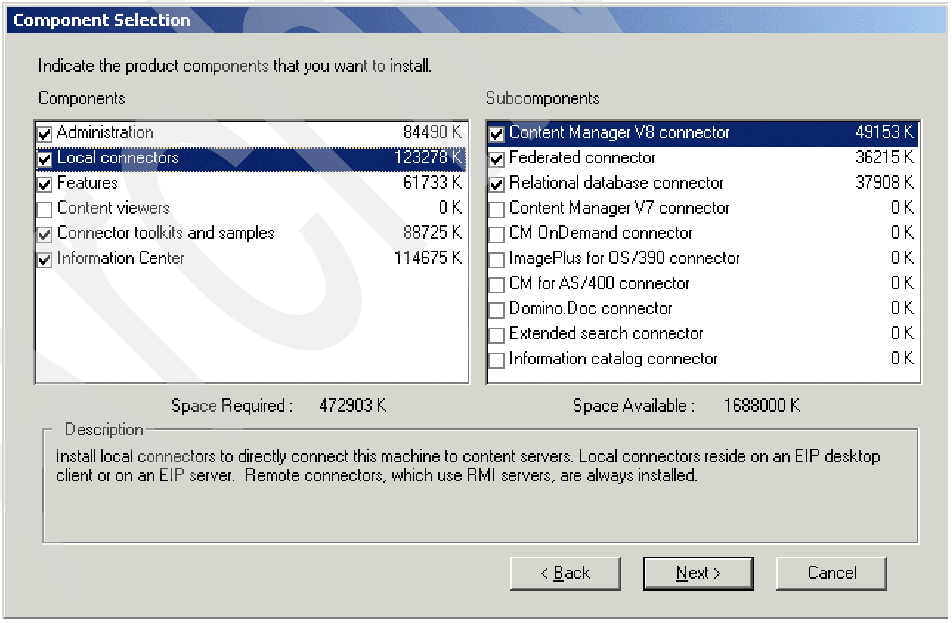
Table 2-2 Select components to install

Component	Subcomponent
Administration	Administration database Administration client

Component	Subcomponent
Local connectors	Content Manager V8 connector Federated connector Relational database connector
Feature	Information mining server
Connector toolkits and samples	Content Manager V8 connector Federated connector Relational database connector
Information center	

The Feature component is optional for EIP installation. We select it in this redbook because we will show you how to integrate Information Mining Service with eClient in a later chapter.

There are two subcomponents for Information Mining Service under the Feature component: the Information Mining Service server and the Information Mining Service client. The Information Mining Service server component includes the Information Mining Service client component.



Indicate the product components that you want to install.

Components

✓	Administration	84490 K
✓	Local connectors	123278 K
✓	Features	61733 K
	Content viewers	0 K
✓	Connector toolkits and samples	88725 K
✓	Information Center	114675 K

Subcomponents

✓	Content Manager V8 connector	49153 K
✓	Federated connector	36215 K
✓	Relational database connector	37908 K
	Content Manager V7 connector	0 K
	CM OnDemand connector	0 K
	ImagePlus for OS/390 connector	0 K
	CM for AS/400 connector	0 K
	Domino.Doc connector	0 K
	Extended search connector	0 K
	Information catalog connector	0 K

Space Required : 472903 K   Space Available : 1688000 K

Description

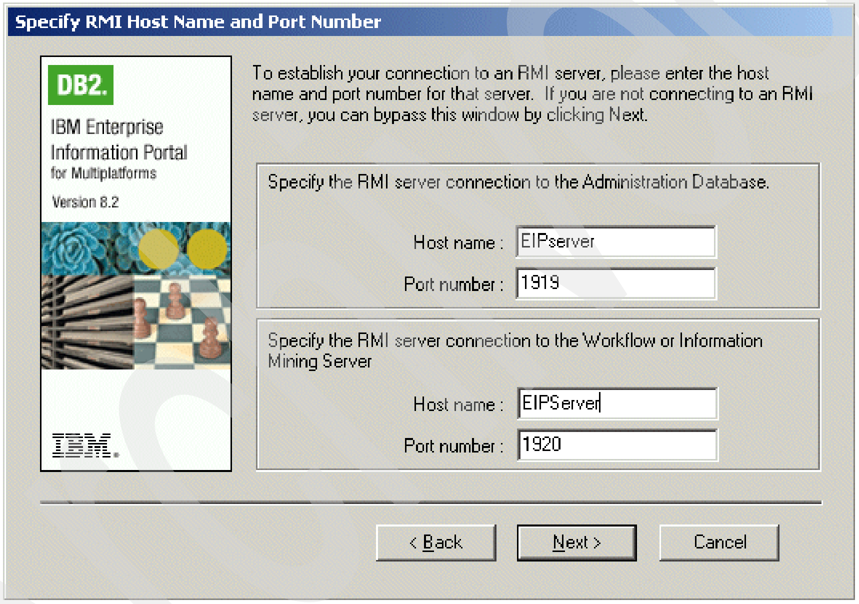
Install local connectors to directly connect this machine to content servers. Local connectors reside on an EIP desktop client or on an EIP server. Remote connectors, which use RMI servers, are always installed.

Figure 2-2 Select components to install

9. On the Specify RMI Host Name and Port Number window, specify the host name and port number as shown in Table 2-3 and Figure 2-3. Click **Next**.

Table 2-3 Specify RMI host name and port number

RMI type	Host name	Port number
IBM DB2 Information Integrator for Content Version 8 administration database	EIP_RMIserver (EIPserver for our scenario)	1919
Workflow and/or Information mining server	IM_RMIserver (EIPserver for our scenario)	1920


Specify RMI Host Name and Port Number

To establish your connection to an RMI server, please enter the host name and port number for that server. If you are not connecting to an RMI server, you can bypass this window by clicking Next.

Specify the RMI server connection to the Administration Database.

Host name : EIPserver

Port number : 1919

Specify the RMI server connection to the Workflow or Information Mining Server

Host name : EIPServer

Port number : 1920

Figure 2-3 Specify RMI host name and port number

There are two different RMI servers. One is used to connect to the EIP administration database to content servers. The other one is used to connect to the workflow and/or Information Mining Service server. The two RMI servers can be on different machines or on the same machine. If you decide to use local connectors to directly connect to the back-end server, RMI servers are not used.

The default port numbers are 1919 and 1920. Use the following commands to see a list of ports currently in use:

– Windows: **`netstat -an`**

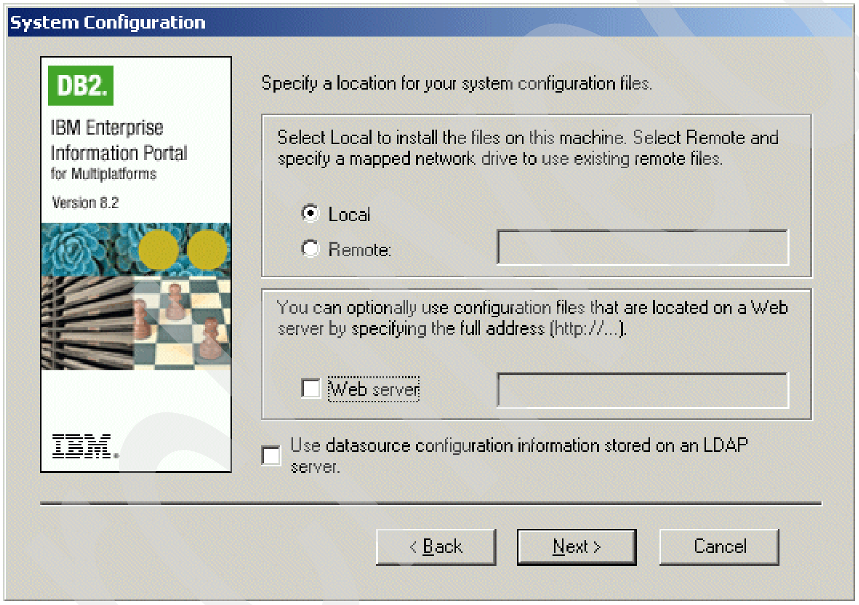
– AIX®, Linux, and Solaris: `netstat -an | grep LISTEN`

If you have conflicts with existing applications and services, you should change the port number. Otherwise, take the default value on the System Configuration window. Click **Next**.

10. Specify the location of your system configuration files on the System Configuration window. The default is Local, as shown in Figure 2-4.

    Note that the server configuration file can reside on the local machine, an HTTP server, or a remote server.

    For our scenario, we take the default location. Click **Next**.



Specify a location for your system configuration files.

Select Local to install the files on this machine. Select Remote and specify a mapped network drive to use existing remote files.

[✓ Local]

Remote:

You can optionally use configuration files that are located on a Web server by specifying the full address (http://...).

☐ Web server

☐ Use datasource configuration information stored on an LDAP server.

Figure 2-4 Specify the location for your system configuration

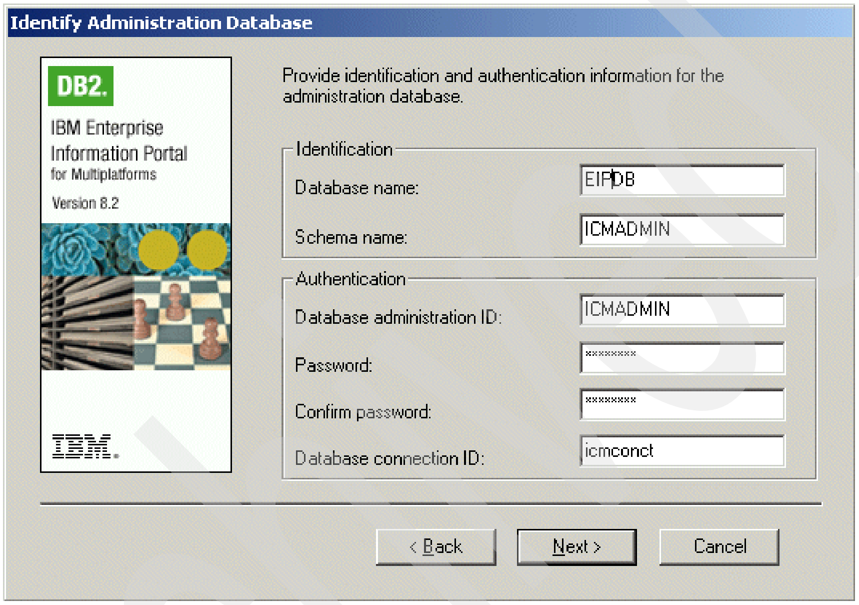
11. On the next window, you may configure to use LDAP. For our scenario, we do not configure the LDAP feature during installation. Click **Next**.

12. Enter the values in Table 2-4 on the Identify Administration Database window. See Figure 2-5 on page 27. Click **Next**.

Table 2-4 Define IBM DB2 Information Integrator for Content Version 8 database

Field	**Value**
Database name	EIPDB
Schema name	icmadmin

Field	Value
Database administration ID	icmadmin
Password	password
Database connection ID	icmconct


Identify Administration Database

Provide identification and authentication information for the administration database.

Identification

Database name: EIPDB

Schema name: ICMADMIN

Authentication

Database administration ID: ICMADMIN

Password: ********

Confirm password: ********

Database connection ID: icmconct

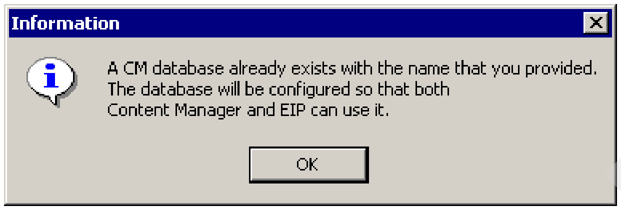
Figure 2-5 Identify EIP Administration Database

You enter EIP database information on this window. If you want to share the same database for EIP, Content Manager and Information Mining Service, you must enable unicode. It is required by Information Mining Service.

In a production environment, we recommend separated databases for EIP and Content Manager. In our scenario, the EIP database is EIPDB, and the Content Manager database is ICMNLSDB.

User icmadmin must be in the DBA group. By default, it is in the Windows administrator group. User icmconct is used to connect to the EIP database if a user is defined as an EIP user but not as a database user.

13. If you share the same database for Content Manager and EIP, you will receive the message shown in Figure 2-6 on page 28. Click **OK**.


Information

A CM database already exists with the name that you provided. The database will be configured so that both Content Manager and EIP can use it.

Figure 2-6 Warning message for database sharing

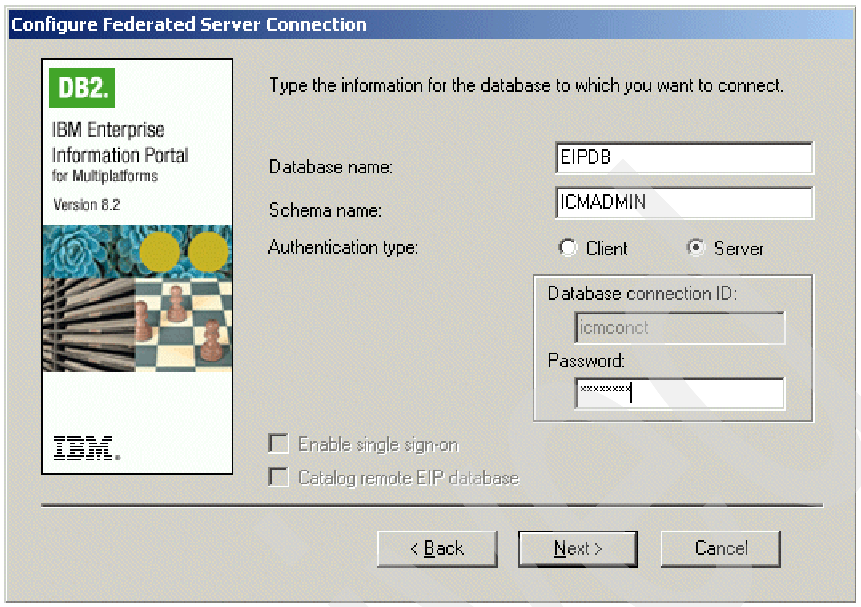
14. If you are creating a new database for EIP, you will see the Select Administration Database Options as the next window. Enter the location where you want to have the database (C drive or D drive) and click **Next**.

Note: Unicode is enabled and the field is grayed out because Information Mining Service feature is selected.

15. On the Configure Federated Server Connection window, enter the values in Table 2-5 and click **Next**. See Figure 2-7 on page 29.

Table 2-5 Configure federated server connection

Field	Value
Database name	EIPDB
Schema name	icmadmin
Authentication type	Server
Password	password


Configure Federated Server Connection
DB2.
IBM Enterprise Information Portal for Multiplatforms
Version 8.2
IBM.
< Back
Next >
Cancel

Type the information for the database to which you want to connect.

Database name: EIPDB

Schema name: ICMADMIN

Authentication type: Client [✓ Server]

Database connection ID: icmconct

Password: ********

Enable single sign-on

Catalog remote EIP database

Figure 2-7 Configure federated server connection

On completion of the EIP installation, the information in this window will be saved as a database entry in the $CMCOMMON\cmbds.ini file. By default, it is in the c:\Program Files\IBM\Cmgmt\ directory.

When you log on to eClient or the EIP administration client, the entries in the file cmbds.ini will be used to populate the EIP server list. Example 2-1 shows database EIPDB entry in this file.

Example 2-1 Sample entries in cmbds.ini file

```
FEDSERVER=EIPDB
FEDSERVERREPTYPE=DB2
FEDSCHEMA=ICMADMIN
FEDSSO=FALSE
FEDDBAUTH=SERVER
FEDREMOTE=FALSE
FEDHOSTNAME=
FEDPORT=
FEDREMOTEDB=
FEDNODENAME=
FEDOSTYPE=
```

Database connection user ID icmconct and its password in Figure 2-7 will be encrypted and saved in the $CMCOMMON\cmbfedenv.ini file. If you define a new user user1 in the EIP administration client and it is not defined as the database user in the EIPDB database, when the new user user1 logs on to the EIPDB server, user icmconct and its password in the cmnfedenv.ini file will actually be used to connect to the EIPDB database.

16. The next window is the Configure Content Manager V8 Server Connection window. Enter the values in Table 2-6 and click **Next**.

Table 2-6 Configure Content Manager server connection

Field	Value
Database name	EIPDB
Schema name	icmadmin
Authentication type	Server
Password	password

Restriction: Notice that `EIPDB` is entered as the database name even if you are configuring Content Manager V8 Server Connection. This is a temporary restriction at the time of writing and is documented in the EIP Readme file.

On completion of the EIP installation, the information in this window will be saved as a database entry in the $CMCOMMON\cmbicmsrvs.ini file. By default, it is in the c:\Program Files\IBM\Cmgmt\ directory.

When you log on to eClient or the Content Manager administration client, the entries in the cmbicmsrvs.ini file will be used to populate the Content Manager server list. Example 2-2 shows the database ICMNLSDB entry in this file.

Example 2-2 Sample content in cmbicmsrvs.ini file

```
ICMSERVER=EIPDB
ICMSERVERREPTYPE=DB2
ICMSCHEMA=ICMADMIN
ICMSSO=FALSE
ICMDBAUTH=SERVER
ICMREMOTE=FALSE
ICMHOSTNAME=
ICMPORT=
ICMREMOTEDB=
ICMNODENAME=
ICMOSTYPE=
```

17. Review your installation option and click **Next** to start copying files.

18. Select **Yes, I want to restart my computer now** when the installation is completed.

2.2.3 Post-installation

Since you selected the Information Mining feature during the EIP installation, you must enter `EIPDB` as the database name in step 16 on page 30 when you configured Content Manager Version 8 server connection.

If you have a separate Content Manager Version 8 database that you want EIP to connect to (in our scenario, it is icmnlsdb), you can run the EIP installation program again and select only the **ICMconnector**. During this second installation, you can specify a different database name in the Configure Content Manager Version 8 Server Connection window. This second installation will update the necessary configuration files.

Alternatively, you may add another database entry for DB2 Content Manager server manually in the $CMCOMMON\cmbicmsrvs.ini file.

For our scenario, the EIPDB database entry has been added in the C:\Program Files\ibm\Cmgmt\cmbicmsrvs.ini file after the installation. See Example 2-2 on page 30. We have to add a database entry for icmnlsdb in the file. After adding the second entry, the cmbicmsrvs.ini file should look similar to Example 2-3.

Example 2-3 Revised cmbicmsrvs.ini file

```
ICMSERVER=EIPDB
ICMSERVERREPTYPE=DB2
ICMSCHEMA=ICMADMIN
ICMSSO=FALSE
ICMDBAUTH=SERVER
ICMREMOTE=FALSE
ICMHOSTNAME=
ICMPORT=
ICMREMOTEDB=
ICMNODENAME=
ICMOSTYPE=
ICMSERVER=ICMNLSDB
ICMSERVERREPTYPE=DB2
ICMSCHEMA=ICMADMIN
ICMSSO=FALSE
ICMDBAUTH=SERVER
ICMREMOTE=FALSE
ICMHOSTNAME=
ICMPORT=
ICMREMOTEDB=
ICMNODENAME=
ICMOSTYPE=
```

If icmnlsdb is a remote database, you have to catalog it on the EIP server machine.

2.2.4 Configuring EIP

Information Integrator for Content Version 8 (EIP) has many configuration files. We discuss two important ones in this section. If the contents of these two files are inaccurate, you will not be able to connect to the content server.

cmbcs.ini file

This file indicates how EIP connects to different content servers. The options are either through local connectors or through remote connectors. The sample content of cmbcs.ini file is shown in Example 2-4.

Example 2-4 Sample content in cmbcs.ini file

```
FED=local
JDBC=local
DB2=local
DJ=local
DL=remote
ICM=local
TS=remote
QBIC=remote
IP=remote
DES=remote
DD=remote
OD=remote
IC=remote
V4=remote
```

For the rest of this redbook, we use local connector to connect to Content Manager server V8.2 and EIP federated database server V8.2.

cmbclient.ini file

If you choose the remote connector for any back-end server in the cmbcs.ini file, such as the OnDemand (OD) server in Example 2-4, the cmbclient.ini file becomes essential. In this case, EIP relies on the RMI server to connect to the OnDemand server, and RMI server information is defined in the cmbclient.ini file.

The cmbclient.ini file defines where the RMI server is running and what is the port number for the RMI server. Example 2-5 shows the sample content of cmbclient.ini file in our scenario. Note that in our scenario, the cmbclient.ini file is irrelevant because we use the local connectors.

Example 2-5 Sample content in cmbclient.ini file

```
# To point to an RMI server uncomment the lines below
# and update the host and port number of the RMI server.
RemoteHost=EIPserver
```

```
RemotePort=1919
```

To start the EIP RMI server, select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Start RMI Server**.

Note: Since you installed EIP Information Mining Service components, you can choose to connect to the Information Mining Service through a remote connector. In this case, you have to define Information Mining Service RMI server information in the cmbsvclient.ini file.

For our scenario, the local connector is used to connect to the Information Mining Service.

To start the Information Mining Service RMI server, select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Start RMI Server**.

2.2.5 Verifying installation

After you have installed EIP components on the machine, you should verify the connections to the content servers before installing eClient.

Verifying connections to Content Manager server V8.2

1. Open an EIP Development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**. All the necessary environment variables are set properly in this window in order to compile and run EIP Java sample codes.
2. Run the command **`cd C:\CMBROOT\samples\java\icm`**.
3. Run the command **`javac SConnectDisconnectICM.java`**.

   Note this command is case sensitive.
4. Run the **`java SConnectDisconnectICM icmnlsdb icmadmin password`** command.

   Note this command is case sensitive. In this command, `icmnlsdb` is the Content Manager Library Server name for our scenario, and `icmadmin` and `password` are the valid user ID and password in the Library Server. After executing the command, you should have output similar to Example 2-6.

Example 2-6 Sample output of SConnectDisconnectICM.java

```
===========================================
IBM Information Integrator for Content v8.2
Sample Program:  SConnectDisconnectICM
-------------------------------------------
```

```
 Database: icmnlsdb
 UserName: icmadmin
===========================================
Connecting to datastore (Database 'icmnlsdb', UserName 'icmadmin')...
Connected to datastore (Database 'icmnlsdb', UserName 'icmadmin').
Disconnecting from datastore & destroying reference...
Disconnected from datastore & destroying reference.

==========================================
Sample program completed.
==========================================
```

5. If the sample code failed to connect to the Content Manager server, you should check the Java error in the command window as well as the log file dklog.log. The dklog.log is the EIP log file; you can find it in the current working directory. In our scenario, the log file is in the C:\CMBROOT\SAMPLES\java\icm directory.

   You should also check the cmbcs.ini and cmbclient.ini files and ensure that the contents are accurate. If you use remote connectors for any content server, make sure that RMI server is running.

Verifying connections to EIP federated database server V8.2

1. Open an EIP Development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**. All the necessary environment variables are set properly in this window in order to compile and run EIP Java sample codes.
2. Run the command **`cd C:\CMBROOT\samples\java\fed`**.
3. Run the command **`javac TConnectFed.java`**.

   Note this command is case sensitive.
4. Run the command **`java SConnectDisconnectICM eipdb icmadmin password`**.

   Note this command is case sensitive. `eipdb` is the EIP database server name for our scenario, and `icmadmin` and `password` are the valid user ID and password. After executing the command, you should see output similar to Example 2-7.

Example 2-7 Sample output of TConnectFed.java

```
C:\CMBROOT\SAMPLES\java\fed>java TConnectFed eipdb icmadmin password
 *** connecting to datastore : eipdb
 *** datastore connected ***
user icmadmin dsName eipdb
datastore disconnected
user icmadmin dsName eipdb
```

5. If the sample code failed to connect to the EIP server, you should check the Java error in the command window as well as the log file dklog.log. The dklog.log is the EIP log file; you can find it in the current working directory. In our scenario, the log file is in C:\CMBROOT\SAMPLES\java\fed.

   You should also check the cmbcs.ini and cmbclient.ini files and ensure that the contents are accurate. If you use remote connectors for any content server, make sure that RMI server is running.

Verifying installation of the Information Mining Service

1. Open the Windows Services window and find the entry DB2EXT - DB2 service.
2. Open an EIP Development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**. All the necessary environment variables are set properly in this window in order to compile and run EIP Java sample codes.
3. If the DB2EXT - DB2 service is not started in Window Services window, run the command **`db2text start`** in the EIP Development window.
4. Run the command **`cd C:\CMBROOT\ikf\bin\tools`** in the EIP Development window.
5. Run the command **`run eipdb icmadmin password`** in the EIP Development window. `eipdb` is the EIP database for our scenario. You should have output similar to Example 2-8.

Example 2-8 Sample output of the Information Mining Service installation verification

```
connected to eipdb as icmadmin
catalog created
document filter finished
language identifier finished (en)
searching (waiting up to 6 minutes for index update)
result retrieved (PID1)
catalog deleted
cleanup finished
```

If the sample code failed, check the Java error on the command window as well as in the log file dklog.log. The dklog.log file is the EIP log file; you can find it in the current working directory. You might also check the Library Server log file. By default, the file is C:\ICMSERVER.LOG.

2.3 Installing and configuring the Information Structuring Tool on WebSphere Application Server

The Information Structuring Tool (IST) is a Web application that is required for the Information Mining Service. If you are going to work with the Information Mining integration described in later chapters of this redbook, IST is required.

Recommendation: To successfully execute Information Mining sample codes in this redbook, you should have at least 1 GB physical memory and 2 GB virtual memory.

Complete the following steps to install and configure the Information Structuring Tool in the WebSphere Application Server V5.0 environment:

1. Start the WebSphere Application Server:
   a. Open a command window.
   b. Run the command **`cd C:\WebSphere\AppServer\bin`**.
   c. Run the command **`startServer server1`**.
2. Identify `<Cell>, <Node> and <AppServer>` in WebSphere Application Server:
   a. Open Windows Explorer.
   b. Expand the directory structure C:\WebSphere\AppServer\config\cells\EIPserver\nodes\EIPserver\servers \server1.

      The cells directory contains available cells in WebSphere Application Server. In our scenario, EIPserver is the cell. The nodes directory contains a list of nodes in WebSphere Application Server. In our scenario, EIPserver is also the node. The servers directory contains a list of server on a node.
3. Set up a shared library in WebSphere Application Server with the necessary environment settings:
   a. Open the C:\CMBROOT\ikf\IST\bin\SetupIMEnv.cmd file in a file editor.
   b. Verify that all directories in this file match your installation directories, including, for example, WebSphere Application Server, EIP, and DB2. If necessary, make changes and save the file.
   a. Open a command window.
   b. Run the command **`cd C:\WebSphere\AppServer\bin`**.
   c. Run the command **`C:\CMBROOT\ikf\IST\bin\SetupIMEnv <Cell> <Node> <AppServer>`**

Where `<Cell>` is the name of the WebSphere Application Server administrative cell, `<Node>` is the name of WebSphere Application Server node where the Information Structuring Tool is installed, and `<AppServer>` is the application server (on `<Node>`) where the Information Structuring Tool is installed.

In our scenario, we use the command

`C:\CMBROOT\ikf\IST\bin\SetupIMEnv EIPserver EIPserver server1`

Note: The `<Cell>`, `<Node>` and `<AppServer>` are all case sensitive.

The output is shown in Example 2-9.

Example 2-9 Sample output of SetupINEnv command

```
C:\WebSphere\AppServer\bin>C:\CMBROOT\ikf\IST\bin\SetupIMEnv EIPserver
EIPserver server1
WASX7209I: Connected to process "server1" on node EIPserver using SOAP
connector;  The type of process is: UnManagedProcess
SetupIMEnv completed successfully.
```

4. Deploy the Information Structuring Tool via the WebSphere Application Server Administrative Console:

   a. Launch a WebSphere Application Server Administrative Console by selecting **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Administrative Console**.

   b. In the Navigation pane on the left, select **Applications -> Install New Application**.

   c. Set the values of the fields as in Table 2-7. Click **Next**.

Table 2-7 Deploying the Information Structuring Tool

Field	Value
Local path	C:\CMBROOT\ikf\IST\IST.war
Context root	/webApps/IST (it must end in /IST)

   d. On the next window, select **Default virtual host name for web modules** for the Virtual Host field.

   e. Click **Next** four times until you are in the Install New Application, Step 4: Summary window.

   f. Click **Finish**. The WebSphere Application Server starts to install the Information Structuring Tool and the output is shown in Example 2-10 on page 38.

Example 2-10   Sample output of installing Information Structuring Tool

```
Installing...

If there are EJB's in the application, the EJB Deploy process may take several
minutes. Please do not save the configuration until the process is complete.

Check the SystemOut.log on the Deployment Manager or Server where the
application is deployed for specific information about the EJB Deploy process
as it occurs.

ADMA5005I: Application IST_war configured in WebSphere repository

ADMA5001I: Application binaries saved in
C:\WebSphere\AppServer\wstemp\db2admin\workspace\cells\EIPserver\applications\I
ST_war.ear\IST_war.ear

ADMA5011I: Cleanup of temp dir for app IST_war done.

ADMA5013I: Application IST_war installed successfully.

Application IST_war installed successfully.
```

g. On the WebSphere Application Server menu bar, click **Save**. Click **Save** again in the Save window.

h. In the Navigation pane on the left, select **Applications -> Enterprise Applications** and select **IST_war**.

i. On the Configuration tab, set the values of the fields as in Table 2-8 and leave the rest of fields at the defaults.

Table 2-8   Configuring enterprise application IST_war

Field	Value
Enable Distribution	False
Reload Enabled	False

j. Click **Apply**.

k. Select **Libraries** under Additional Properties at the bottom of the window.

l. Click **Add**.

m. Select **InformationMiningEnvironment** from the Library Name drop-down list.

n. Click **OK**.

o. On the WebSphere Application Server menu bar, click **Save**. Click **Save** again in the Save window.

5. In the Navigation pane on the left, select **Environment -> Update Web Server Plugin**.
6. Click **OK** to update the Web server plug-in.
7. Stop the WebSphere Application Server:
   a. Open a command window.
   b. Run the command **`cd C:\WebSphere\AppServer\bin`**.
   c. Run the command **`stopServer server1`**.
8. Run the command **`C:\CMBROOT\ikf\IST\bin\ISTconfig.cmd`**:
   a. Open the C:\CMBROOT\ikf\IST\bin\ISTconfig.cmd file in a file editor.
   b. Verify that all directories in this file match your installation directories, including, for example, WebSphere Application Server, EIP and DB2. If necessary, make changes and save the file.
   c. Open a command window.
   d. Run the command **`cd C:\CMBROOT\ikf\IST\bin`** to switch to the IST source directory.
   e. Run the command **`ISTconfig <WAS_HOME> <Node>`**

      If `<WAS_HOME>` contains spaces, use quotes around it. If you are prompted whether files should be replaced, enter YES.

      In our scenario, we use the command:

      ```
      ISTconfig C:\WebSphere\AppServer EIPserver
      ```

      Example 2-11 shows the command output.

Note: `<Node>` is case sensitive.

Example 2-11 Sample output of ISTconfig command

```
C:\CMBROOT\ikf\IST\bin>ISTconfig C:\WebSphere\AppServer EIPserver
Configuring IST found under
 C:\WebSphere\AppServer\installedApps\EIPserver\IST_war.ear\IST.war...
Overwrite
C:\WebSphere\AppServer\installedApps\EIPserver\IST_war.ear\IST.war\applets.
jar (Yes/No/All)? y
C:\CMBROOT\ikf\IST\applets.jar
1 File(s) copied
Overwrite
C:\WebSphere\AppServer\installedApps\EIPserver\IST_war.ear\IST.war\clientUp
load.jar (Yes/No/All)? y
C:\CMBROOT\ikf\IST\clientUpload.jar
1 File(s) copied
```

```
Overwrite
C:\WebSphere\AppServer\installedApps\EIPserver\IST_war.ear\IST.war\clientTr
ee.jar (Yes/No/All)? y
C:\CMBROOT\ikf\IST\clientTree.jar
1 File(s) copied
IST configuration completed successfully.
```

9. By default, the IST uses icmnlsdb as the EIP database. Because we use EIPDB as the EIP database in our scenario, we need to change the appropriate parameter:
   a. Switch to the directory where the IST configuration has been deployed: <WAS_HOME>\config\cells\<Node>\applications\IST_war.ear\deployments\IST_war\IST.war\WEB-INF>.

      In our scenario, it is C:\WebSphere\AppServer\config\cells\EIPserver\applications\IST_war.ear\deployments\IST_war\IST.war\WEB-INF.
   b. Open the file web.xml. Be sure to back up the file before making changes.
   c. Replace `icmnlsdb` with `EIPDB`.
   d. Save the file.
10. Restart the WebSphere Application Server.
11. To launch the Information Structuring Tool:
   a. Open a browser.
   b. Enter the URL `http://localhost/<WebPath>/login.html`.

      In our scenario, since we use `/webApps/IST` as the `<WebPath>`, the URL is `http://localhost/webApps/IST/login.html`.
   c. Enter `icmadmin` and `password` to log in.
   d. The Information Structuring Tool home page is displayed.

Tip: If you receive the error message:

```
Your current security settings prohibit running ActiveX controls on this
page. As a result, the page may not display correctly.
```

while launching the Information Structuring Tool, most likely you do not have Java plug-in installed on your browser. Do the following:

1. Go to `http://java.sun.com/products/plugin` and download the Java plug-in, for example, Version 1.4.1_02.
2. Close all browsers.
3. Install the Java plug-in.
4. Open a browser and launch the Information Structuring Tool.

If you receive the error message `Loading Java Applet Failed` in the browser status bar while launching the Information Structuring Tool, make sure that you are not using a SOCKS proxy server. To turn off the SOCKS proxy server in Internet Explorer:

1. Select **Tools -> Internet options** from menu bar.
2. Open the **Connections** tab.
3. Click the **LAN Settings** button.
4. Unselect **Use a proxy server**.
5. Close all browsers.
6. Open a browser and launch the Information Structuring Tool.

2.4 Installing Content Manager Version 8 eClient

This section provides instructions for installing, configuring and verifying IBM DB2 Content Manager Version 8 eClient on a Windows platform.

If you have already installed Content Manager and Information Integrator for Content on your machine, your machine should have all the prerequisites required for installing eClient. For eClient hardware and software requirements, refer to Chapter 2, “Requirements” in *IBM Content Manager for Multiplatforms / IBM Information Integrator for Content: Installing, Configuring, and Managing eClient*, SC27-1350.

2.4.1 Preparing for installation

The eClient installation procedure will automatically deploy the eClient application server in WebSphere Application Server. In order to ensure successful deployment, make sure WebSphere Application Server is in the proper status at the time of installation depending on the version of the WebSphere Application Server you are using:

- If you are using WebSphere Application Server AES V4.05, stop any WebSphere Application Server server that is running.
- If you are using WebSphere Application Server AE V4.05, the WebSphere Application Server administration server (AE) service in a Windows Control Panel must be running.
- If you are using WebSphere Application Server V5.0, the WebSphere Application Server administration server (server1) must be started.

2.4.2 Installing eClient

Complete the following steps to install eClient:

1. Log on to the machine as a system administrator user.

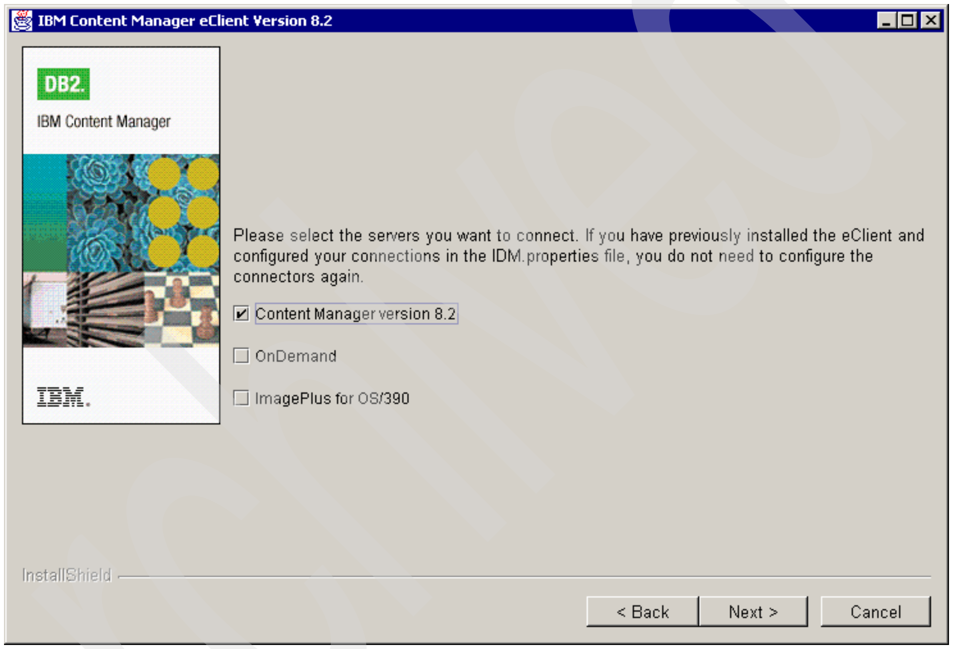
> **Note:** It is best practice to create a user, such as admin, and assign it to the system administrator group on the machine right after the operating system is installed. Then, you should log in as user admin to install all products, including DB2, WebSphere Application Server, Content Manager, EIP, and eClient.

2. Make sure that your WebSphere Application Server is in the proper status. For more details, see 2.4.1, "Preparing for installation" on page 42.

   If the WebSphere Application Server has an incorrect status, eClient installation process may not be able to successfully deploy the eClient application server.
3. Insert the eClient V8.2 CD into the CD-ROM drive.
4. On the eClient installation LaunchPad window, click **Install**.
5. Click **Next** on the Welcome window.
6. Select a language and click **Next**.
7. Enter the installation directory. This is C:\CMeClient for our scenario. Click **Next**.

8. If you have not installed EIP connectors, you will received error message and the installation process is stopped. See 2.2.2, “Installing EIP” on page 22 for connector installation instruction.
9. After reviewing installation options, click **Next** to start copying files.
10. Select **Content Manager Version 8.2** as shown in Figure 2-8 and click **Next**.

    Depending on what you select on this window, the installation will display different windows for you to configure various content servers. For our scenario, we will connect to the Content Manager V8.2 server.



Please select the servers you want to connect. If you have previously installed the eClient and configured your connections in the IDM.properties file, you do not need to configure the connectors again.

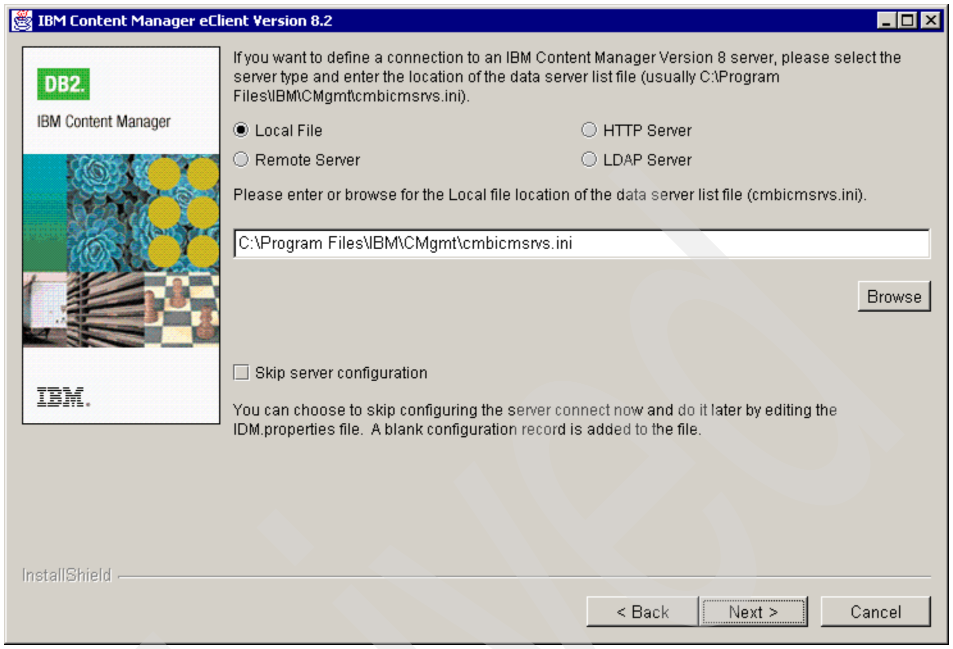
[✓] Content Manager version 8.2
[ ] OnDemand
[ ] ImagePlus for OS/390

Figure 2-8 Select server to connect

11. On the next window, shown in Figure 2-9 on page 44, enter the location of the cmbicmsrvs.ini file. The default location is C:\Program Files\IBM\CMgmt\cmbicmsrvs.ini on your local machine.

    Note the server configuration file can reside on the local machine, an HTTP server, remote server, or LDAP server. eClient uses the cmbicmsrvs.ini file to populate the Content Manager server list on the logon window.

    Click **Next**.


IBM Content Manager eClient Version 8.2

If you want to define a connection to an IBM Content Manager Version 8 server, please select the server type and enter the location of the data server list file (usually C:\Program Files\IBM\CMgmt\cmbicmsrvs.ini).

[● Local File] ○ HTTP Server
○ Remote Server ○ LDAP Server

Please enter or browse for the Local file location of the data server list file (cmbicmsrvs.ini).

C:\Program Files\IBM\CMgmt\cmbicmsrvs.ini

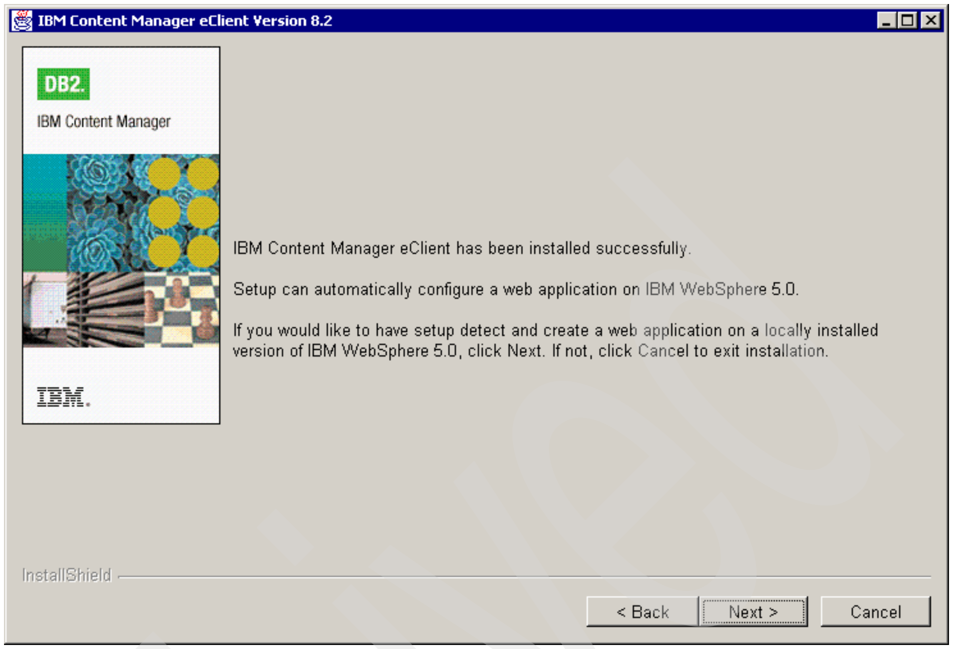
☐ Skip server configuration

You can choose to skip configuring the server connect now and do it later by editing the IDM.properties file. A blank configuration record is added to the file.

Figure 2-9 Specify the location for cmbicmsrvs.ini file

12. If you want the installation process to deploy the eClient application server, click **Next** in the window shown in Figure 2-10 on page 45. If you prefer to manually deploy it later, click **Cancel**.

    In order to successfully deploy eClient application server, your WebSphere Application Server must be in the proper status. Find details in 2.4.1, “Preparing for installation” on page 42.


IBM Content Manager eClient Version 8.2

IBM Content Manager eClient has been installed successfully.

Setup can automatically configure a web application on IBM WebSphere 5.0.

If you would like to have setup detect and create a web application on a locally installed version of IBM WebSphere 5.0, click Next. If not, click Cancel to exit installation.

Figure 2-10 Automatically deploy in WebSphere Application Server

13. On the next window, enter the WebSphere Application Server administrator user name and password if you have enabled the security feature in WebSphere Application Server. Otherwise, you do not have to enter data on this window.

    Click **Next** to deploy the eClient Web application server. Be patient, since it may take few minutes.

14. After the eClient application server is deployed successfully, the next window displays important information:

    a. To start WebSphere Advanced Single Server with the integrated eClient application installed, run **`startIDMAES.bat`** which is located in the eClient save directory.

       If you are using WebSphere Application Server AES V4.05 or WebSphere Application Server V5.0, **`startIDMAES.bat`** is the command to start the eClient application server. If you are using WebSphere Application Server AE V4.05, you need to start WebSphere Application Server in the Windows Services window.

       In our scenario, we use WebSphere Application Server V5.0. The startIDMAES.bat file is in C:\CMeClient\Save directory.

   b. To launch the eClient application, the URL is `http://<hostname>/eClient82/IDMInit`. This leads you to the eClient logon window.

15. Click **Next**.

16. Click **Finish**.

2.4.3 Configuring eClient

We discuss two of the eClient configuration files in the following sections: IDM.properties and IDMadminDefaults.properties. These two files control which features are available to end users and how documents are retrieved and displayed in eClient. In addition, we cover enabling the viewer applet.

IDM.properties file

The IDM.properties file is in the C:\CMeClient directory. This file controls what features are enabled or disabled for eClient. Changes made in the file take effect the next time the eClient property daemon checks the properties. If you have disabled the property daemon, you must restart the eClient application server to make the changes effective.

For example, to have the following four menu options on your eClient home page:

- Search
- Import
- Worklists
- Create Folder

you must set the parameters in the IDM.properties file as in Example 2-12. By default, the search capability is assigned to everybody.

Example 2-12 Enabling importing, worklist and creating folder capability

```
workFlowEnabled=true
importSupported=true
CreateFolderEnabled=true
```

For a list of sample parameters in the IDM.properties file, see Table 2-9 on page 47.

Table 2-9 Parameter description in IDM.properties file

Parameter	Description
TraceLevel	0 = tracing off. 1 = exceptions and errors. 2 = level 1 with general info, method entry/exit points. 3 = level 2 with API calls. 4 = level 3 with EIP non-visual bean tracing. 5 = performance tracing.
WorkingDir	Logging, tracing, and data conversion directory.
CacheDir	Storage area for document caching.
ImageURL	Path for the JSP images
MaxResults	Maximum search results displayed per screen. Default is 10.
TotalMaxResults	Maximum search results retrieved from the server per search criteria. Default is -1 for all hits.
cmbCC2MimeURL	Location of the cmbcc2mime.ini file consisting of the content classes associated with a MIME type.
CsIniURL	Location of the `cmbcs.ini` configuration file for the EIP connectors.
ClientIniURL	Location of the cmbclient.ini file that defines the RMI server.
cmbsvclient	Location of the cmbsvclient.ini file that defines the workflow RMI server.
ICMServersURL	Location of the CM V8 server initialization file.
ConnectionType	0 = local, 1 = remote, 2 = dynamic.
max_import_file_size	Maximum file size allowed during import. Default is 2 MB.
workFlowEnabled	Enable/disable the workflow functionality. Default is false.
checkInOutEnabled	Enable/disable check in/out capabilities. Default is false.
reIndexEnabled	Enable/disable reindexing of documents. Default is false.
emailEnabled	Enable/disable email support. Default is false.

Parameter	Description
viewerAppletEnabled	Enable/disable Applet Viewer. Default is false.
importSupported	Enable/disable import. Default is false.
CreateFolderEnabled	Enable/disable create folders. Default is false.
directRetrieveEnabled	Enable/disable direct retrieve from V8 Resource Manager. Default is true.
displayServerType	Display the server type with the server name on the logon page.
preferred_scale	Set the preferred scale for the document. A scale of 1.0 represents the actual size of 100%.
enhance_mode	Disable/enable image enhanced mode for the document viewer. This value does not apply to the viewer applet
enable_search_arguments	Repopulate search values on search jsps
SortChildAttributeValues	Disable/enable sorting of child attribute values within an item.
createFedFolderEnabled	Specifies if creation of federated folders is enabled.
ICMDisplayOrderEnabled	Specifies if attribute columns in search results page will be aligned by the order defined in ICM servers.

IDMadminDefaults.properties file

The IDMadminDefaults.properties file controls how eClient displays objects retrieved from content servers. For each supported MIME type, you can specify one of three options:

- *Applet*: Use the new eClient applet viewer.
- *Don't launch*: Force a file conversion on the eClient application server to a rendered type that can be handled by the browser.
- *Launch*: Launch the third-party application as viewer.

If you choose **launch** for a MIME type, you can select which application you use to view the retrieved objects. This is controlled by the file extensions in the IDMadminDefaults.properties file. In a Windows operating system, a MIME type is associated with a file extension and an application. Example 2-13 on page 49 shows a sample IDMadminDefaults.properties file.

Example 2-13 Sample IDMadminDefaults.properties file

```
application/pdf=launch
text/plain=don't launch
image/tiff=applet
image/gif=don't launch
image/jpeg=don't launch
text/html=launch
video/x-ibm-ivs=launch

## Format: [MIME_TYPE].extension={ preferred extension for the document type }

application/pdf.extension=pdf
application/vnd.ibm.modcap.extension=mda
audio/basic.extension=wav
audio/mpeg.extension=mp3
image/gif.extension=gif
image/jpeg.extension=jpg
image/tiff.extension=tif
text/html.extension=htm
text/plain.extension=txt
text/xml.extension= xml
video/x-ibm-ivs.extension=ivs
```

Enable viewer applet

Using a viewer applet enables users to perform actions such as annotation editing, rotation, zooming, and printing on retrieved documents. Enabling a viewer applet may improve performance if users frequently view large documents, view many documents per login session, or frequently manipulate the retrieved documents.

To enable the viewer applet, set the viewerAppletEnabled parameter to true, and use the IDMadminDefaults.properties file to set the display option to applet, as described in "IDMadminDefaults.properties file" on page 48.

A Java plug-in is also required to run the viewer applet. The plug-in might not be installed on the viewing machine. For Windows, specify the location from which the plug-in is automatically installed by Microsoft Internet Explorer or Netscape Navigator in the plugin_exe and plugin_page parameters, respectively. The default values for these parameters point to a JavaSoft Web site. You can change these default values for performance reasons, or to prevent your users from retrieving this plug-in from outside your firewall. For AIX and Solaris, you should install the Java plug-in Version 1.3.1 prior to running the eClient.

2.4.4 Verifying eClient installation

After you have installed eClient, configured the eClient application server and the Web server plug-in, you can access eClient via `http://<hostname>/eClient82/IDMInit`.

1. Start the eClient application server by running **`C:\CMeClient\Save\startIDMServer.bat`**.

   Remember that we are using WebSphere Application Server V5.0. If you are using WebSphere Application Server AES V4.05, use the same command to start the eClient application server. If you are using WebSphere Application Server AE V4.05, you must have the WebSphere Application Server administration service running and start the eClient application server in the WebSphere Application Server Administration Console.

2. Open a browser.
3. Enter `http://<hostname>/eClient82/IDMInit`. If you do not have Web server or the Web server plug-in is not configured properly, you can alternatively use `http://<hostname>:9080/eClient82/IDMInit` (assuming eClient is using port 9080). In this case, the HTTP request from the browser bypasses Web server and goes directly to the eClient application server.
4. You should see the eClient logon window as shown in Figure 2-11.

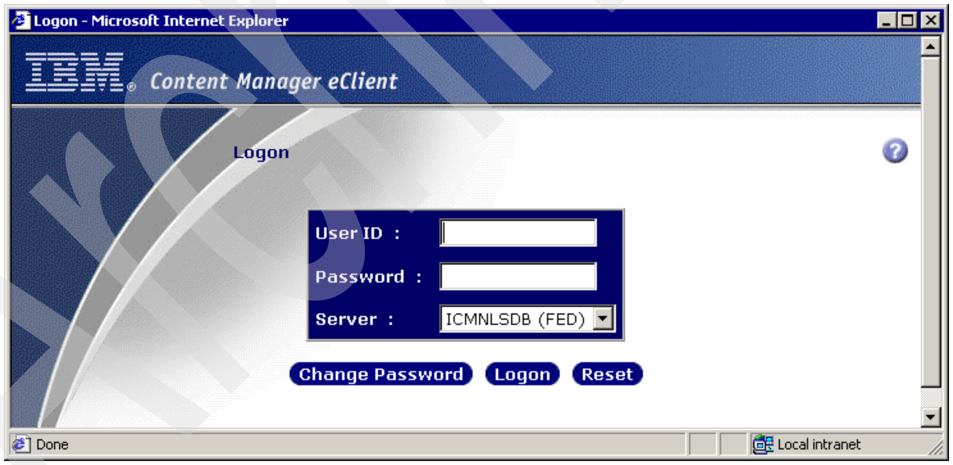


Figure 2-11 eClient logon window

This concludes the eClient installation verification.

3

Installing eClient in a WebSphere Network Deployment environment

This chapter provides detailed procedures for installing and configuring eClient in a WebSphere Application Server Network Deployment environment on the Windows platform. Before starting your installation, read Chapter 1, “Introducing Content Manager” on page 3 and optionally read Chapter 5, “eClient architecture” on page 101. They will help you understand Content Manager product portfolio and the eClient architecture for a successful installation.

This chapter covers the following topics:

- Installing HTTP Web server
- Installing WebSphere Network Deployment
- Configuring WebSphere Network Deployment
- Installing EIP
- Installing eClient
- Creating eClient cluster server
- Configuring HTTP Web server
- Monitoring workload balance

Note: This chapter focuses on a more complex eClient installation. For a simple, first-time installation, read Chapter 2, "Installing eClient" on page 17.

3.1 Introduction

If a small group of users needs to access the eClient server, a single Web server (HTTP server) and a single WebSphere Application Server are used. The simple installation and configuration discussed in Chapter 2, "Installing eClient" on page 17 should serve well.

If a large group of users needs to access the eClient server, an eClient server with a single HTTP server and a single WebSphere Application Server do not have enough process power to serve everyone. In this case, you may need a cluster of HTTP server, or a cluster of WebSphere Application Server, or both, depending on where the bottleneck is. Optionally, you may also consider having multiple Resource Managers for the Content Manager system.

3.1.1 Introducing the scenario

In this chapter, we show you how to install and configure a system with one HTTP server and a cluster of eClient servers with two members. If you follow all the procedures in this chapter, upon completion of the chapter, you should have a system similar to the one shown in Figure 3-1 on page 53.

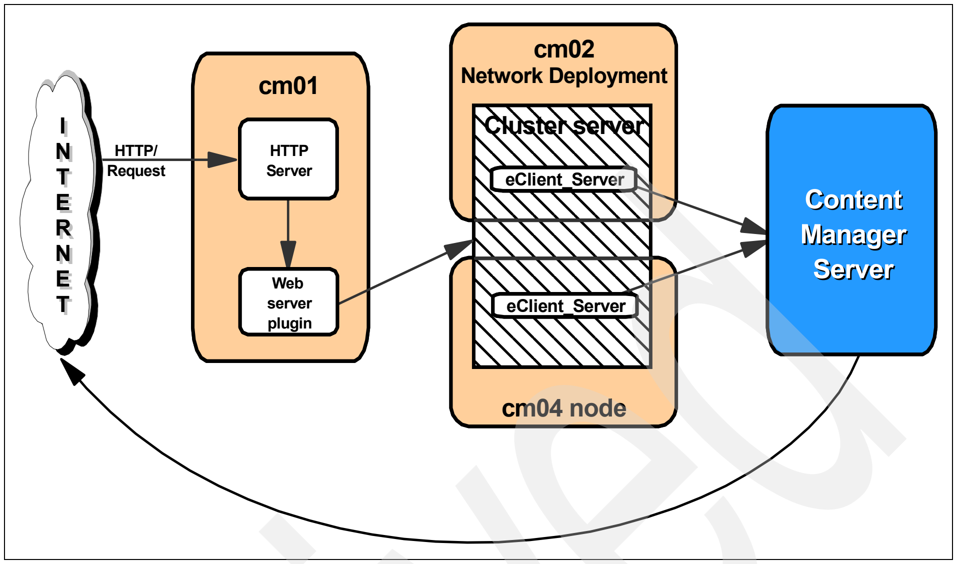


Figure 3-1 Scenario diagram

The configuration consists of the following:

- An HTTP Web server on machine cm01 with the proper Web server plug-in file for the eClient cluster server in the WebSphere Network Deployment.
- A WebSphere Application Server Network Deployment environment on machine cm02 with two node in the cell. One node is on machine cm02 and the other one is on machine cm04.
- An eClient cluster server created in the WebSphere Network Deployment environment. It has two cluster members. One is on node cm02 and the other is on node cm04.
- A Content Manager server that has been preinstalled and preconfigured on machine cm03. Since the Content Manager configuration is outside the scope of this redbook, we do not cover it here.

3.1.2 What is WebSphere Network Deployment?

In this chapter, we use WebSphere Application Server Network Deployment to implement the cluster application server for eClient.

WebSphere Application Server Network Deployment is designed to add non-programming enhancements to the features provided in the base WebSphere Application Server configuration. These enhancements allow you to run applications on multiple servers and on multiple physical nodes.

Network Deployment provides scalability, performance, availability and centralized management for IBM WebSphere Application Server Enterprise. A Network Deployment environment consists of multiple WebSphere Application Server V5.0 nodes grouped into a single administrative domain.

IBM WebSphere Application Server Network Deployment provides three types of workload management (WLM):

- Web server WLM: Use the Load Balancer feature from Edge Components as an IP sprayer to distribute HTTP requests across Web servers.
- Web server plug-in WLM: Distribute servlet requests across Web containers.
- Enterprise Java Services (EJS) WLM: Distributes EJB requests across EJB containers.

Along with the workload management feature, the Webpshere Application Server Network Deployment also provides failover support. The application may continue to process client requests when one of the servers is stopped or restarted.

3.2 Installing HTTP Web server

To demonstrate the concept clearly, we install the HTTP Web server on a different machine other than where the WebSphere Application Server and Network Deployment are installed. The HTTP Web server and its plug-in are installed on machine cm01 for our scenario.

Complete the following steps to install HTTP Web server on the Windows platform:

1. Log on as an administrator user to the machine where the HTTP server is installed.
2. Insert the IBM WebSphere Application Server V5 CD into the CD-ROM drive. This CD-ROM also contains the IBM HTTP Server and its plug-in.
3. Open a Windows Explorer, and switch to the \NT directory on the CD.
4. Double-click **LaunchPad.bat** to start the install.
5. Select a language for the LaunchPad and click **OK**.
6. Review the Readme file and installation guide.
7. Click **Install the product**.
8. Select a language to be used for the installation and click **OK**.
9. On the Welcome window, click **Next** to continue.

10. Accept the license agreement and click **Next**.
11. The installation will verify that your system has the required prerequisites.
12. Select **Custom** for the installation type and click **Next** to continue.
13. On the next window, a list of features to install is displayed. Since we will install and run only HTTP Web server on this machine (cm01), select the following entries from the list:
    – IBM HTTP Server Version 1.3.26
    – Web Server Plugin for IBM HTTP Server
14. Select **Next** to continue.
15. Enter the directories to be used for the IBM HTTP Server and Web server plug-in installation. In our scenario, it is C:\IBMHttpServer. The WebSphere installation directory is used to hold the Web server plug-in and the installation log files.
16. Click **Next** to continue.
17. If you choose to install the HTTP server as a Windows service, you are prompted to provide a user ID and password for starting and stopping the HTTP Server service.
18. Click **Next**.
19. Review the information in the Summary window.
20. Click **Next** to begin copying the files.
21. Register the product.
22. Click **Finish** to complete the installation.

Verification

To verify if the installation has completed successfully, follow these steps:

1. Start IBM HTTP Administration 1.3.26 and IBM HTTP Server 1.3.26 services in the Windows Control Panel.
2. Open a browser and enter `http://localhost`.
3. If the IBM HTTP Server home page is displayed, your installation is successful.

3.3 Installing WebSphere Network Deployment

A Network Deployment configuration includes support for multiple nodes, each with a node agent process and several application servers, all coordinated within an administrative cell by the Deployment Manager process. Clusters of load-balanced application servers can be configured within a Network Deployment cell.

The configuration and application binaries of all components in the cell are managed by the Deployment Manager and synchronized out to local copies on each of the nodes.

A cell is a group of nodes in a single administrative domain. Each node is identified by a logical name for configuration purposes. The configuration and application binaries of all nodes in the cell are centralized in a cell master configuration repository. This centralized repository is managed by the Deployment Manager process and synchronized out to local copies held on each of the nodes.

3.3.1 Installing WebSphere Application Server

A Network Deployment environment consists of multiple WebSphere Application Server V5 nodes grouped into a single administrative domain. On each node in the cell, you need to install WebSphere Application Server.

Follow the normal installation procedure (the same instruction that is used if the Network Deployment configuration is not involved) to install WebSphere Application Server on each node. This should be done for each node in the cell before you start installing the WebSphere Application Server Network Deployment.

Since the system will be quite complicated after everything is installed and configured, you should consider doing verification after each installation. This helps you to isolate and catch any problems in the early stages.

Follow these steps to verify the WebSphere installation:

1. Open a command window.
2. Run the command **`cd C:\WebSphere\ApServer\bin`**.
3. Run the command **`serverStatus -all`** to check the server's running status.
4. Run the command **`startServer server1`** to start it if server1 is not running.
5. Open a browser and enter `http://localhost/snoop`.
6. If you fail to display the servlet page, you should refer to the WebSphere troubleshooting procedure to fix the problem.

For our scenario, we have two nodes in the cell. One of them resides on machine cm02. The other one is on cm04.

3.3.2 Installing WebSphere Network Deployment

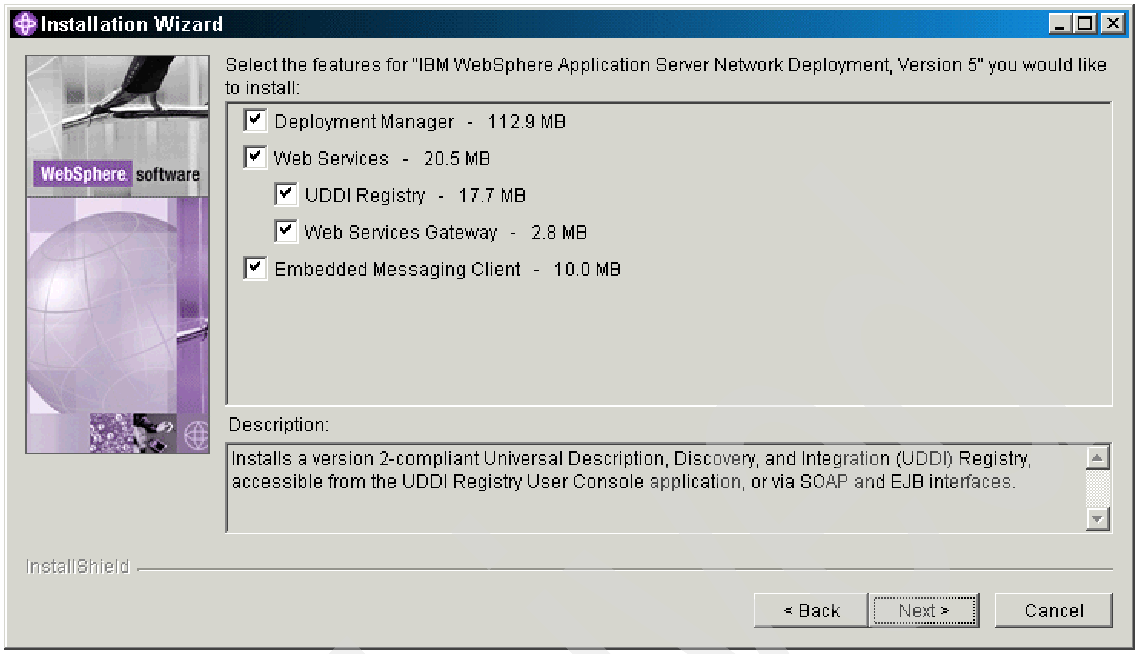
You may install WebSphere Application Server Network Deployment on separate machines or on the same machine where one of your nodes in the cell resides.

For our demonstration, we chose the latter option. We will install the WebSphere Application Server Network Deployment on the machine cm02 that has one of the two nodes in the cell.

Complete the following steps to install WebSphere Network Deployment on a Windows platform:

1. Log on as an administrator user to the machine where you will install the WebSphere Network Deployment.
2. Insert the Network Deployment CD. Note that the Network Deployment CD is not included in the Content Manager package. You need to purchase it separately.
3. Start the installation by clicking **LaunchPad.bat** in the \NT directory.
4. Select a language for the LaunchPad and click **OK**.
5. Review the Readme file and the installation guide.
6. Click **Install the product**.
7. Select a language to be used for the installation, and select **OK**.
8. Click **Next** on the Welcome window.
9. The License Agreement window appears. After reading the License Agreement, accept the agreement and click **Next**.
10. If you have installed a previous version of WebSphere Application Server, you have the opportunity to select alternate ports. For our scenario, the default port number is used. Click **Next** to continue.
11. Once the installation wizard confirms that all prerequisites have been met, or you elect to ignore the missing prerequisites, the wizard continues.

    If you are missing any prerequisite, a list of what is missing appears. You should stop at this point and bring the system up to the required level. If you choose to continue, the installation wizard continues.
12. The features that are available for installation are displayed on the next window. See Figure 3-2 on page 58. Take the default and click **Next**.


Installation Wizard

Select the features for "IBM WebSphere Application Server Network Deployment, Version 5" you would like to install:

- [✓] Deployment Manager - 112.9 MB
- [✓] Web Services - 20.5 MB
  - [✓] UDDI Registry - 17.7 MB
  - [✓] Web Services Gateway - 2.8 MB
- [✓] Embedded Messaging Client - 10.0 MB

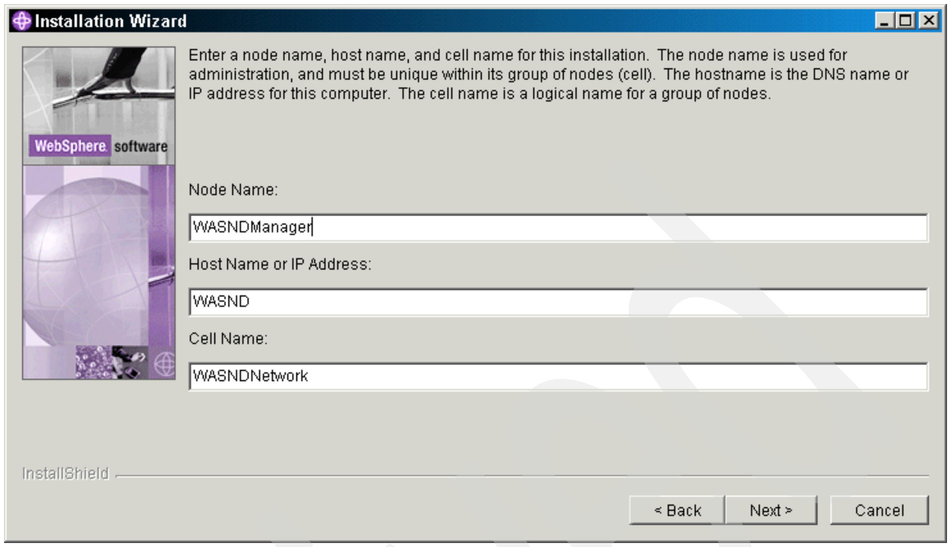
Description:

Installs a version 2-compliant Universal Description, Discovery, and Integration (UDDI) Registry, accessible from the UDDI Registry User Console application, or via SOAP and EJB interfaces.

Figure 3-2 Select WebSphere Network Deployment installation components

13. Enter the installation directory, such as C:\WebSphere\DeploymentManager, verify that you have enough free space on your hard disk, and click **Next**.
14. On the next window, enter a WebSphere Application Server node name, network host name, and cell name for your installation. See Figure 3-3 on page 59.

    The node name must be unique in the cell. If you select a name that is not unique, the install continues, but you will have problems later when you attempt to add the node to the cell.


Installation Wizard

Enter a node name, host name, and cell name for this installation. The node name is used for administration, and must be unique within its group of nodes (cell). The hostname is the DNS name or IP address for this computer. The cell name is a logical name for a group of nodes.

Node Name: WASNDManager

Host Name or IP Address: WASND

Cell Name: WASNDNetwork

Figure 3-3 Enter a WebSphere Application Server node name

15. Click **Next**.
16. You may choose to run WebSphere Application Server Network Deployment as a service on your Windows system. If you choose to do so, enter a user ID and password to start the Deployment Manager service.
17. if the user who is doing the installation does not have adequate access privileges, you may receive the warning “`INST0056E: The user name and password specified cannot be validated due to insufficient privileges of the current user. The privileges have been given, but will not take effect until the next login`”. Click **OK** to continue.
18. Review your installation option and click **Next** to start installation process.
19. Click **Next** to register the product.
20. Click **Finish** to complete the installation.

3.3.3 Verification

After the installation completes, First Steps is launched automatically. From this window, you can view the InfoCenter, start and stop the Deployment Manager, run the installation verification tests, and access the Administrative Console.

Click **Verify Installation** in the First Steps window. First Steps launches the verification program. If everything is successfully installed, you will see the messages shown in Example 3-1 on page 60.

Example 3-1 Output of verification program

```
OS: Windows 2000
locale: en_US
hostname: WASND
cmd.exe /c "C:\WebSphere\DeploymentManager\bin\ivt"
IVTL0095I: defaulting to host CM02 and port 9090
IVTL0010I: Connecting to the WebSphere Application Server CM02 on port: 9090

IVTL0020I: Could not connect to Application Server, waiting for server to start
IVTL0025I: Attempting to start the Application Server
osName = Windows 2000
IVTL0030I: Running cmd.exe /c "C:\WebSphere\DeploymentManager\bin\startManager"
>ADMU0116I: Tool information is being logged in file
>           C:\WebSphere\DeploymentManager\logs\dmgr\startServer.log
>ADMU3100I: Reading configuration for server: dmgr
>ADMU3200I: Server launched. Waiting for initialization status.
>ADMU3000I: Server dmgr open for e-business; process id is 1828
IVTL0070I: IVT Verification Succeeded

IVTL0080I: Installation Verification is complete
```

If you encounter any problem starting the server during the verification, check the installation log files. You may find the log files in the <WAS_ND_HOME>\logs directory, for example C:\WebSphere\\DeploymentManage\logs.

3.4 Configuring WebSphere Network Deployment

You have just gone through the process of installing Network Deployment. This gives you a cell framework; however, there are no nodes in the cell yet. In order for a node to be managed by the Deployment Manager, it must be added to the cell.

Complete the following steps to configure the WebSphere Application Server Network Deployment environment:

1. On the Deployment Manager node (which is machine cm02 in our case), start the Deployment Manager:
   a. Open a command window.
   b. Run the command **`cd <WAS_ND_HOME>\bin`**.

      For our scenario, run **`cd C:\WebSphere\DeploymentManager\bin`**.
   c. Run the command **`startManager`**.
2. On each of the WebSphere Application Server nodes (on machines cm02 and cm04), stop all the servers.

a. Open a command window.

b. Run the command **`cd <WAS_HOME>\bin`**.

For our scenario, run **`cd C:\WebSphere\AppServer\bin`**.

c. Run the command **`serverStatus -all`** to check the running status for all application servers. See a sample output in Example 3-2.

Example 3-2 Output of command serverStatus -all

```
C:\WebSphere\AppServer\bin>serverstatus -all
ADMU0116I: Tool information is being logged in file C:\Program
           Files\WebSphere\AppServer\logs\serverStatus.log
ADMU0500I: Retrieving server status for all servers
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU0509I: The Application Server "server1" cannot be reached. It appears to be
           stopped.
```

d. If you have a new WebSphere Application Server installation on the node, you should only have one server, server1. If server1 is running, stop it by executing the command **`stopServer server1`**.

3. On each of the WebSphere Application Server nodes (on machines cm02 and cm04), add the node to the cell with the following commands in the same command window as in step 2:

```
addNode <DM_hostname> 8879 -includeapps
```

Note that the **`addNode`** command is run on the machine where the application server resides, not the machine where the Network Deployment resides. The `<DM_hostname>` is the host name of the Network Deployment machine and the port (`8879`) is the SOAP connector port for the Deployment Manager.

In our case, we run the command:

```
addNode cm02 8879 -includeapps
```

on both cm02 and cm04, because we have application servers on both machines, although we have also chosen to install the Network Deployment on machine cm02.

When the **`addNode`** command is completed, you should see messages similar to Example 3-3.

Example 3-3 Sample output of command addNode

```
C:\WebSphere\AppServer\bin>addnode cm02 8879 -includeapps
ADMU0116I: Tool information is being logged in file
           C:\WebSphere\AppServer\logs\addNode.log
ADMU0001I: Begin federation of node cm04 with Deployment Manager at cm02:8879.
ADMU0009I: Successfully connected to Deployment Manager Server: cm02:8879
```

```
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU2010I: Stopping all server processes for node cm04
ADMU0512I: Server server1 cannot be reached. It appears to be stopped.
ADMU0024I: Deleting the old backup directory.
ADMU0015I: Backing up the original cell repository.
ADMU0012I: Creating Node Agent configuration for node: cm04
ADMU0014I: Adding node cm04 configuration to cell: cm02Network
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0018I: Launching Node Agent process for node: cm04
ADMU0020I: Reading configuration for Node Agent process: nodeagent
ADMU0022I: Node Agent launched. Waiting for initialization status.
ADMU0030I: Node Agent initialization completed successfully. Process id is:
           1532
ADMU0523I: Creating Queue Manager for node cm04 on server jmsserver
ADMU0525I: Details of Queue Manager creation may be seen in the file:
           createMQ.cm04_jmsserver.log
ADMU9990I:
ADMU0300I: Congratulations! Your node cm04 has been successfully incorporated
           into the cm02Network cell.
ADMU9990I:
ADMU0306I: Be aware:
ADMU0302I: Any cell-level documents from the standalone cm04 configuration have
           not been migrated to the new cell.
ADMU0307I: You might want to:
ADMU0303I: Update the configuration on the cm02Network Deployment Manager with
           values from the old cell-level documents.
ADMU9990I:
ADMU0003I: Node cm04 has been successfully federated.
```

4. On each node in the cell (on machine cm02 and cm04), start the node agent. The node agent is started automatically during the addNode process in step 3, but subsequently must be started manually by doing the following on the WebSphere Application Server machine:

   a. Open a command window.

   b. Run the command **cd <WAS_HOME>\bin**.

      For our scenario, run **cd C:\WebSphere\AppServer\bin**.

   c. Run the command **startnode**.

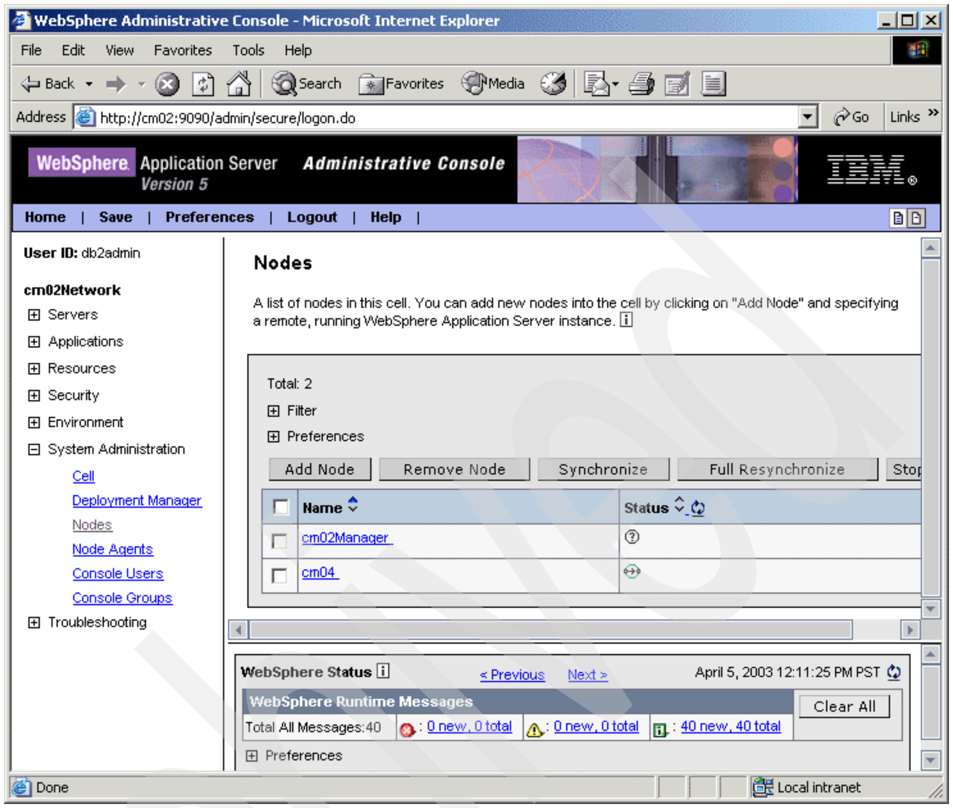
   A node agent is required to be running on the node in order for the Deployment Manager to communicate with the node. When the command is completed, you should see messages similar to Example 3-4.

Example 3-4 Sample output of command startnode

```
C:\WebSphere\AppServer\bin>startNode
ADMU0116I: Tool information is being logged in file
```

```
         C:\WebSphere\AppServer\logs\nodeagent\startServer.log
ADMU3100I: Reading configuration for server: nodeagent
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server nodeagent open for e-business; process id is 764
```

5. On the Deployment Manager node (machine cm02), stop and restart the Deployment Manager (the IBM WebSphere Application Server V5 - dmgr service in the Windows Services panel).
6. On the Deployment Manager node (machine cm02), launch a Network Deployment Administrative Console by selecting **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Network Deployment -> Administrative Console**.
7. In the Network Deployment Administrative Console, select **System Administration -> Nodes** in the Navigation pane on the left.
8. Check the status of the new node (cm02 and cm04) and make sure they are synchronized. If not, select the new node that is not synchronized and click the **Synchronize** button. See Figure 3-4 on page 64.


WebSphere Administrative Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://cm02:9090/admin/secure/logon.do Go Links
WebSphere Application Server Version 5 Administrative Console IBM
Home | Save | Preferences | Logout | Help |
User ID: db2admin
cm02Network
Servers
Applications
Resources
Security
Environment
System Administration
Cell
Deployment Manager
Nodes
Node Agents
Console Users
Console Groups
Troubleshooting

Nodes

A list of nodes in this cell. You can add new nodes into the cell by clicking on "Add Node" and specifying a remote, running WebSphere Application Server instance.

Total: 2

Filter

Preferences

Add Node | Remove Node | Synchronize | Full Resynchronize | Stop

	Name	Status
☐	cm02Manager	?
☐	cm04	⇔

WebSphere Status < Previous Next > April 5, 2003 12:11:25 PM PST

WebSphere Runtime Messages Clear All

Total All Messages:40 : 0 new, 0 total : 0 new, 0 total : 40 new, 40 total

Preferences

Figure 3-4 Node status in WebSphere Administrative Console

9. In the Network Deployment Administrative Console, select **System Administration -> Node Agents** in the Navigation pane on the left. In the Node agents pane on the right, it displays the node agent status for each node.

3.5 Installing EIP

To install and configure the eClient server in a WebSphere Application Server Network Deployment environment, you need to install minimal Information Integrator for Content (EIP) components on each node in the cell. The EIP components are part of the eClient prerequisites.

Follow the same steps in 2.2, “Installing Information Integrator for Content Version 8 (EIP)” on page 19 to install EIP. However, for step 8, 12, 13, 14 and 16, you must substitute the input values with the ones provided below.

In our scenario, we installed the EIP on machines cm02 and cm04. The installation steps are identical for both of them.

For step 8 on page 23, on the Component Selection window, select the components and subcomponents in Table 3-1.

Table 3-1 Select components to install

Component	Subcomponent
Local connectors	Content Manager V8 connector Federated connector Relational database connector
Connector toolkits and samples	Content Manager V8 connector Federated connector Relational database connector

For our scenario, we chose not to install EIP administration database and administration client on either cm02 or cm04. We have the EIP server running on a separate machine (cm03).

Step 12 on page 26, step 13 on page 27 and step 14 on page 28 do not apply in this installation because we chose not to install EIP administration database and administration client on either cm02 or cm04.

For step 16 on page 30, on the Configure Content Manager V8 Server Connection window, enter the values in Table 3-2.

Table 3-2 Configure Content Manager server connection

Field	Value
Database name	icmnlsdb
Schema name	icmadmin
Authentication type	Server
Password	password

icmnlsdb is the Library Server name on machine cm03. You need to catalog the databases icmnlsdb and EIPDB on both the cm02 and cm04 machines. (EIPDB is the EIP administration database on machine cm03.)

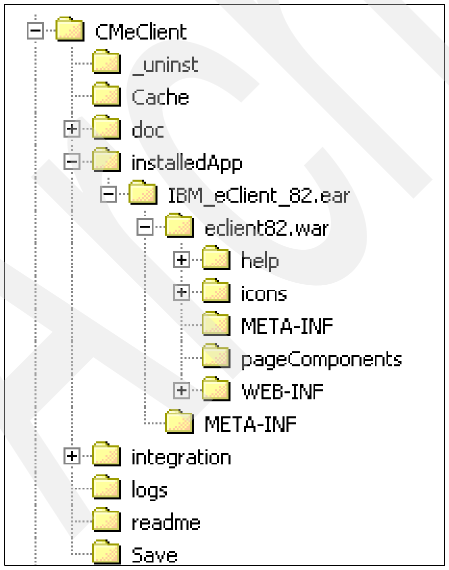
3.6 Installing eClient

We set up the WebSphere Application Server Network Deployment environment and installed EIP components (on cm02 and cm04). Now, we show you how to install the eClient server on each node in the cell. You must make the eClient components and its environment available on every node before you create the eClient cluster server.

Follow the same steps in 2.4, “Installing Content Manager Version 8 eClient” on page 41 to install the eClient.

In our scenario, we will install the eClient on the machine cm02 first. The eClient installation process automatically deploys the eClient application server in WebSphere. It can be verified in one of the following ways:

- In the Windows file system
  a. Open Windows Explorer.
  a. Go to the directory where the eClient is installed. In our example, this is C:\CMeClient.
  a. Expand the installedApp subdirectory. A directory tree structure appears as shown in Figure 3-5.



```
CMeClient
  _uninst
  Cache
  doc
  installedApp
    IBM_eClient_82.ear
      eclient82.war
        help
        icons
        META-INF
        pageComponents
        WEB-INF
      META-INF
  integration
  logs
  readme
  Save
```

Figure 3-5 eClient is deployed automatically on machine cm02

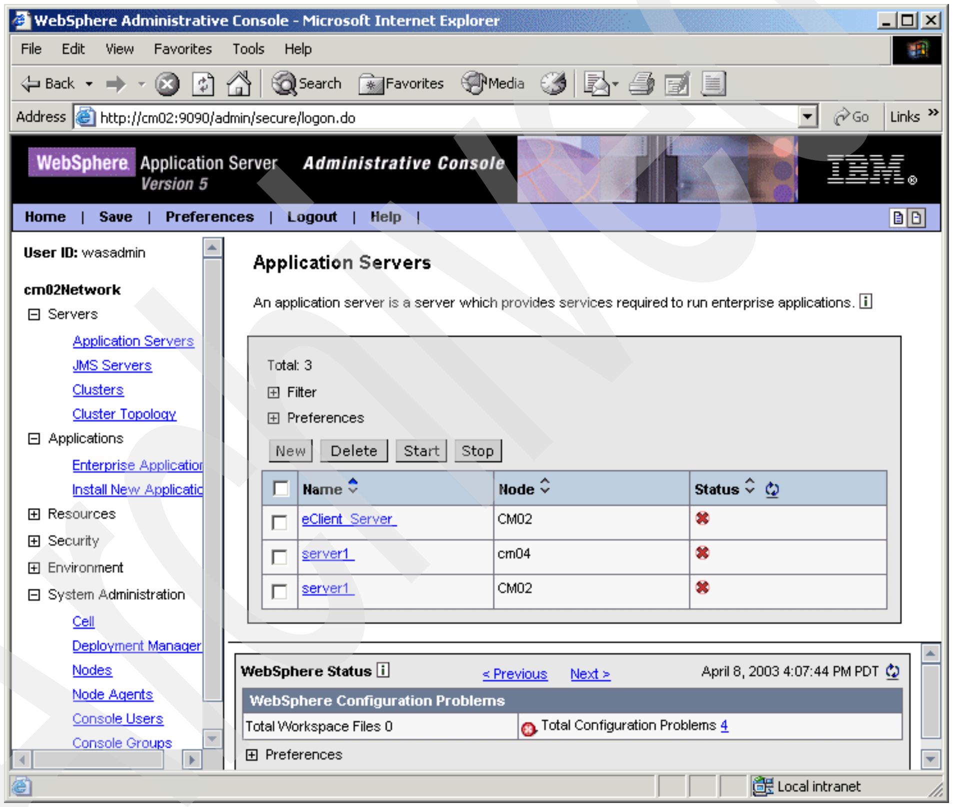
- The WebSphere Administrative Console

a. On the Deployment Manager node (machine cm02), launch the Network Deployment Administrative Console. Click **Start -> Programs -> IBM WebSphere --> Application Server v5.0 -> Network Deployment -> Administrative Console**.

b. In the Network Deployment Administrative Console, select **Servers -> Application Servers** in the Navigation pane on the left.

c. A list of servers is displayed in the Nodes window on the right. eClient_Server is one of the servers on node cm02. See Figure 3-6.

d. Select server **eClient_Server** and click **Start** to start the eClient server.



Application Servers

An application server is a server which provides services required to run enterprise applications.

Total: 3

Filter

Preferences

New | Delete | Start | Stop

	Name	Node	Status
☐	eClient_Server	CM02	✖
☐	server1	cm04	✖
☐	server1	CM02	✖

WebSphere Status   < Previous   Next >   April 8, 2003 4:07:44 PM PDT

WebSphere Configuration Problems

Total Workspace Files 0	Total Configuration Problems 4

Preferences

Figure 3-6 eClient_Server on node cm02

After the verification, we install eClient on machine cm04. The installation must be completed successfully. However, the eClient installation process does not automatically deploy the eClient application server in WebSphere this time, because the server eClient_Server exists in the WebSphere Network Deployment environment. This is the result of the previous eClient installation on cm02. The application server name must be unique in the WebSphere environment.

Tip: It is not a failure that the eClient server is not automatically deployed in Webpshere on node cm04.

Verification

Before creating an eClient cluster server, you should test if the standard eClient installation completed successfully on machine cm02. This may be done with the following steps:

1. On the Deployment Manager node (machine cm02), launch the Network Deployment Administrative Console. Click **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Network Deployment Administrative Console**.
2. In the Network Deployment Administrative Console, select **Servers -> Application Servers** in the Navigation pane on the left.
3. A list of servers appears in the Nodes window on the right. eClient_Server is one of the servers on node cm02. See Figure 3-6 on page 67.
4. Select server **eClient_Server** and click **Start** to start the eClient server.
5. Follow the steps in 3.8, "Configuring HTTP Web server" on page 73 to regenerate the Web server plug-in.
6. Open a browser on any machine and enter `http://cm01/eClient82/IDMInit`. This should lead you to the eClient logon window.
7. If eClient does not work on cm02, you may consider removing the eClient application server. Then, manually deploy it again in the following steps:
   a. Open a command window.
   b. Run the command cd **`c:\CmeClient\Save`**.
   c. Run the command **`idmwas.bat [userid] [password]`** to deploy it again.

To verify the eClient installation on machine cm04, check to make sure the C:\CMeClient directory and its subdirectory have been created.

3.7 Creating eClient cluster server

Up to this point, you have all the necessary components to create the eClient cluster server. The WebSphere Application Server and Network Deployment environment have been configured. EIP and eClient have been installed on each node in the cell. The eClient has been deployed on node cm02, but not on node cm04. On node cm02, the directory C:\CmeClient\installedApp\ IBM_eClient_82.ear has been created and populated. On node cm04, the directory C:\CmeClient\installedApp\ IBM_eClient_82.ear does not exist.

3.7.1 Creating eClient cluster server

Complete the following steps to create an eClient cluster server:

1. On the Deployment Manager node (machine cm02), launch the Network Deployment Administrative Console. Click **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Network Deployment Administrative Console**.
2. In the Network Deployment Administrative Console, select **Servers -> Clusters** in the Navigation pane on the left. The Server Cluster window is displayed on the right.
3. Click the **New** button to add a new cluster server.
4. Set the values of the fields in Table 3-3. See Figure 3-7 on page 70.

Table 3-3 Enter basic cluster information

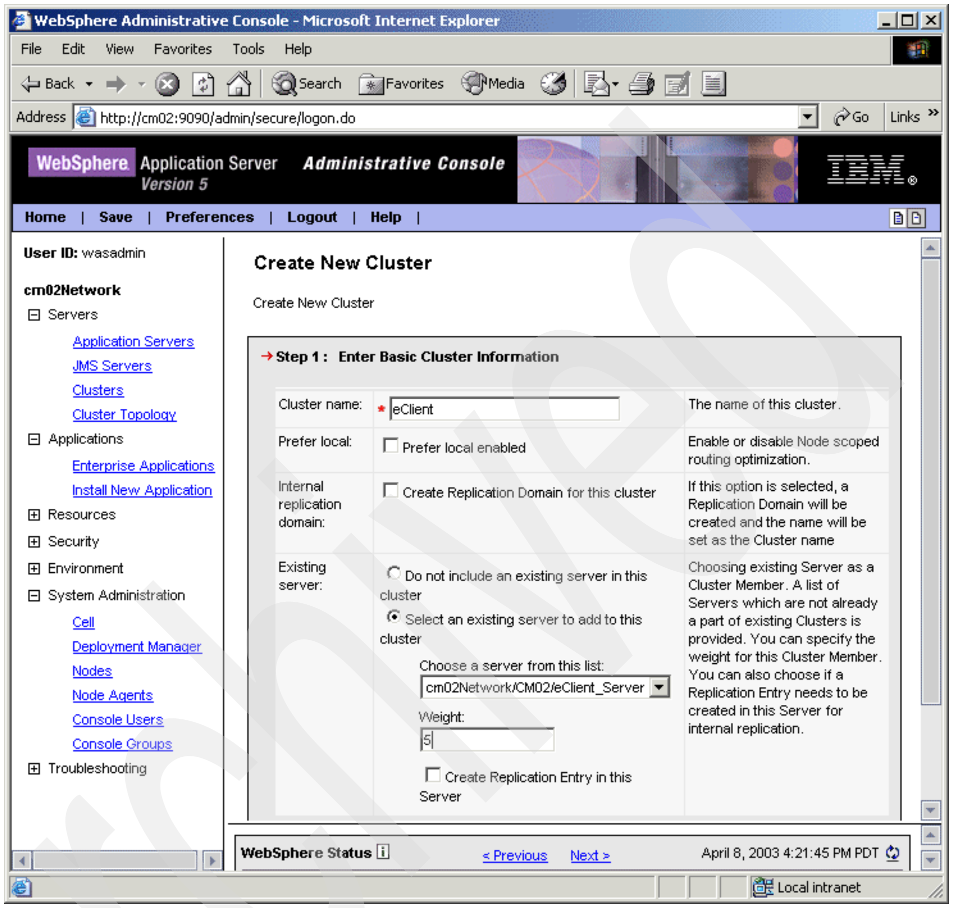
Field	**Value**
Cluster name	eClient
Prefer local	False
Internal replication domain	False
Existing server	Select an existing server to add to this cluster, and select **cm02Network/CM02/eClient_Server** from the server list.
Weight	5
Create replication entry in this server	False

The field Prefer local indicates if a request to an EJB should be routed to an EJB on the local node if available. The field Internal replication domain indicates if you want to use memory-to-memory replication for persistent session management and that a replication domain should be created.

The field Server weight determines how workload is distributed among the cluster members. For example, if all cluster members have identical weights, work will be distributed among the cluster members equally. Servers with higher weight values are given more work. A rule of thumb formula for determining routing preference would be:

```
% routed to Server1 = weight1 /(weight1+weight2+...+weight n)
```

where there are n cluster members in the cluster.


WebSphere Administrative Console - Microsoft Internet Explorer

Address http://cm02:9090/admin/secure/logon.do

User ID: wasadmin

cm02Network

Create New Cluster

Create New Cluster

→ **Step 1 : Enter Basic Cluster Information**

Cluster name:	* eClient	The name of this cluster.
Prefer local:	☐ Prefer local enabled	Enable or disable Node scoped routing optimization.
Internal replication domain:	☐ Create Replication Domain for this cluster	If this option is selected, a Replication Domain will be created and the name will be set as the Cluster name
Existing server:	○ Do not include an existing server in this cluster ● Select an existing server to add to this cluster Choose a server from this list: cm02Network/CM02/eClient_Server Weight: 5 ☐ Create Replication Entry in this Server	Choosing existing Server as a Cluster Member. A list of Servers which are not already a part of existing Clusters is provided. You can specify the weight for this Cluster Member. You can also choose if a Replication Entry needs to be created in this Server for internal replication.

WebSphere Status < Previous Next > April 8, 2003 4:21:45 PM PDT

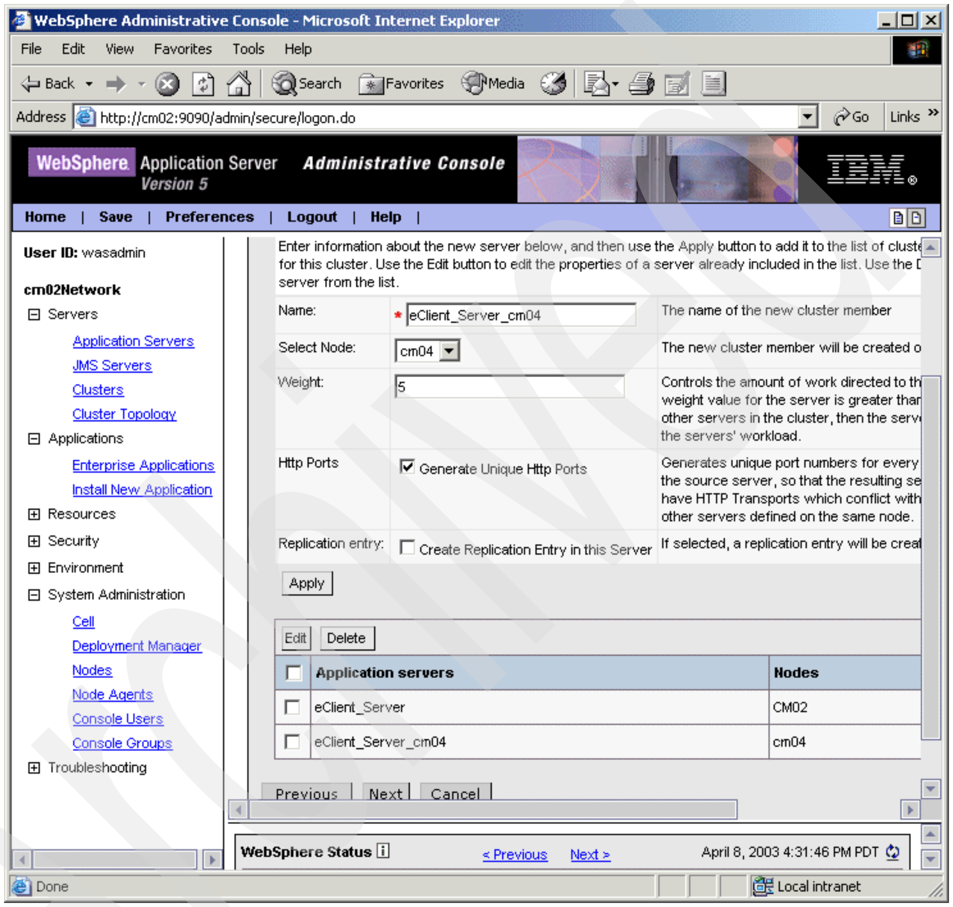
Figure 3-7 Basic cluster information

5. Click **Next** to go to the next window.
6. Set the values for the fields in Table 3-4. See Figure 3-8 on page 71.

Table 3-4 More cluster server information

Field	Value
Name	eClient_Server_cm04
Server Node	cm04
Weight	5

Field	Value
Select Generate Unique Http Ports	True
Create Replication Entry in this Server	False



User ID: wasadmin

Enter information about the new server below, and then use the Apply button to add it to the list of cluste for this cluster. Use the Edit button to edit the properties of a server already included in the list. Use the [server from the list.

Name:	* eClient_Server_cm04	The name of the new cluster member
Select Node:	cm04	The new cluster member will be created o
Weight:	5	Controls the amount of work directed to th weight value for the server is greater thar other servers in the cluster, then the serv the servers' workload.
Http Ports	☑ Generate Unique Http Ports	Generates unique port numbers for every the source server, so that the resulting se have HTTP Transports which conflict with other servers defined on the same node.
Replication entry:	☐ Create Replication Entry in this Server	If selected, a replication entry will be creat

	Application servers	Nodes
☐	eClient_Server	CM02
☐	eClient_Server_cm04	cm04

April 8, 2003 4:31:46 PM PDT

Figure 3-8 More cluster server information

7. Click the **Apply** button to add the new cluster member.
8. If you plan to have more cluster members in the cluster server, repeat step 6 and step 7 for each additional cluster member.
9. After adding all cluster members, click **Next**.
10. Review the cluster summary information, then click **Finish**.
11. Click **Save** in the top-right corner to apply changes to the master configuration.

12. Select **Synchronize changes with Nodes** and click **Save**.

At the beginning of 3.7, “Creating eClient cluster server” on page 68, the directory C:\CmeClient\installedApp\ IBM_eClient_82.ear did not exist on machine cm04. After you created the eClient cluster server, the directory is created and populated with the proper data.

3.7.2 Viewing cluster topology

The WebSphere Administrative Console provides a graphical view of the existing clusters and their members. To see the view:

1. Select **Servers ->Cluster Topology** in the Navigation pane on the left.
2. On the right-hand side, expand the cluster server eClient all to way to the leaf of the tree. The cluster server should have two nodes. One is on cm02. Another one is on cm04. See Figure 3-9.

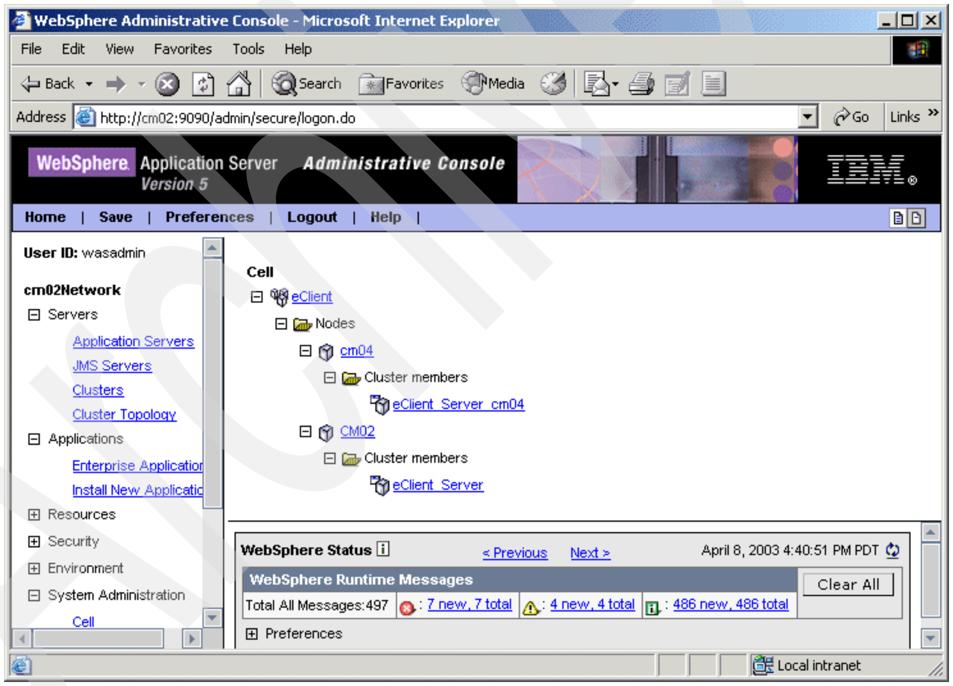


Figure 3-9 Cluster topology

3.7.3 Starting the eClient cluster server

You can start your eClient cluster server in the WebSphere Administrative Console:

1. Select **Servers ->Clusters** in Navigation pane on the left.
2. Select entry **eClient** in the Server Cluster window on the right.
3. Click **Start** to start the cluster server. This starts all cluster members. For our scenario, both server eClient_Server on node cm02 and eClient_Server_04 on node cm04 are started.
4. Select **Servers ->Application Servers** in the Navigation pane on the left. The Application Servers window on the right shows the running status for all servers.

3.8 Configuring HTTP Web server

Since many changes have been made in the WebSphere Application Server Network Deployment environment, you must regenerate the Web server plug-in file and deploy it to the Web server.

1. On the Deployment Manager node (machine cm02), launch a Network Deployment Administrative Console by selecting **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Network Deployment Administrative Console**.
2. Select **Environment -> Update Web Server Plugin** in the Navigation pane on the left.
3. Click **OK** to re-generate the Web server plug-in. The plug-in file is c:\WebSphere\DeploymentManager\config\cells\plugin-cfg.xml and it is on machine cm02.
4. Copy the new generated Web server plug-in file to the Web server machine, which is the machine cm01 for our scenario. The file must be copied to the c:\WebSphere\appServer\config\cells\ directory. You should back up the original plug-in file plugin-cfg.xml before replacing the old plug-in file with the new one.
5. Since the WebSphere directory structure is different on machines cm01 and cm02, you must modify the new generated Web server plug-in file.
   a. Open the file c:\WebSphere\appServer\config\cells\plugin-cfg.xml on machine cm01 in a file editor.
   b. Replace c:\WebSphere\DeploymentManager with c:\WebSphere\appServer in all occurrences.
   c. Save the changes.

6. Stop and restart HTTP Web server on cm01 to make the new plug-in file effective.

3.9 Monitoring workload balance

Up to now, you have installed and configured the following environment:

- The HTTP Web server on machine cm01 with the proper Web server plug-in file for the eClient cluster server in the WebSphere Network Deployment. It is started.
- The WebSphere Application Server Network Deployment on machine cm02 with two nodes in the cell. One node is on machine cm02 and the other one is on machine cm04. The Deployment Manager on machine cm02 and the node agent on each node are started.
- The cluster server eClient has been created in the WebSphere Network Deployment environment. It has two cluster members. One is on node cm02 and the other is on node cm04. The cluster server has been started.
- A Content Manager server has been preinstalled and preconfigured on machine cm03 and is running. Since the Content Manager configuration is beyond the scope of this redbook, we do not cover it here.

In this section, we test the entire system and see how the workload is distributed between nodes cm02 and cm04 in the cell.

3.9.1 Enabling performance monitoring for eClient

Complete the following steps to enable performance monitoring:

1. On the Deployment Manager node (machine cm02), launch a Network Deployment Administrative Console by selecting **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Network Deployment Administrative Console**.
2. Select **Servers -> Application Servers** in the Navigation pane on the left.
3. Select one of the eClient cluster servers, for example **eClient_Server**. This opens its configuration window.
4. Scroll down and select the entry **Performance Monitoring Service**.
5. Set the values for the fields in Table 3-5 on page 75.

Table 3-5 Enabling performance monitoring

Field	Value
Startup	True
Initial specification level	Standard

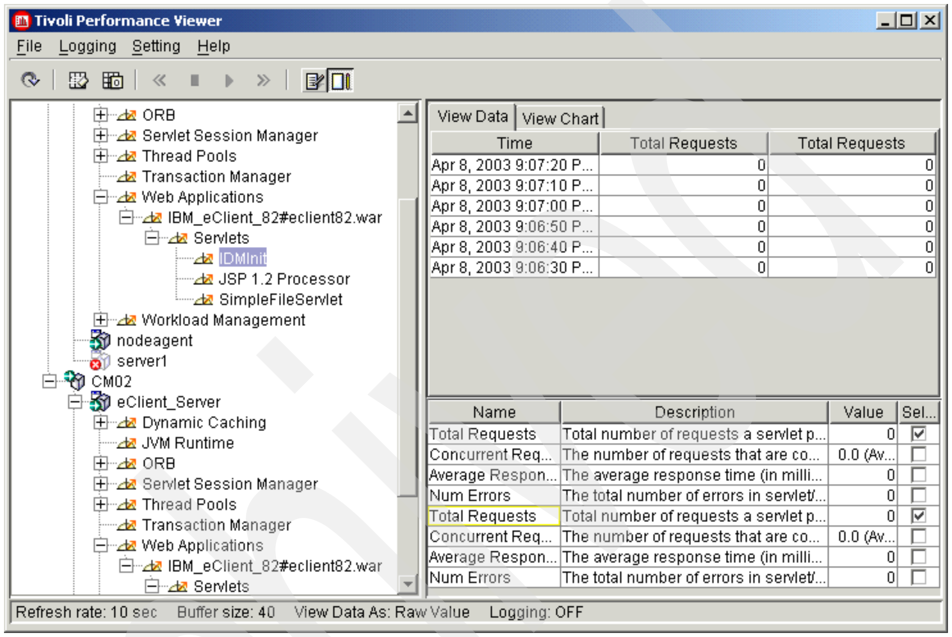
6. Click **OK**.
7. Click **Save** to apply the changes to the master configuration.
8. Select **Synchronize changes with Nodes** and click **Save**.
9. Repeat steps 2 to 8 for the server eClient_Server_cm04.
10. Select **Servers ->Application Servers** in the Navigation pane on the left.
11. Select both **eClient_Server** and **eClient_Server_cm04** and click **Stop**.
12. Select both **eClient_Server** and **eClient_Server_cm04** and click **Start**.

3.9.2 Monitoring workload balance

In this section, we use Tivoli Performance Viewer to monitor how the eClient workload is distributed between the cluster member eClient_Server on node cm02 and eClient_Server_cm04 on node cm04.

1. Start a Tivoli Performance Viewer on the cm02 machine by selecting **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Tivoli Performance Viewer**.
2. If any of your servers in WebSphere Application Server is not running or their performance monitoring feature is disabled, you will receive a PMON3009W warning. Click **OK** on the warning window.
3. In the Navigation pane on the left, expand **Viewer -> cm04 -> eClient_Server_cm04 -> Web Applications -> IBM_eClient_82#eClient82.war -> Servlets -> IDMInit**, and highlight **IDMInit**.
4. In the Navigation pane on the left, expand **Viewer -> cm02 -> eClient_Server -> Web Applications -> IBM_eClient_82#eClient82.war -> Servlets -> IDMInit**, hold the Ctrl key and select **IDMInit**.
5. This highlights the IDMInit from both cm02 and cm04. Notice that on the bottom half of the right-hand pane, there are eight monitoring parameter entries: four from cm02 and another four from cm04.
6. Select only two **Total Requests** entries on the bottom half of the right-hand window. The Total Requests entry keeps track of how many hits to IDMInit servlet were generated on each node. De-select the rest of the entries.

7. Select **Setting ->Clear Buffer** from the menu bar. It clears entries in the top half of the pane on the right. As time passes, new entries are displayed in the top pane. Pay attention to the two Total Requests columns: one is for cm02, the other for cm04. Both of them should have zero counts currently. See Figure 3-10.



Time	Total Requests	Total Requests
Apr 8, 2003 9:07:20 P...	0	0
Apr 8, 2003 9:07:10 P...	0	0
Apr 8, 2003 9:07:00 P...	0	0
Apr 8, 2003 9:06:50 P...	0	0
Apr 8, 2003 9:06:40 P...	0	0
Apr 8, 2003 9:06:30 P...	0	0

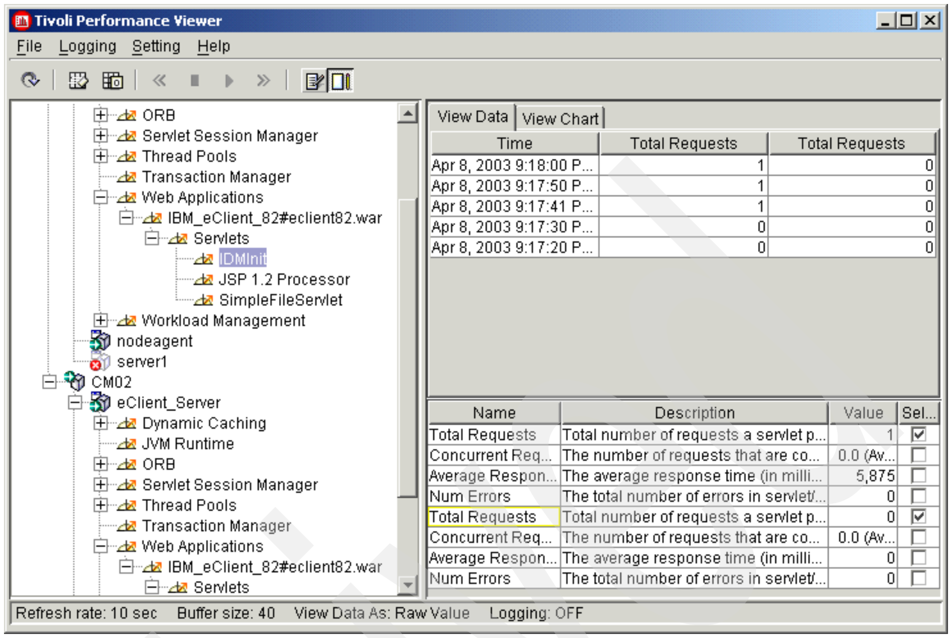
Name	Description	Value	Sel...
Total Requests	Total number of requests a servlet p...	0	✓
Concurrent Req...	The number of requests that are co...	0.0 (Av...	☐
Average Respon...	The average response time (in milli...	0	☐
Num Errors	The total number of errors in servlet/...	0	☐
Total Requests	Total number of requests a servlet p...	0	✓
Concurrent Req...	The number of requests that are co...	0.0 (Av...	☐
Average Respon...	The average response time (in milli...	0	☐
Num Errors	The total number of errors in servlet/...	0	☐

Refresh rate: 10 sec Buffer size: 40 View Data As: Raw Value Logging: OFF

Figure 3-10 Initial status in Tivoli Performance Viewer

8. Open a browser on any machine and enter `http://cm01/eClient82/IDMInit`. This brings up the eClient logon window.

9. In Tivoli Performance Viewer on the cm02 machine, the count in one of the Total Requests column becomes 1, as shown in Figure 3-11 on page 77.



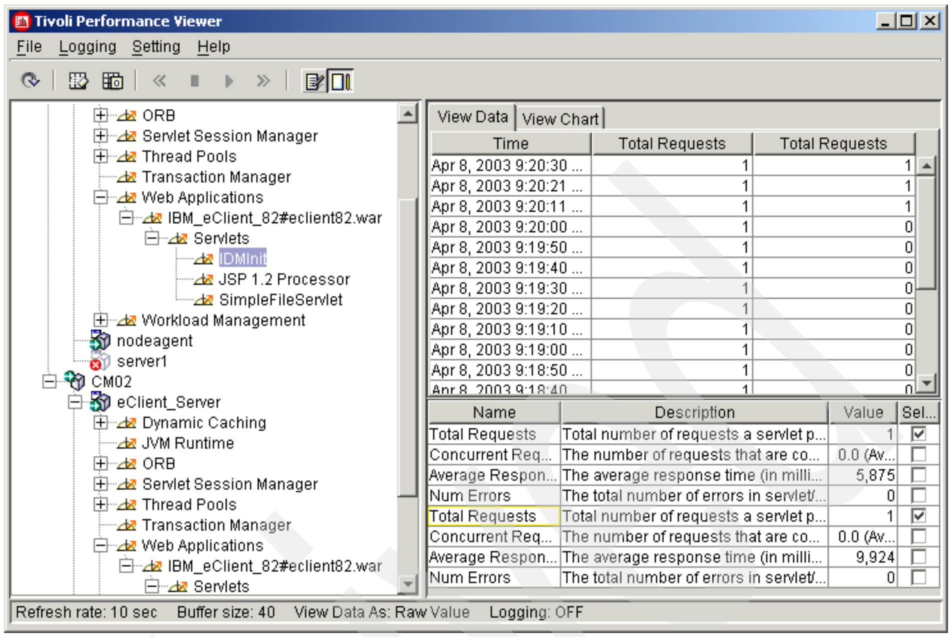
Time	Total Requests	Total Requests
Apr 8, 2003 9:18:00 P...	1	0
Apr 8, 2003 9:17:50 P...	1	0
Apr 8, 2003 9:17:41 P...	1	0
Apr 8, 2003 9:17:30 P...	0	0
Apr 8, 2003 9:17:20 P...	0	0

Name	Description	Value	Sel...
Total Requests	Total number of requests a servlet p...	1	✓
Concurrent Req...	The number of requests that are co...	0.0 (Av...	
Average Respon...	The average response time (in milli...	5,875	
Num Errors	The total number of errors in servlet/...	0	
Total Requests	Total number of requests a servlet p...	0	✓
Concurrent Req...	The number of requests that are co...	0.0 (Av...	
Average Respon...	The average response time (in milli...	0	
Num Errors	The total number of errors in servlet/...	0	

Refresh rate: 10 sec Buffer size: 40 View Data As: Raw Value Logging: OFF

Figure 3-11 After one hit to eClient cluster server

10. Open a browser on another machine and enter `http://cm01/eClient82/IDMInit`.
11. In Tivoli Performance Viewer on the cm02 machine, the count in the other Total Requests column also becomes 1. See Figure 3-12 on page 78.
12. If you open a new browser on the third machine and enter the same URL, the value in first Total Requests column becomes 2. As you can see, the WebSphere Network Deployment uses the round-robin fashion to manage the workload among the cluster members.



```
    ORB
    Servlet Session Manager
    Thread Pools
    Transaction Manager
    Web Applications
      IBM_eClient_82#eclient82.war
        Servlets
          IDMInit
          JSP 1.2 Processor
          SimpleFileServlet
    Workload Management
  nodeagent
  server1
CM02
  eClient_Server
    Dynamic Caching
    JVM Runtime
    ORB
    Servlet Session Manager
    Thread Pools
    Transaction Manager
    Web Applications
      IBM_eClient_82#eclient82.war
        Servlets
```

Time	Total Requests	Total Requests
Apr 8, 2003 9:20:30 ...	1	1
Apr 8, 2003 9:20:21 ...	1	1
Apr 8, 2003 9:20:11 ...	1	1
Apr 8, 2003 9:20:00 ...	1	0
Apr 8, 2003 9:19:50 ...	1	0
Apr 8, 2003 9:19:40 ...	1	0
Apr 8, 2003 9:19:30 ...	1	0
Apr 8, 2003 9:19:20 ...	1	0
Apr 8, 2003 9:19:10 ...	1	0
Apr 8, 2003 9:19:00 ...	1	0
Apr 8, 2003 9:18:50 ...	1	0
Apr 8, 2003 9:18:40	1	0

Name	Description	Value	Sel...
Total Requests	Total number of requests a servlet p...	1	✓
Concurrent Req...	The number of requests that are co...	0.0 (Av...	
Average Respon...	The average response time (in milli...	5,875	
Num Errors	The total number of errors in servlet/...	0	
Total Requests	Total number of requests a servlet p...	1	✓
Concurrent Req...	The number of requests that are co...	0.0 (Av...	
Average Respon...	The average response time (in milli...	9,924	
Num Errors	The total number of errors in servlet/...	0	

Refresh rate: 10 sec   Buffer size: 40   View Data As: Raw Value   Logging: OFF

Figure 3-12 After two hits to eClient cluster server

4

Using eClient

This chapter provides a high-level description of how to use the out-of-the box eClient application. This chapter gets you up to speed with eClient so that in later chapters you can customize and integrate your project.

This chapter covers the following topics:

- Logging on to eClient
- Searching for documents
- Displaying documents
- Importing documents
- Creating folders
- Document routing

4.1 Logging on to eClient

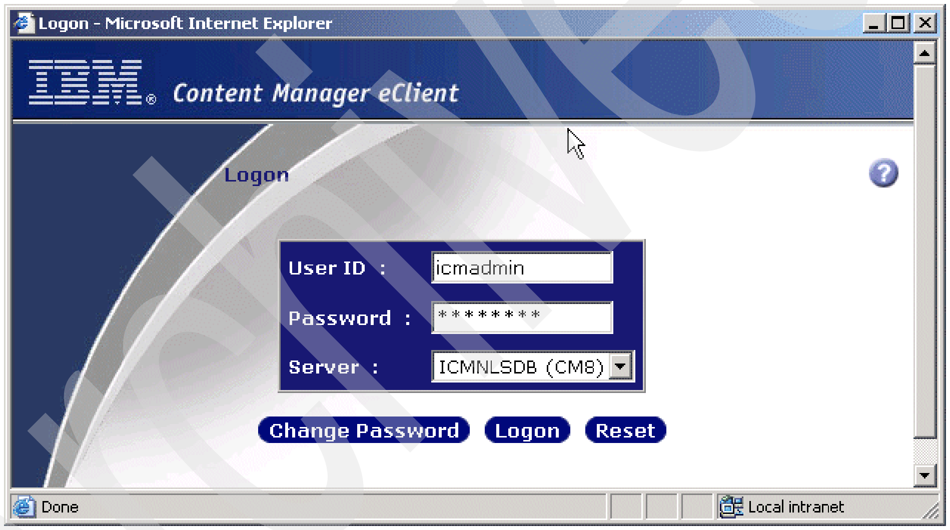
By means of a Web browser, eClient provides the ability for users to log on, search, and display documents stored in one or more EIP back-end repositories. In the examples provided in this chapter, we use Content Manager Version 8 as the back-end datastore.

To log on to the eClient, do the following:

1. Open a Web browser and enter the following URL:

   ```
   http://<hostname>/eClient82/IDMInit
   ```

   This invokes the IDMInit servlet on the Web server, which initializes the application and calls the IDMLogon2.jsp to display the logon window shown in Figure 4-1.


Logon - Microsoft Internet Explorer
IBM Content Manager eClient
Logon
Change Password
Logon
Reset
Done
Local intranet

User ID : icmadmin
Password : ********
Server : ICMNLSDB (CM8)

Figure 4-1 eClient logon window

2. Enter the user ID and password. Select the server you want to log on to. Click **Logon**.

   The eClient validates your logon information and displays the main eClient window, shown in Figure 4-2 on page 81, that allows you to perform various tasks.

4.2 Searching for documents

After you log on to the eClient, the main eClient window shown in Figure 4-2 is displayed.

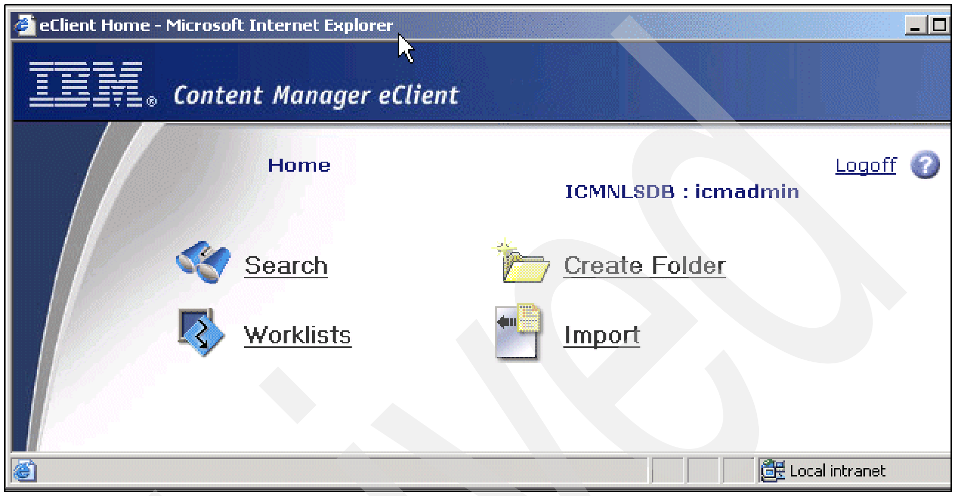


Figure 4-2 Main eClient window

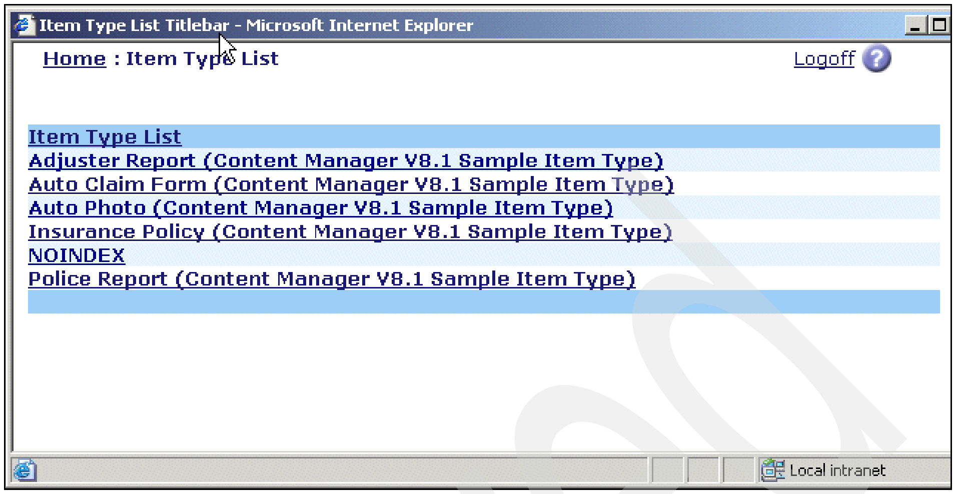
Note that on your system, all of the functions should be available. You can enable or disable some of the functions by changing the values in the IDM.properties file. The IDM.properties file is normally located in c:\Program Files\IBM\CMeClient. The following lines in the properties file control which options are available on the main eClient window:

```
importEnabled=true
createFolderSupported=true
workFlowEnabled=true
```

After changing the values in IDM.properties, you must stop and restart the WebSphere application.

To perform a search, do the following:

1. Click **Search** in the main eClient window. Figure 4-3 on page 82, which allows you to select an item type to search, appears.


Item Type List Titlebar - Microsoft Internet Explorer

Home : Item Type List

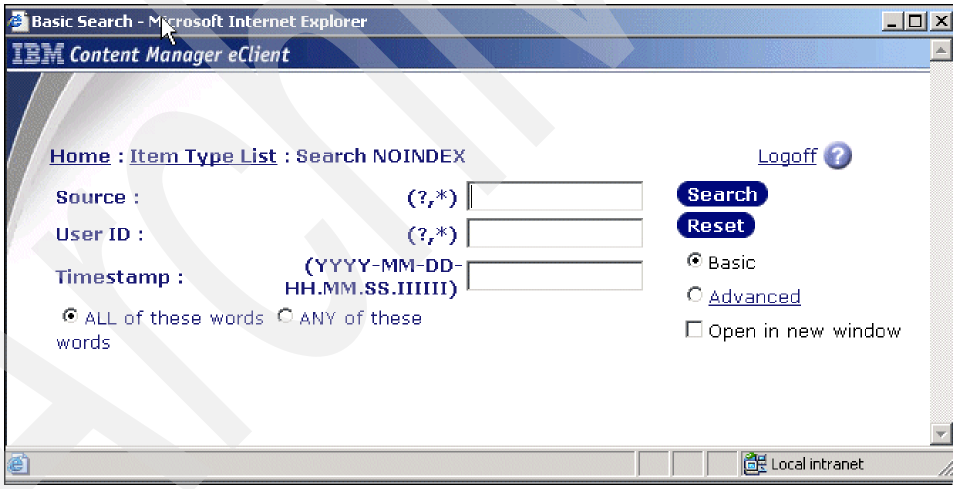
Logoff

Item Type List
Adjuster Report (Content Manager V8.1 Sample Item Type)
Auto Claim Form (Content Manager V8.1 Sample Item Type)
Auto Photo (Content Manager V8.1 Sample Item Type)
Insurance Policy (Content Manager V8.1 Sample Item Type)
NOINDEX
Police Report (Content Manager V8.1 Sample Item Type)

Local intranet

Figure 4-3 Search - Item type list input window

2. Select an item type, such as **NOINDEX**. Figure 4-4, with the attributes for the item type selected, appears.


Basic Search - Microsoft Internet Explorer

IBM Content Manager eClient

Home : Item Type List : Search NOINDEX

Logoff

Source : (?,*)

User ID : (?,*)

Timestamp : (YYYY-MM-DD-HH.MM.SS.IIIIII)

[✓ ALL of these words] ANY of these words

Search

Reset

[✓ Basic]

Advanced

Open in new window

Local intranet

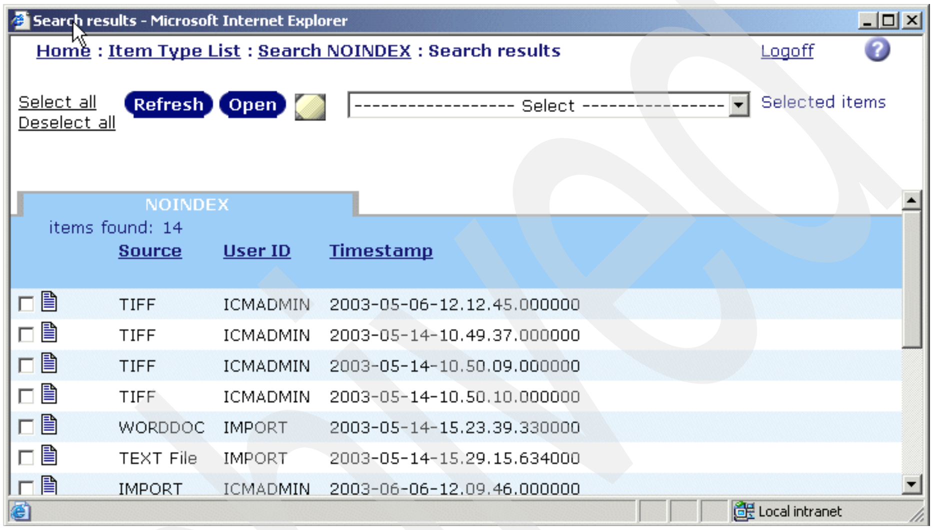
Figure 4-4 Search - Document attributes input window

This search window allows you to specify the search criteria for one or more of the attributes. The (?, *) by the attribute name indicates that wildcard can be used for specific attributes. The ? is a single character wildcard and the * is a multi-character wildcard. The Advanced option adds advanced search operators such as `LIKE, BETWEEN`.

3. Enter your search criteria and click **Search** to view the search results similar to Figure 4-5.

4.2.1 Search results

After performing a search, the search results window shown in Figure 4-5 is displayed.


Search results - Microsoft Internet Explorer
Home : Item Type List : Search NOINDEX : Search results
Logoff
Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items
NOINDEX
Local intranet

items found: 14

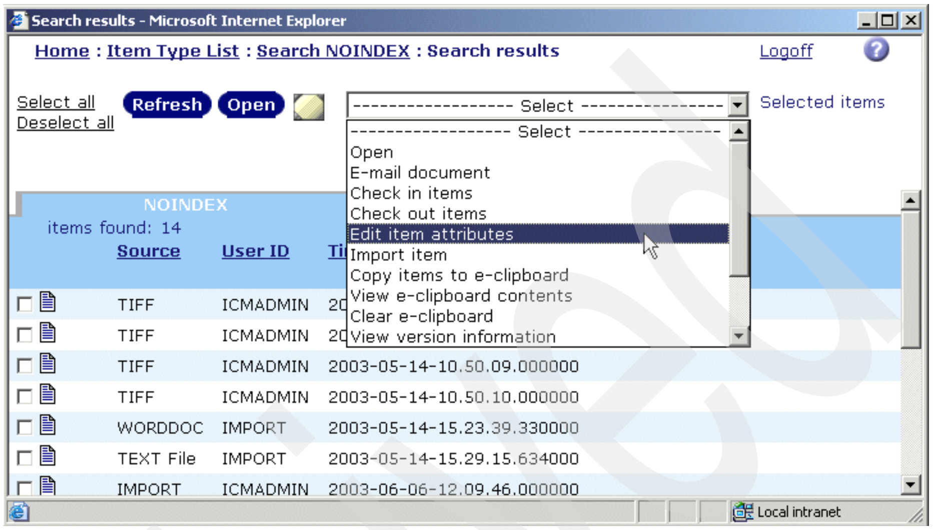
	Source	User ID	Timestamp
☐	TIFF	ICMADMIN	2003-05-06-12.12.45.000000
☐	TIFF	ICMADMIN	2003-05-14-10.49.37.000000
☐	TIFF	ICMADMIN	2003-05-14-10.50.09.000000
☐	TIFF	ICMADMIN	2003-05-14-10.50.10.000000
☐	WORDDOC	IMPORT	2003-05-14-15.23.39.330000
☐	TEXT File	IMPORT	2003-05-14-15.29.15.634000
☐	IMPORT	ICMADMIN	2003-06-06-12.09.46.000000

Figure 4-5 Search - Search results window

A list of documents that met the search criteria is displayed in the window. The ItemType (NOINDEX) is displayed in the tab across the top of the window. A check box, document icon, and document attributes are displayed for each document.

To view a document, click the document icon and the document will be displayed in another window.

There are many things you can do with a document besides view it. The options available are listed in the drop-down box at the top of the search results window shown in Figure 4-6.


Search results - Microsoft Internet Explorer
Home : Item Type List : Search NOINDEX : Search results
Logoff
Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items
------------------ Select ----------------
Open
E-mail document
Check in items
Check out items
Edit item attributes
Import item
Copy items to e-clipboard
View e-clipboard contents
Clear e-clipboard
View version information
NOINDEX
items found: 14
Local intranet

Source	User ID	Ti
TIFF	ICMADMIN	20
TIFF	ICMADMIN	20
TIFF	ICMADMIN	2003-05-14-10.50.09.000000
TIFF	ICMADMIN	2003-05-14-10.50.10.000000
WORDDOC	IMPORT	2003-05-14-15.23.39.330000
TEXT File	IMPORT	2003-05-14-15.29.15.634000
IMPORT	ICMADMIN	2003-06-06-12.09.46.000000

Figure 4-6 Search - Search results drop-down box

There are a number of different options in the drop-down box. These options are actions you can apply to selected (check-marked) documents, as follows:

Open	This option opens the document.
E-mail document	This option allows you to E-mail one or more documents to other users.
Edit item attributes	This option allows the user to re-index a document.
Import item	This option allows you to import a new document into the system.
Copy items to e-clipboard	This option allows you to add documents to a virtual clipboard. This is useful for adding documents to a folder using the eClient.
View e-clipboard contents	This option allows you to view the content in the e-clipboard.
Clear e-clipboard	This option allows you to clear all content in the e-clipboard.
View version information	This option allows you to view version history, if any, for the document.

Start process	This option, which is not shown in Figure 4-6 on page 84, allows you to start the document in a workflow
Process information	This option shows you the current workflow status of the document.

To add your own custom functions in the drop-down box, refer to Chapter 11, "Adding custom functions to the search results window" on page 239.

4.3 Displaying documents

In order to display a document, click the document icon in the search results window, or select one or more documents and select **Open** from the drop-down box at the top of the search results window.

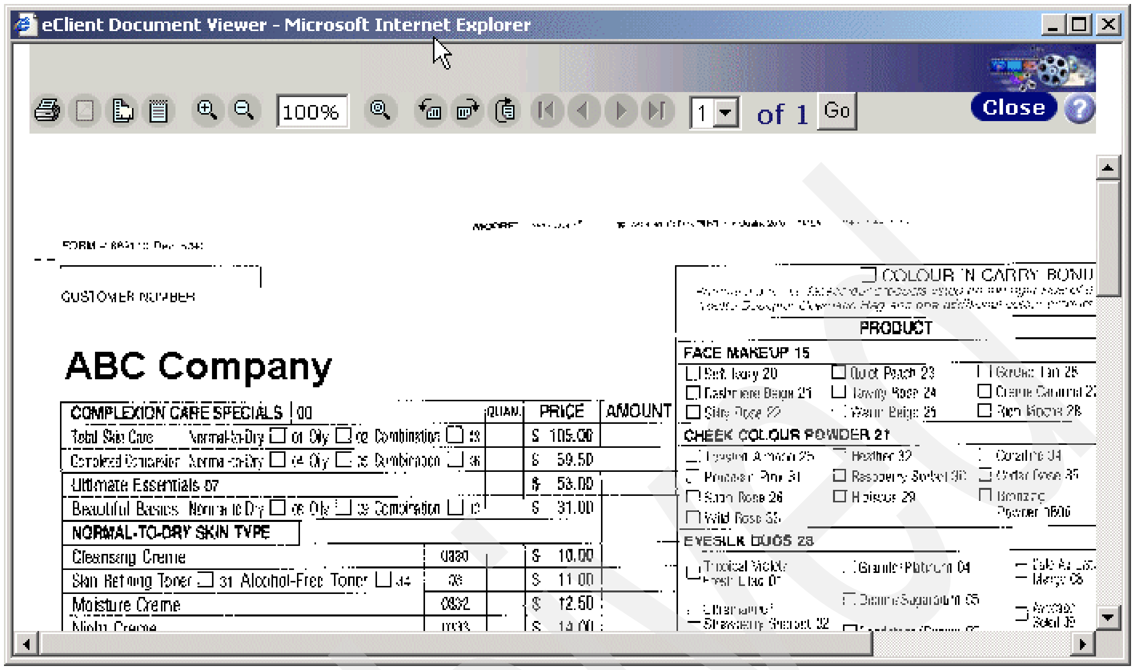
Each document has an associated MIME type that is used to control how the document should be viewed. For example, if you are viewing a Microsoft Word document, then Word should be used to display the document. If the file is a TIFF file, then an image viewer is used.

There are two viewers for TIFF documents included with the eClient: the basic viewer and the applet viewer. We discuss each in the following sections.

Basic viewer

The basic viewer is the default viewer. With this viewer, the eClient server converts a TIFF file to a GIF file first and then sends it down to the browser. This viewer gives you basic image manipulation functions such as change page, zoom, scroll, and rotate.

An example of the basic viewer displaying a document is shown in Figure 4-7 on page 86.



FORM # 8841

CUSTOMER NUMBER

ABC Company

COMPLEXION CARE SPECIALS 00	QUAN.	PRICE	AMOUNT
Total Skin Care Normal-to-Dry ☐ 01 Oily ☐ 02 Combination ☐ 03		$ 105.00	
Complexion Normal-to-Dry ☐ 04 Oily ☐ 05 Combination ☐ 06		$ 59.50	
Ultimate Essentials 07		$ 53.00	
Beautiful Basics Normal-to-Dry ☐ 08 Oily ☐ 09 Combination ☐ 10		$ 31.00	
NORMAL-TO-DRY SKIN TYPE			
Cleansing Creme	0830	$ 10.00	
Skin Refining Toner ☐ 31 Alcohol-Free Toner ☐ 34	[illegible]	$ 11.00	
Moisture Creme	0832	$ 12.50	
Night Creme	0833	$ 14.00	

☐ COLOUR 'N CARRY BONU

PRODUCT

FACE MAKEUP 15

☐ Soft Ivory 20	☐ Quiet Peach 23	☐ Golden Tan 26
☐ Cashmere Beige 21	☐ Tawny Rose 24	☐ Creme Caramel 27
☐ Silky Rose 22	☐ Warm Beige 25	☐ Rich Bronze 28

CHEEK COLOUR POWDER 21

☐ Crystal Almond 25	☐ Heather 32	☐ Coralline 34
☐ Precious Pink 31	☐ Raspberry Sorbet 30	☐ Charter Rose 35
☐ Satin Rose 26	☐ Hibiscus 29	☐ Bronzing Powder 1606
☐ Wild Rose 33		

EYESILK DUOS 28

☐ Tropical Violet/Fresh Lilac 01	☐ Granite/Platinum 04	☐ [illegible]
☐ [illegible]	☐ Creme/Sugarplum 05	☐ [illegible]
☐ Strawberry Sherbet 32	☐ [illegible]	

Figure 4-7 Default image viewer

The toolbar on the basic viewer window provides the basic image manipulation functions and also the document attributes access and printing functions.

Applet viewer

The applet viewer has more functionality than the basic viewer. Instead of having the eClient server converting a TIFF file to a GIF for viewing, the applet viewer displays TIFF files and other file formats on the client. When a user requests a rotate, the rotation of the document is done on the client machine; unlike the basic viewer, it does not go back to the server to perform the action.

The applet viewer also supports image annotations such as sticky notes, highlights, stamps, and text overlays.

To enable the applet viewer, you must modify the following two files:

- IDM.properties
- IDMadminDefaults.properties

For the IDM.properties file, set viewerAppletEnabled parameter to true as follows:

```
viewerAppletEnabled=true
```

For the IDMadminDefaults.properties file (located in the c:\program files\IBM\CMeClient directory), specify which viewer to use for which MIME type. Example 4-1 shows a sample of the file.

Example 4-1 Sample IDMadminDefaults.properties file

```
## To indicate that a mimeType should be converted to a viewable
## format specify "don't launch".
## Use don't launch to have the image rendered on the mid-tier
## application server
##
## example: application/vnd.modcap=don't launch
##
## Format:      [MIME_TYPE]={ launch | applet | don't launch }


application/afp=launch
application/pdf=launch
application/vnd.ibm.modcap=don't launch
application/x-rtsp=launch
audio/basic=launch
audio/mpeg=launch
image/gif=don't launch
image/jpeg=don't launch
image/tiff=applet
text/html=launch
text/plain=don't launch
text/xml=launch
video/mpeg=launch
video/quicktime=launch
video/x-ibm-ivs=launch
```

As shown in Example 4-1, to view TIFF files with the applet viewer, set the image/tiff MIME type to applet:

```
image/tiff=applet
```

Stop and restart WebSphere to make the change effective. The next time you display a TIFF file from the eClient, the applet viewer is used. Figure 4-8 on page 88 is an example.

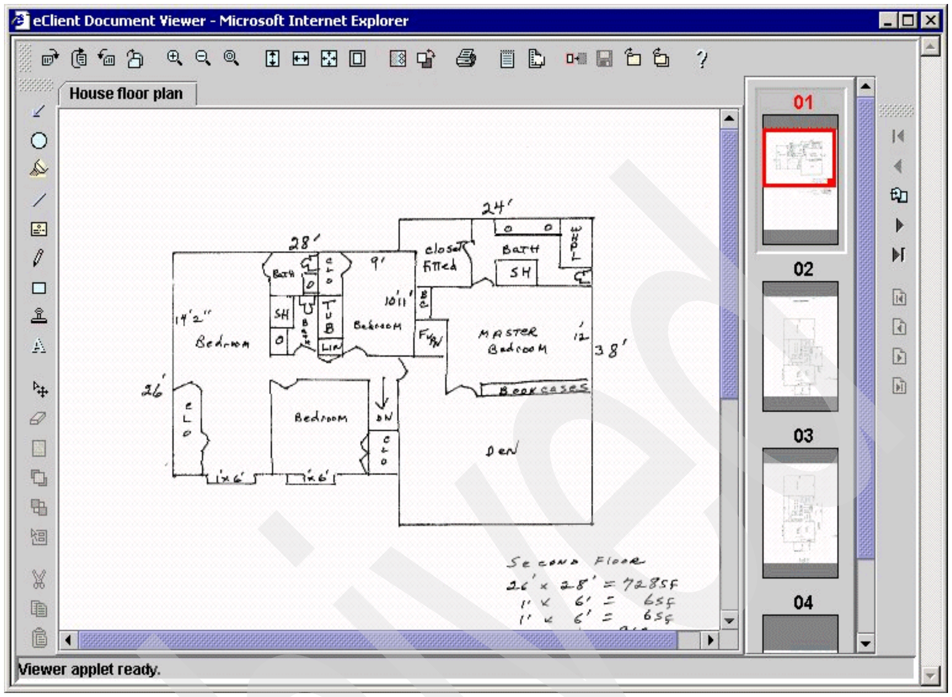


Figure 4-8   Applet viewer

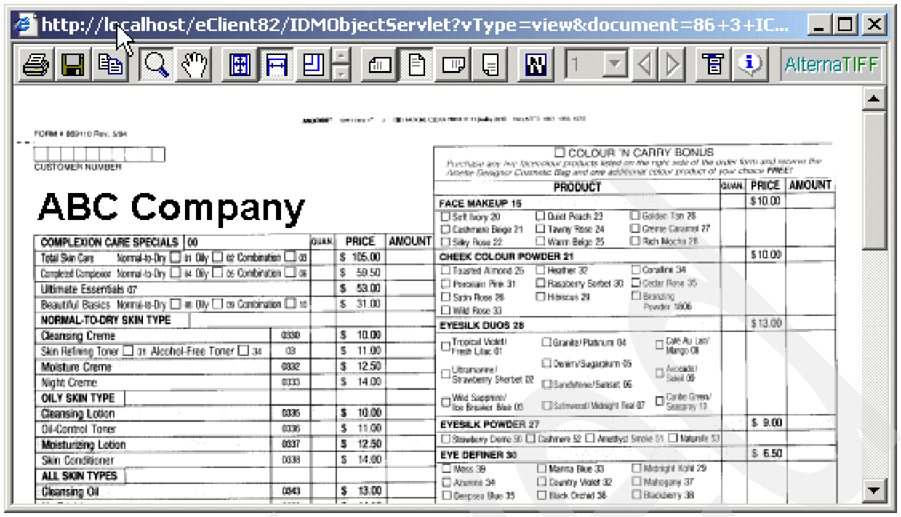
TIFF plug-in viewer

If you installed a TIFF plug-in for your browser, you need to modify the IDMadminDefaults.properties file before using it. Set the image/tiff MIME type to launch in the file:

```
image/tiff=launch
```

With this setup, the eClient sends the TIFF file to the browser and the application tool that is configured on your browser is used to display the file.

For example, you can download a free TIFF plug-in from http://www.alternatiff.com, install it in your Internet Explorer, modify the line above in the IDMadminDefaults.properties file, restart WebSphere, and display a TIFF document that looks similar to Figure 4-9 on page 89.


http://localhost/eClient82/IDMObjectServlet?vType=view&document=86+3+IC...

FORM # 869110 Rev. 5/94

CUSTOMER NUMBER

ABC Company

COMPLEXION CARE SPECIALS 00		QUAN	PRICE	AMOUNT
Total Skin Care Normal-to-Dry ☐ 01 Oily ☐ 02 Combination ☐ 03			$ 105.00	
Completed Complexion Normal-to-Dry ☐ 04 Oily ☐ 05 Combination ☐ 06			$ 59.50	
Ultimate Essentials 07			$ 53.00	
Beautiful Basics Normal-to-Dry ☐ 08 Oily ☐ 09 Combination ☐ 10			$ 31.00	
NORMAL-TO-DRY SKIN TYPE				
Cleansing Creme	0030		$ 10.00	
Skin Refining Toner ☐ 31 Alcohol-Free Toner ☐ 34	03		$ 11.00	
Moisture Creme	0032		$ 12.50	
Night Creme	0033		$ 14.00	
OILY SKIN TYPE				
Cleansing Lotion	0035		$ 10.00	
Oil-Control Toner	0036		$ 11.00	
Moisturizing Lotion	0037		$ 12.50	
Skin Conditioner	0038		$ 14.00	
ALL SKIN TYPES				
Cleansing Oil	0043		$ 13.00	

☐ COLOUR 'N CARRY BONUS

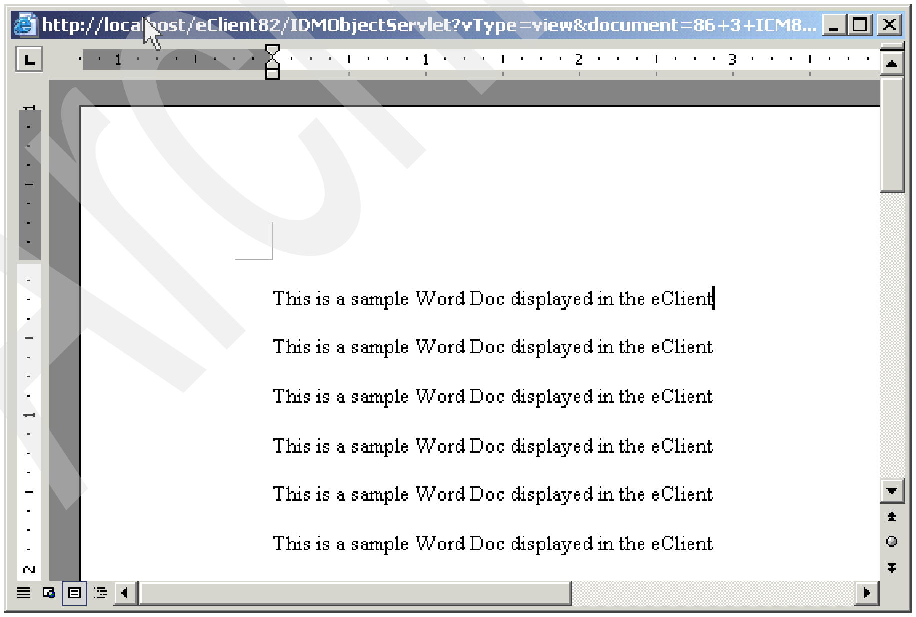
Purchase any two [illegible] products listed on the right side of the order form and receive the [illegible] Designer Cosmetic Bag and one additional colour product of your choice FREE!

PRODUCT	QUAN	PRICE	AMOUNT
FACE MAKEUP 15		$ 10.00	
☐ Soft Ivory 20 ☐ Quiet Peach 23 ☐ Golden Tan 26 ☐ Cashmere Beige 21 ☐ Tawny Rose 24 ☐ Creme Caramel 27 ☐ Silky Rose 22 ☐ Warm Beige 25 ☐ Rich Mocha 28			
CHEEK COLOUR POWDER 21		$ 10.00	
☐ Toasted Almond 25 ☐ Heather 32 ☐ Coraline 34 ☐ Porcelain Pink 31 ☐ Raspberry Sorbet 30 ☐ Cedar Rose 35 ☐ Satin Rose 26 ☐ Hibiscus 29 ☐ Bronzing Powder 1806 ☐ Wild Rose 33			
EYESILK DUOS 28		$ 13.00	
☐ Tropical Violet/ Fresh Lilac 01 ☐ Granite/Platinum 04 ☐ Café Au Lait/ Mango 08 ☐ Denim/Sugarplum 05 ☐ Ultramarine/ Strawberry Sherbet 02 ☐ Avocado/ Soleil 09 ☐ Sandstone/Sunset 06 ☐ Wild Sapphire/ Ice Breaker Blue 03 ☐ Softwood/Midnight Teal 07 ☐ Caribe Green/ Seaspray 10			
EYESILK POWDER 27		$ 9.00	
☐ Strawberry Creme 50 ☐ Cashmere 52 ☐ Amethyst Smoke 51 ☐ Naturelle 53			
EYE DEFINER 30		$ 6.50	
☐ Moss 39 ☐ Marina Blue 33 ☐ Midnight Kohl 29 ☐ Aramina 34 ☐ Country Violet 32 ☐ Mahogany 37 ☐ Deepsea Blue 35 ☐ Black Orchid 36 ☐ Blackberry 38			

Figure 4-9 TIFF document displayed with AlternaTIFF plug-in

Displaying office documents

By default, office documents such as Microsoft Word documents are "launched" in the browser. Figure 4-10 is an example of what a Word document looks like when viewed from the eClient.


http://localhost/eClient82/IDMObjectServlet?vType=view&document=86+3+ICM8...

This is a sample Word Doc displayed in the eClient

This is a sample Word Doc displayed in the eClient

This is a sample Word Doc displayed in the eClient

This is a sample Word Doc displayed in the eClient

This is a sample Word Doc displayed in the eClient

This is a sample Word Doc displayed in the eClient

Figure 4-10 Displaying Microsoft Word document in eClient

4.4 Importing documents

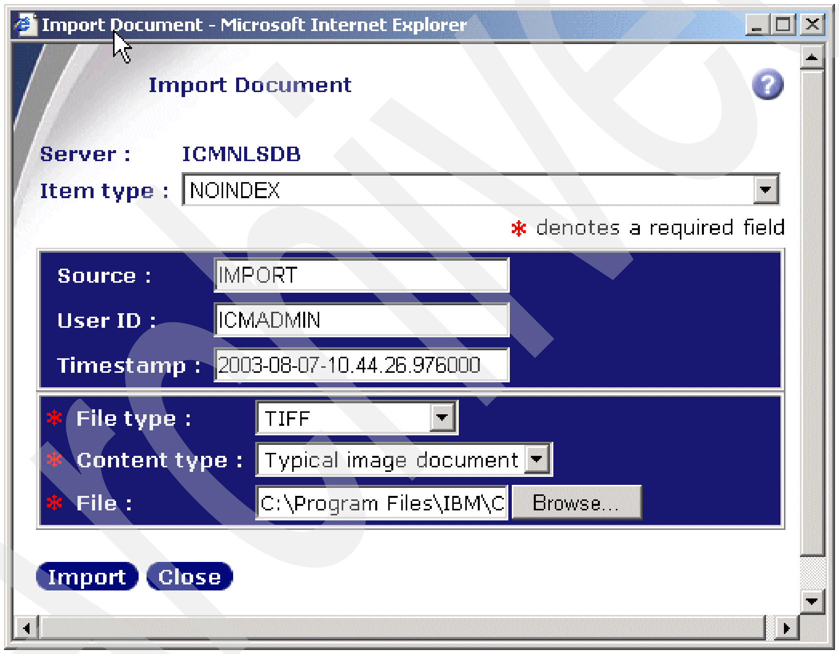
You can import documents from a file system using the eClient. From the main eClient window, use the Import option to achieve this. If the Import option is not available, set the importSupported parameter to true in the IDM.properties file. If the entry does not exist, add it to the file:

```
importSupported=true
```

Stop and restart WebSphere to make the change effective.

To import documents, do the following:

1. From the main eClient window, click **Import**. A window similar to Figure 4-11 appears.

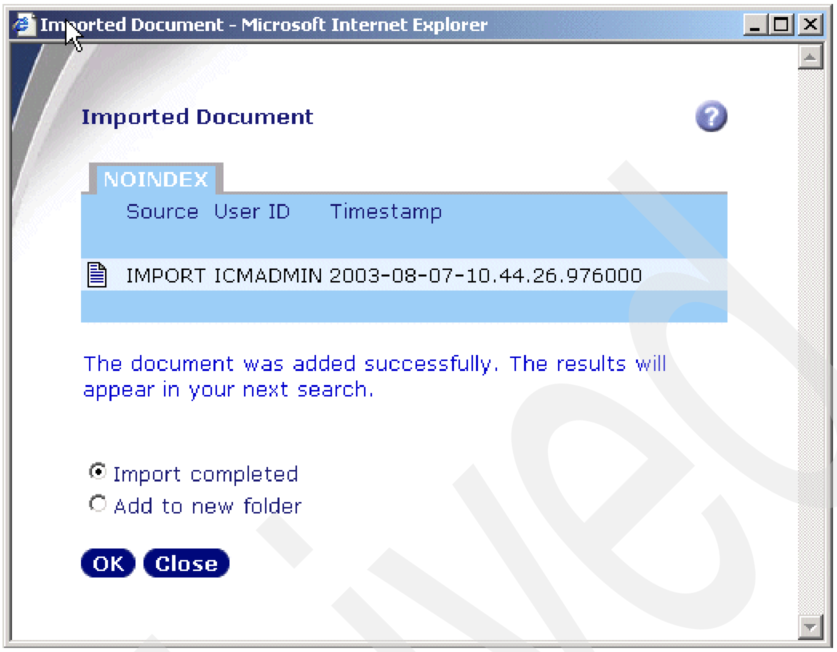


Server : ICMNLSDB
Item type : NOINDEX
* denotes a required field
Source : IMPORT
User ID : ICMADMIN
Timestamp : 2003-08-07-10.44.26.976000
* File type : TIFF
* Content type : Typical image document
* File : C:\Program Files\IBM\C

Figure 4-11 Import - Import input window

2. Select an item type from the drop-down box for the document you want to import.
3. Enter the index values and select the file type of the document you are importing. You can click **Browse** to find a file on your local or network drive to import. When ready, click **Import**.

   After importing the document, a window similar to Figure 4-12 appears.



Source	User ID	Timestamp
IMPORT	ICMADMIN	2003-08-07-10.44.26.976000

The document was added successfully. The results will appear in your next search.

[✓ Import completed] Add to new folder

Figure 4-12 Import - Import complete window

4. Click **OK** to complete the import, or you can add the document to a folder. If you want to add it to a folder, select **Add to new folder** and click **OK**.
5. Select an item type for the folder and enter the folder attributes.

4.5 Creating folders

This section shows you how to create a folder and add documents to it using the eClient. The main eClient window should have a Create Folder option on it. If your eClient does not have this option, set the CreateFolderEnabled parameter to true in the IDM.properties file as follows:

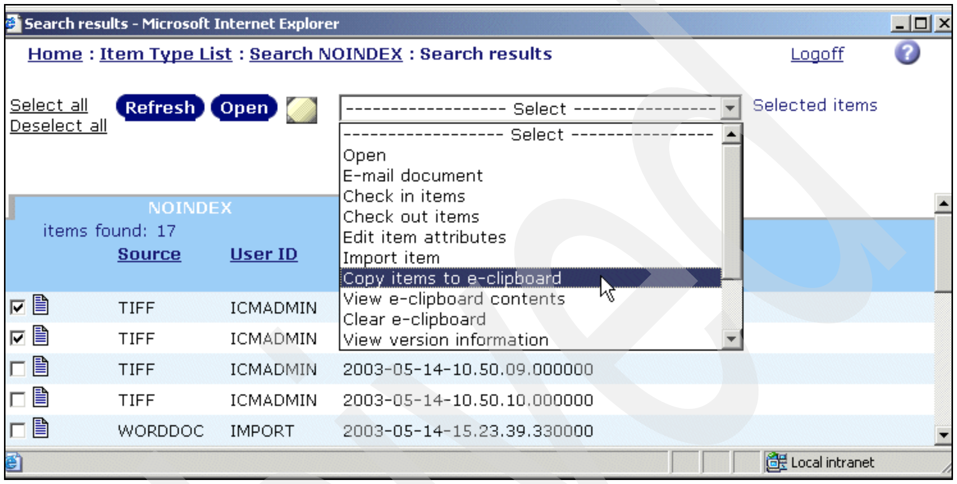
```
CreateFolderEnabled=true
```

Restart WebSphere to make the change effective.

To create a folder and put documents in the folder, do the following:

1. From the main eClient window, click **Create Folder**. A window that asks you for the item type and attributes for the folder appears.
2. Select an item type and specify the correct attribute values.

3. After the folder is created, use the search function to locate the documents you want to add to the folder.
4. From the search results window, select the document and then use **Copy items to e-clipboard** in the search results drop-down box to add the documents to the e-clipboard, as shown in Figure 4-13.



✓	TIFF	ICMADMIN	
✓	TIFF	ICMADMIN	
	TIFF	ICMADMIN	2003-05-14-10.50.09.000000
	TIFF	ICMADMIN	2003-05-14-10.50.10.000000
	WORDDOC	IMPORT	2003-05-14-15.23.39.330000

Figure 4-13 Folders - Copy selected documents to e-clipboard

5. Use the search function to find the new folder you created.
6. Check the box next to the folder and click **Paste items from e-clipboard to folder** from the search results drop-down box, as shown in Figure 4-14 on page 93.

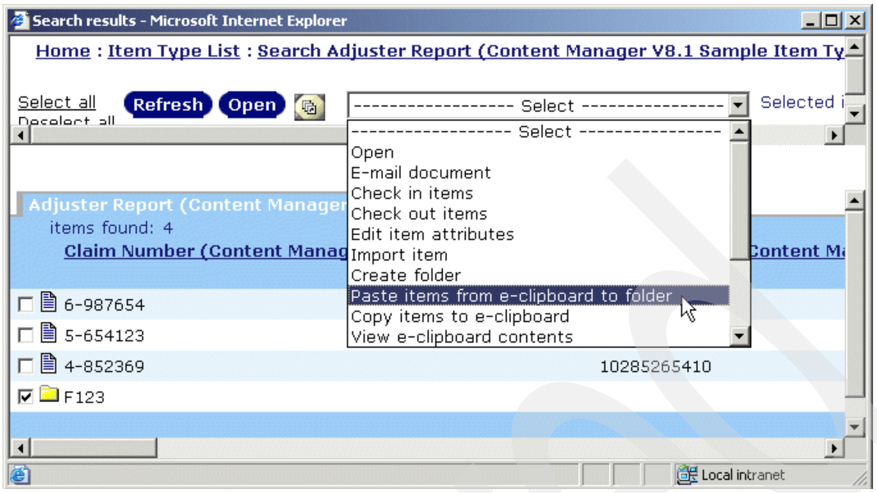

Search results - Microsoft Internet Explorer
Home : Item Type List : Search Adjuster Report (Content Manager V8.1 Sample Item Ty
Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected i
Open
E-mail document
Check in items
Check out items
Edit item attributes
Import item
Create folder
Paste items from e-clipboard to folder
Copy items to e-clipboard
View e-clipboard contents
Adjuster Report (Content Manager
items found: 4
Claim Number (Content Manag
Content Ma
6-987654
5-654123
4-852369
10285265410
F123
Local intranet

Figure 4-14 Folders - Paste items from e-clipboard

The documents are now contained in the folder you just created.

4.6 Document routing

The Content Manager Version 8 System Administration client allows you to define a simple workflow for document routing. A *Node* is a single step in a workflow, a *process* links different nodes into a workflow process, and a *worklist* is used to gather work items from one or more nodes into a list of documents that a user can view. For more information on building workflow nodes, processes, and worklists, refer to *IBM Content Manager for Multiplatforms: System Administration Guide*, SC27-1335.

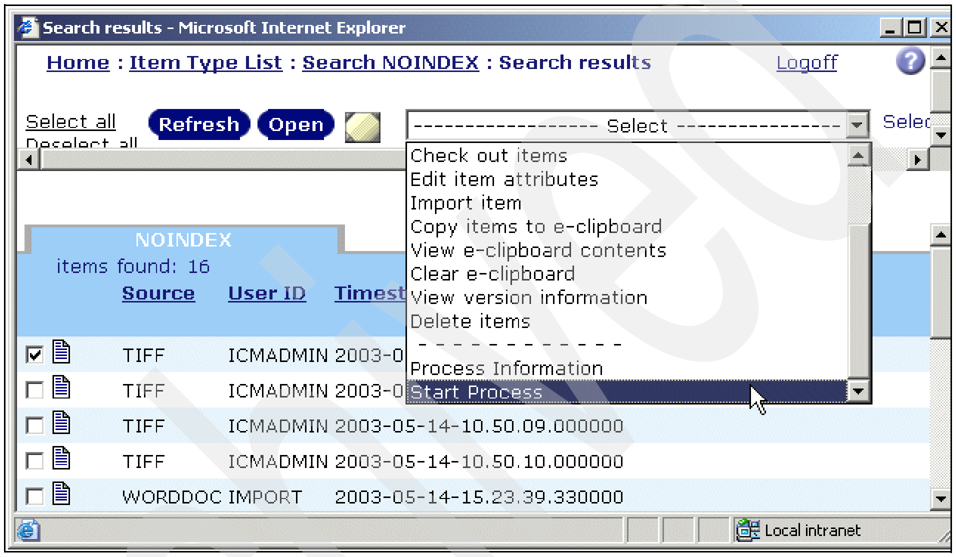
The workflow you create with the Content Manager System Administration client application can be accessed from both the Content Manager Windows (thick) client and from the eClient.

Note that the eClient and Content Manager support both the Document Routing and Advanced Workflow functions. In June 2003, the Content Manager lab announced that the Advanced Workflow function would likely undergo significant changes in the future. If you plan to use Advanced Workflow, check with your Content Manager representative to get the latest recommendations before pursuing this path.

4.6.1 Adding documents to a workflow process

To add a document to a workflow, do the following:

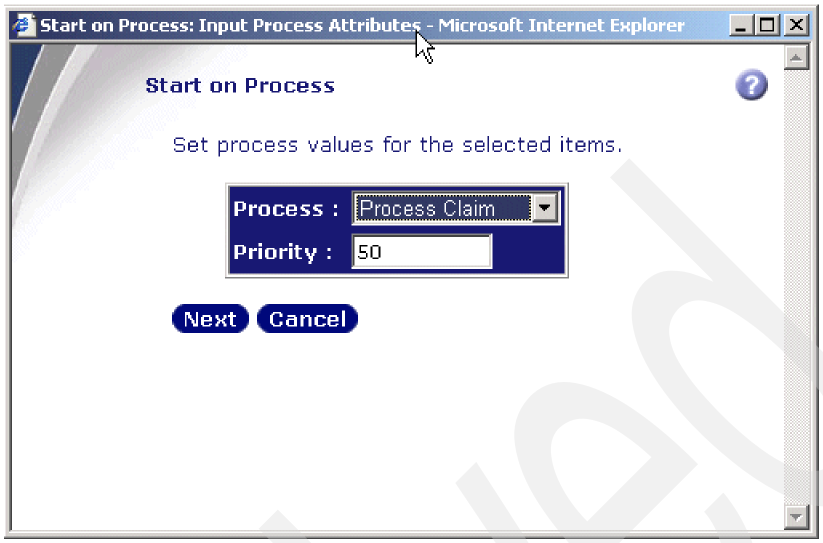
1. Perform a document search.
2. Select one or more documents from the search results set.
3. Select **Process document** from the search results drop-down box (Figure 4-15).



	Source	User ID	Timest
✓	TIFF	ICMADMIN	2003-0
	TIFF	ICMADMIN	2003-0
	TIFF	ICMADMIN	2003-05-14-10.50.09.000000
	TIFF	ICMADMIN	2003-05-14-10.50.10.000000
	WORDDOC	IMPORT	2003-05-14-15.23.39.330000

Figure 4-15 Document routing - Start document in a workflow process

4. Select **Start Process**. A window similar to Figure 4-16 on page 95, that allows users to select which process to start for the selected documents, appears.


Start on Process: Input Process Attributes - Microsoft Internet Explorer

Start on Process

Set process values for the selected items.

Process : Process Claim

Priority : 50

Figure 4-16 Document routing - Select process for document

5. A list of workflow processes is displayed in the process drop-down box.Select the correct process and the priority. Click **Next**.

After adding the document to a workflow, the document is inserted into the first node of the workflow. To access the document in the workflow nodes, use a worklist. A worklist can include documents from a single node or multiple nodes, and can prioritize the documents from different nodes.

A worklist can be accessed from the main eClient window as shown in Figure 4-17 on page 96.

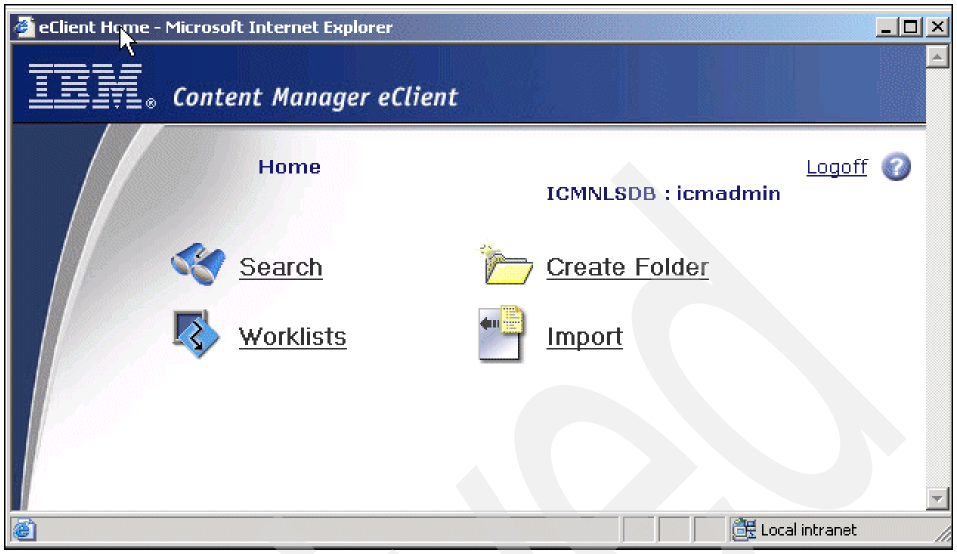


Figure 4-17 Main eClient window with access to worklists

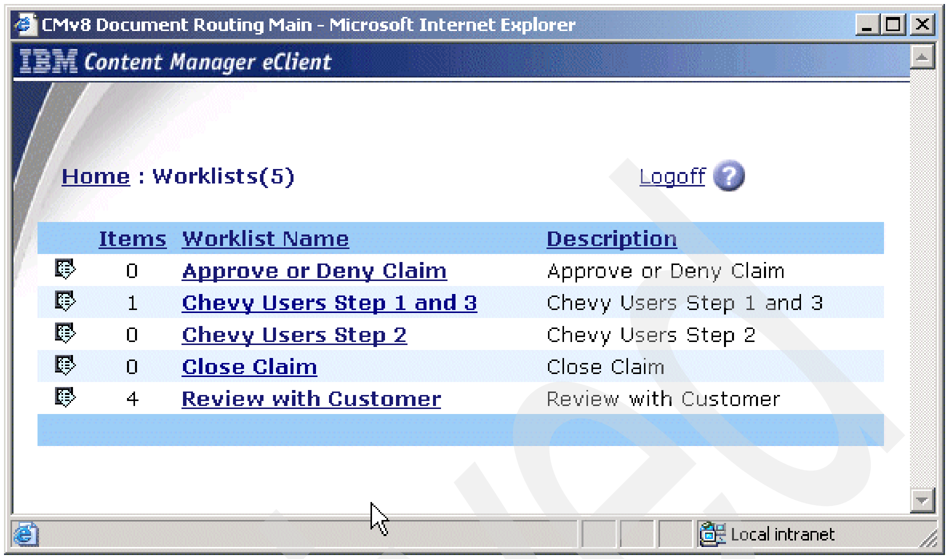
If your eClient does not show the Worklists option, check the IDM.properties file and make sure that the workFlowEnabled parameter is set to true as follows:

```
workFlowEnabled=true
```

After changing the IDM.properties file, restart WebSphere to make the change effective.

To work with worklists, do the following:

1. From the main eClient window, click **Worklists**. A window similar to Figure 4-18 on page 97 appears.



CMv8 Document Routing Main - Microsoft Internet Explorer

IBM Content Manager eClient

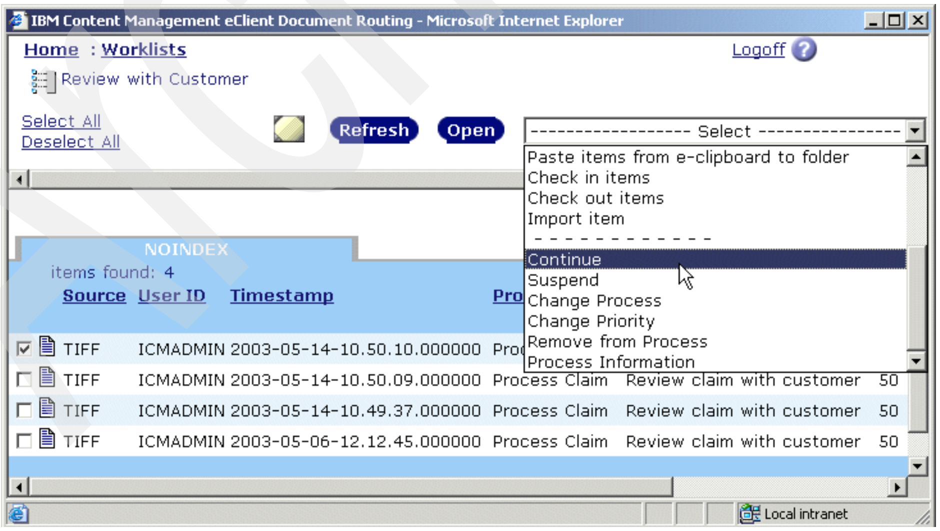
Home : Worklists(5) Logoff

Items	Worklist Name	Description
0	Approve or Deny Claim	Approve or Deny Claim
1	Chevy Users Step 1 and 3	Chevy Users Step 1 and 3
0	Chevy Users Step 2	Chevy Users Step 2
0	Close Claim	Close Claim
4	Review with Customer	Review with Customer

Local intranet

Figure 4-18 Worklist - Worklist selection window

2. Select the worklist you want to work with. In our sample workflow, the Review with Customer worklist displays documents in the first node. Click **Review with Customer** to obtain a list of documents in the worklist. A window appears that shows a list of workflow actions that can be performed on a document in the worklist (see Figure 4-19).



IBM Content Management eClient Document Routing - Microsoft Internet Explorer

Home : Worklists Logoff

Review with Customer

Select All
Deselect All

Refresh Open

------------------ Select ------------------
Paste items from e-clipboard to folder
Check in items
Check out items
Import item
- - - - - - - - - - - -
Continue
Suspend
Change Process
Change Priority
Remove from Process
Process Information

NOINDEX

items found: 4

	Source	User ID	Timestamp	Pro		
☑	TIFF	ICMADMIN	2003-05-14-10.50.10.000000	Proc		
☐	TIFF	ICMADMIN	2003-05-14-10.50.09.000000	Process Claim	Review claim with customer	50
☐	TIFF	ICMADMIN	2003-05-14-10.49.37.000000	Process Claim	Review claim with customer	50
☐	TIFF	ICMADMIN	2003-05-06-12.12.45.000000	Process Claim	Review claim with customer	50

Local intranet

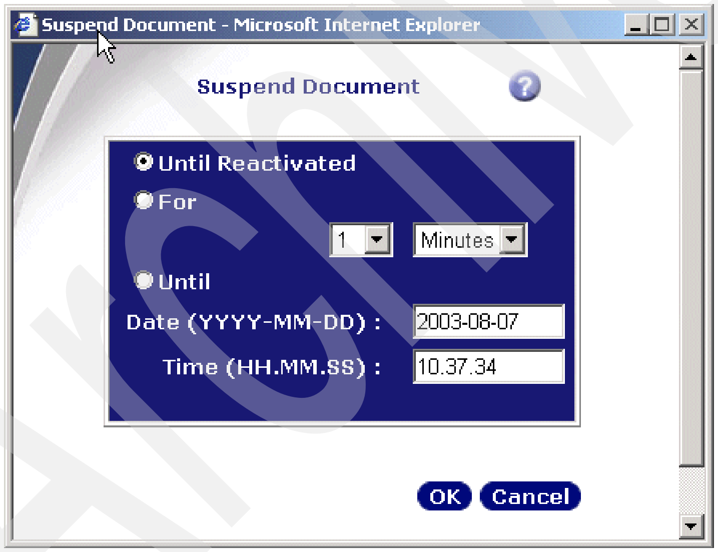
Figure 4-19 Worklist - Action options

3. Select the document you want to process, and select an action you want to perform on the document from the drop-down box to process the document.

From the default example we provided, there are several actions you can perform for the selected documents:

Continue	Moves the document to the next step in the process.
Suspend	Removes the document from the worklist for a specified time.
Change Process	Moves the document to a different workflow process.
Change Priority	Changes the priority of the document.
Remove from Process	Removes the document from the workflow process.

Sometimes, you need to wait for several days before you can continue processing a particular document. In this case, you may want to use the Suspend action on the document. You need to enter additional information if you suspend any documents. See Figure 4-20 for input required.



Date (YYYY-MM-DD) : 2003-08-07

Time (HH.MM.SS) : 10.37.34

Figure 4-20 Worklist - Suspend document input window

Part 2

Preparing for eClient customization and integration

This part prepares you for eClient customization and integration. We include an eClient architectural overview and an inspection of a basic eClient control flow. We also demonstrate the usage of APIs with many sample codes, and provide step-by-step instructions on setting up a development environment for eClient customization and integration. In addition, we provide design and implementation considerations for eClient application customization and integration.

5

eClient architecture

eClient architecture is based on the J2EE standard. This chapter describes the standard and the appropriate technologies used in eClient. We also describe the eClient application structure, and provide a detailed explanation of one core function and an introduction to the customization and the integration of eClient.

5.1 Introducing J2EE

J2EE stands for Java 2 Platform, Enterprise Edition. It defines a standard that applies to all aspects of architecture and developing multi-tier server-based applications.

J2EE applications are made up of components where each component is a self-contained functional software unit. A component has its related classes and files and communicates with other components.

Four types of components are defined:

- **Application clients**: Java programs that execute on a client machine and access other J2EE components.
- **Applets**: Java components that execute on a client machine within a browser.
- **Web components**: Servlets and JavaServer Pages (JSP) components that run on the server and provide the controller and view functionality.
- **Enterprise JavaBeans** (EJBs): Components that run on the server for business logic and database access.

J2EE components are written in the Java programming language and are compiled in the same way as any Java program. In addition, J2EE components are assembled into a J2EE application, verified to be compliant with the J2EE specification, and deployed to the J2EE server, where they are managed and run.

To manage and run the components of a J2EE application, the J2EE server defines four types of containers:

- **Application client container**, a stand-alone Java runtime environment
- **Applet container**, provided by the browser
- **Web component container**, provided by the J2EE application server
- **Enterprise JavaBeans container**, provided by the J2EE application server

The components use the services provided by the container either by implementing interfaces or by calling certain APIs as defined in the J2EE specification. J2EE components never interact with other J2EE components directly; the interaction takes place using services provided by the containers.

Figure 5-1 on page 103 shows the components running in their containers.

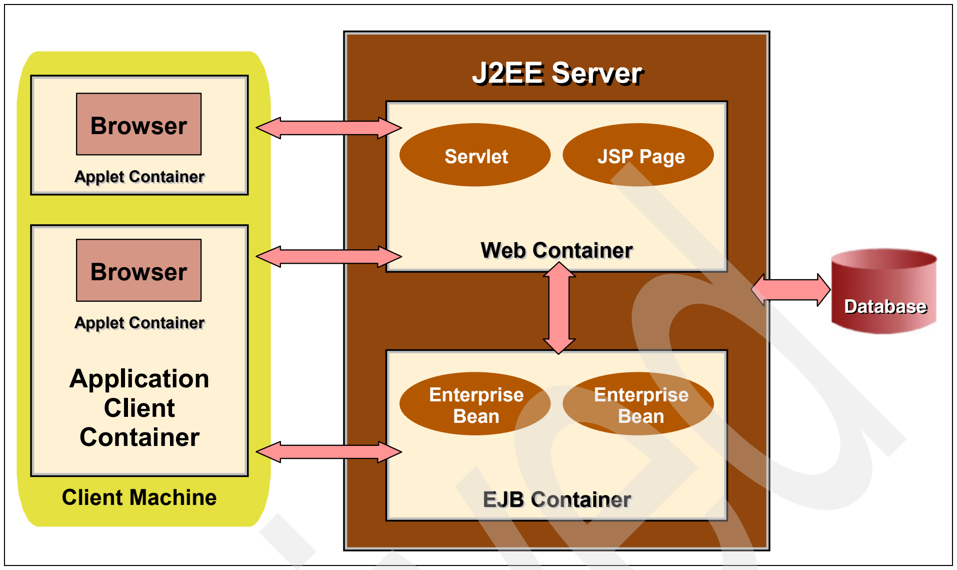


Figure 5-1 J2EE components running in containers

J2EE applications benefit from the services and the scalability provided by the J2EE servers. Because standard protocols are used, the interoperability with other systems is ensured.

The WebSphere Studio Application Developer is an integrated development environment for developing J2EE components as well as assembling and testing J2EE applications. The WebSphere Application Server is a J2EE-compliant server. It comes with the Application Assembly Tool, which can also be used to assemble J2EE applications from existing components.

In the following section, we provide a short introduction to servlets and JavaServer Pages (JSP) because the eClient is based on these technologies. You need to change the JSPs and provide new servlets for the eClient customization and integration.

If you are new to J2EE and need more information, the following steps should help you to quickly learn everything about J2EE:

1. Download IBM WebSphere Studio Application Developer and IBM WebSphere Application Server from:

   `http://www.software.ibm.com/wsdd`

2. To learn the J2EE concepts and get sample codes, go to:

   `http://java.sun.com/j2ee/tutorial`

Implement, assemble and run the sample applications with the downloaded products.

3. To learn about J2EE in a class, enroll in the course “Servlet and JSP Development for WebSphere using WebSphere Studio Application Developer V5.0” (course code WF311). The course catalog can be found at:

   http://www.ibm.com/services/learning/us/catalog/websphere/all.html

5.1.1 What is a servlet?

A servlet is a Java programming language class used to extend the capabilities of servers. Commonly, servlets are used to extend the applications hosted by Web servers (HTTP servers). They are accessed via a request-response programming model. Figure 5-2 shows how this works with the WebSphere Application Server.

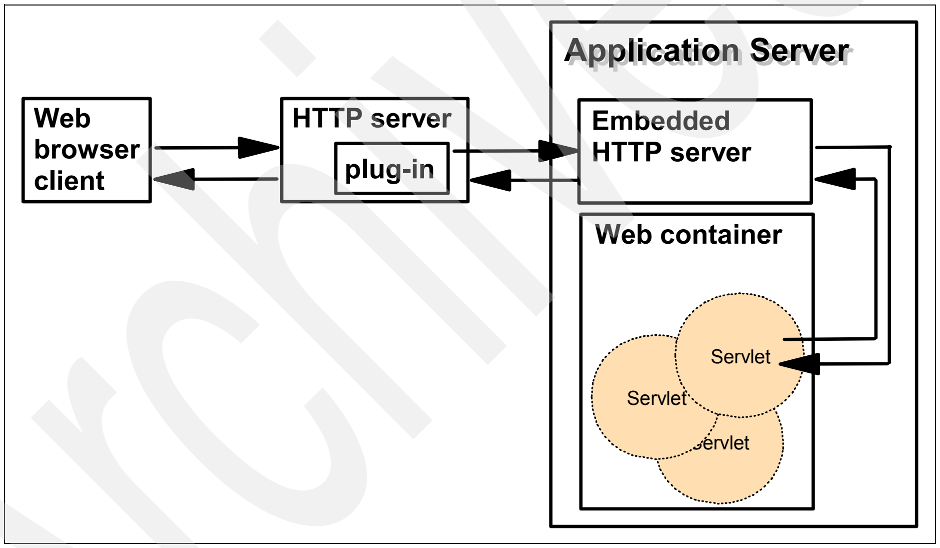


Figure 5-2 Servlet requests and responses in WebSphere Application Server

The client sends a request to the HTTP server that has a plug-in installed that uses a configuration file to determine whether the request should be handled by the HTTP server or the application server. Using the standard HTTP or the secure HTTPS, the request is forwarded to the application server, which calls a servlet to process the request. The response is returned to the client.

The life cycle of a servlet is controlled by the Web container in which the servlet has been deployed. When a request is mapped to a servlet, the container performs the following steps:

1. If an instance of the servlet does not exist, the Web container:
   a. Loads the servlet class.
   b. Creates an instance of the servlet class.
   c. Initializes the servlet instance by calling the init method.
2. Invokes the service method, for example doGet or doPost depending on the HTTP request, and passes a request and response object.

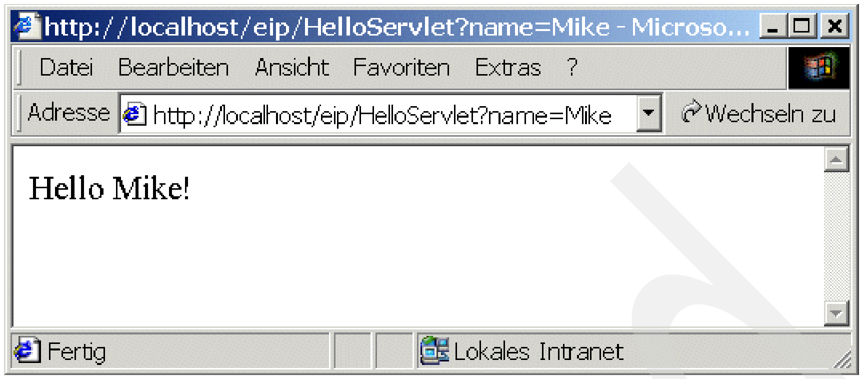
If the container needs to remove the servlet, it finalizes the servlet by calling the destroy method of the servlet.

To implement a servlet, you just need to extend the class HttpServlet, overwrite a service method and send a response to the client. Parameters specified in the request can be retrieved from the request object. This is shown in Example 5-1.

Example 5-1 Implementation of servlet HelloServlet.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class HelloServlet extends HttpServlet {

    public void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {
        String name = req.getParameter("name");
        Writer clientWriter = resp.getWriter();
        clientWriter.write("<HTML><BODY>");
        clientWriter.write("Hello " + name + "!");
        clientWriter.write("</BODY></HTML>");
    }
}
```

After a J2EE application has been created that contains the servlet and the application has been deployed in the application server, you can use a browser to send a request to the servlet. Figure 5-3 on page 106 shows the URL that results in a request to the servlet and the response page returned by the servlet.


http://localhost/eip/HelloServlet?name=Mike - Microso...
Datei Bearbeiten Ansicht Favoriten Extras ?
Adresse http://localhost/eip/HelloServlet?name=Mike
Wechseln zu
Fertig
Lokales Intranet

Hello Mike!

Figure 5-3 Output of the sample servlet

In a servlet, you can also call other servlets and share information with these servlets by storing the information in either the servlet context, session or request object. Each of the three objects has a different scope in the application:

- The servlet context object relates to a group of servlets or JavaServer Pages (a Web application) and is therefore visible to all servlets and JavaServer Pages of this group.
- The session object represents a short-term relationship established between the client browser and the application server. For each client, a new session is created.
- The request object corresponds to the current client request and is visible until the response has been sent to the client.

5.1.2 What is a JavaServer Page (JSP)?

The servlet in Example 5-1 on page 105 plays two different roles. In its first role, it is responsible for the request processing. This is done by collecting information and sending a response to the client. In the second role, it is responsible for the result page layout. This is done with certain HTML tags that are sent to the client.

The problem with this approach is that although the page layout has nothing to do with the request processing, both are tightly coupled in the servlet. If you just want to change the HTML page, you need to change the code that also deals with request processing. The HTML page cannot be changed after the servlet has been compiled and you cannot separate the development of the HTML page from the development of the request processing.

JavaServer Pages (JSP) is a technology that lets you mix regular, static HTML with dynamically generated HTML on the server where the dynamic content can be generated by Java code. JSPs can be called from servlets and have access to the servlet context, the session and the request object, where the data to be displayed has been stored by the servlet. It enables the separation of the HTML code from the business logic.

This design approach is also known as Model-View-Controller (MVC). A controller (the servlet) determines the user request, parses input parameters and invokes the model (the business logic, for example JavaBeans). The view (a JSP) is a “window” into the model and usually presents the result data. This is illustrated in Figure 5-4.

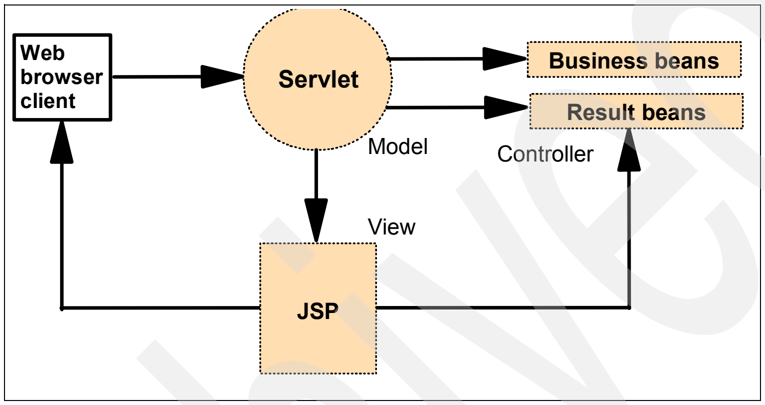


Figure 5-4 Model-View-Controller with J2EE

JSP files have an extension of .jsp and can contain any combination of JSP tags and HTML tags. Along with servlets, JSP files belong to the Web component of a J2EE application. Example 5-2 shows a JSP that displays a message retrieved from the request object with the JSP useBean tag.

Example 5-2 Implementation of JavaServer Page HelloResponse.jsp

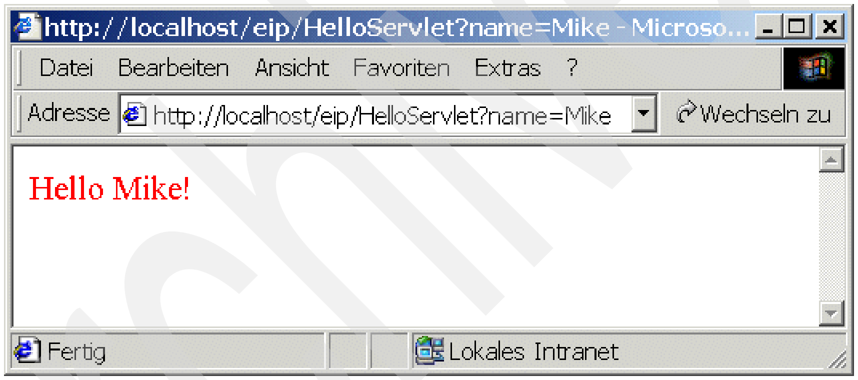
```
<HTML>
<BODY>
<jsp:useBean id="message" class="java.lang.String" scope="request">
</jsp:useBean>
<font color="#ff0000"> <%=message%> </font>
</BODY>
</HTML>
```

The servlet that calls this JSP has to store the message to be displayed in the request object as shown in Example 5-3.

Example 5-3 Implementation of servlet HelloResponse.java with a call to JSP

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class HelloServlet extends HttpServlet {

    public void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {
        String name = req.getParameter("name");
        req.setAttribute("message", "Hello " + name + "!");
    req.getRequestDispatcher("/HelloResponse.jsp").forward(req, resp);
    }
}
```

After a J2EE application has been created that contains the servlet and JSP file and the application has been deployed in the application server, you can again use a browser to send a request to the servlet. Figure 5-5 shows the URL that results in a request to the servlet and the response page that is now created by the JSP.


http://localhost/eip/HelloServlet?name=Mike - Microso...
Datei Bearbeiten Ansicht Favoriten Extras ?
Adresse http://localhost/eip/HelloServlet?name=Mike
Wechseln zu
Fertig
Lokales Intranet

Hello Mike!

Figure 5-5 Output of JSP HelloResponse.jsp

Because the page layout is separated from the servlet, you can now change the layout and the servlet independently. For example, the servlet does not need to be changed if you just want to change the text color. You can also have multiple JSPs, for example, to serve different output devices.

Of course, the page generated by a JSP can again contain links or forms that invoke servlets.

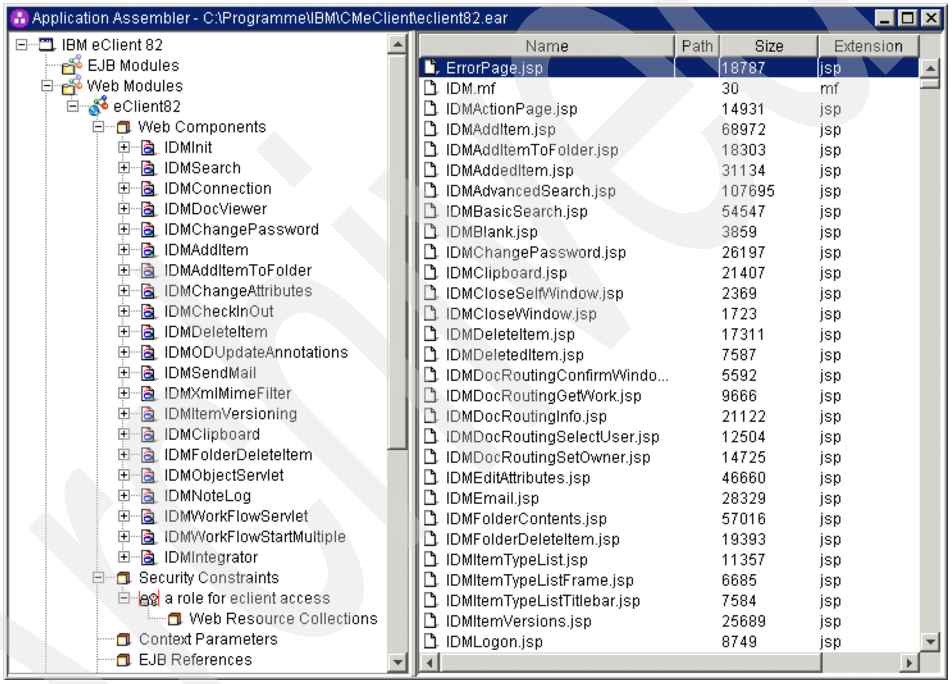
To avoid Java code and reuse functions in JSPs, you can define custom JSP tags for the functions you need. Custom JSP tags are grouped in a tag library, which is an XML file that defines each tag with its parameters and the name of the Java class that implements the tag behavior. The appropriate Java class has to implement one of the interfaces in package java.servlet.jsp.tagext.

5.2 Understanding eClient architecture

In the following section, we provide an overview of the eClient application structure and describe a typical control flow for an eClient function.

5.2.1 Overview

The eClient is a J2EE application. It consists of one Web application, eClient82, which contains several servlets and JSPs. Figure 5-6 shows the structure of the application as it appear in the Application Assembly Tool that comes with WebSphere Application Server.


Application Assembler - C:\Programme\IBM\CMeClient\eclient82.ear

```
IBM eClient 82
    EJB Modules
    Web Modules
        eClient82
            Web Components
                IDMInit
                IDMSearch
                IDMConnection
                IDMDocViewer
                IDMChangePassword
                IDMAddItem
                IDMAddItemToFolder
                IDMChangeAttributes
                IDMCheckInOut
                IDMDeleteItem
                IDMODUpdateAnnotations
                IDMSendMail
                IDMXmlMimeFilter
                IDMItemVersioning
                IDMClipboard
                IDMFolderDeleteItem
                IDMObjectServlet
                IDMNoteLog
                IDMWorkFlowServlet
                IDMWorkFlowStartMultiple
                IDMIntegrator
            Security Constraints
                a role for eclient access
                    Web Resource Collections
            Context Parameters
            EJB References
```

Name	Path	Size	Extension
ErrorPage.jsp		18787	jsp
IDM.mf		30	mf
IDMActionPage.jsp		14931	jsp
IDMAddItem.jsp		68972	jsp
IDMAddItemToFolder.jsp		18303	jsp
IDMAddedItem.jsp		31134	jsp
IDMAdvancedSearch.jsp		107695	jsp
IDMBasicSearch.jsp		54547	jsp
IDMBlank.jsp		3859	jsp
IDMChangePassword.jsp		26197	jsp
IDMClipboard.jsp		21407	jsp
IDMCloseSelfWindow.jsp		2369	jsp
IDMCloseWindow.jsp		1723	jsp
IDMDeleteItem.jsp		17311	jsp
IDMDeletedItem.jsp		7587	jsp
IDMDocRoutingConfirmWindo...		5592	jsp
IDMDocRoutingGetWork.jsp		9666	jsp
IDMDocRoutingInfo.jsp		21122	jsp
IDMDocRoutingSelectUser.jsp		12504	jsp
IDMDocRoutingSetOwner.jsp		14725	jsp
IDMEditAttributes.jsp		46660	jsp
IDMEmail.jsp		28329	jsp
IDMFolderContents.jsp		57016	jsp
IDMFolderDeleteItem.jsp		19393	jsp
IDMItemTypeList.jsp		11357	jsp
IDMItemTypeListFrame.jsp		6685	jsp
IDMItemTypeListTitlebar.jsp		7584	jsp
IDMItemVersions.jsp		25689	jsp
IDMLogon.jsp		8749	jsp

Figure 5-6 eClient structure in the Application Assembly Tool

The servlets are listed in the Web component category in the left window in Figure 5-6. The right window lists the resource files of the Web application, which are the JSP files. A complete list of the JSPs and their purpose is provided in 5.3.2, "eClient JSPs" on page 114.

The Web application also defines two custom JSP tag libraries (cmb.tld, cme.tld) that are used in the JSPs. They provide JSP tags for iteration and flow control.

In addition to servlets and JSPs, the Web application also defines a security constraint that is mapped to the user role AccessEclient defined in the J2EE application. If the application is deployed in WebSphere Application Server and security is enabled, the user role can be mapped to real user IDs to restrict the access to the client.

The eClient uses configuration properties that you can edit in the IDM.properties file. The IDM.properties file resides in the root directory where the eClient is installed. Most aspects of managing the eClient application are controlled by the parameters in this file. The parameters are described in the Information Center. Select **eClient -> Managing your eClient application -> Setting and changing your configuration parameters**.

5.2.2 Inspecting eClient control flow

Understanding the control flow of the eClient is the key to eClient customization and integration.

In the following section, we describe the control flow of the search function. We illustrate the eClient architecture to help you understand how the Model-View-Controller design is implemented. With the provided code snippets, this is also a reusable example of how to determine the control flow of an arbitrary eClient function.

Figure 5-7 on page 111 gives an overview of the search control flow and is followed by a detailed description of each step.

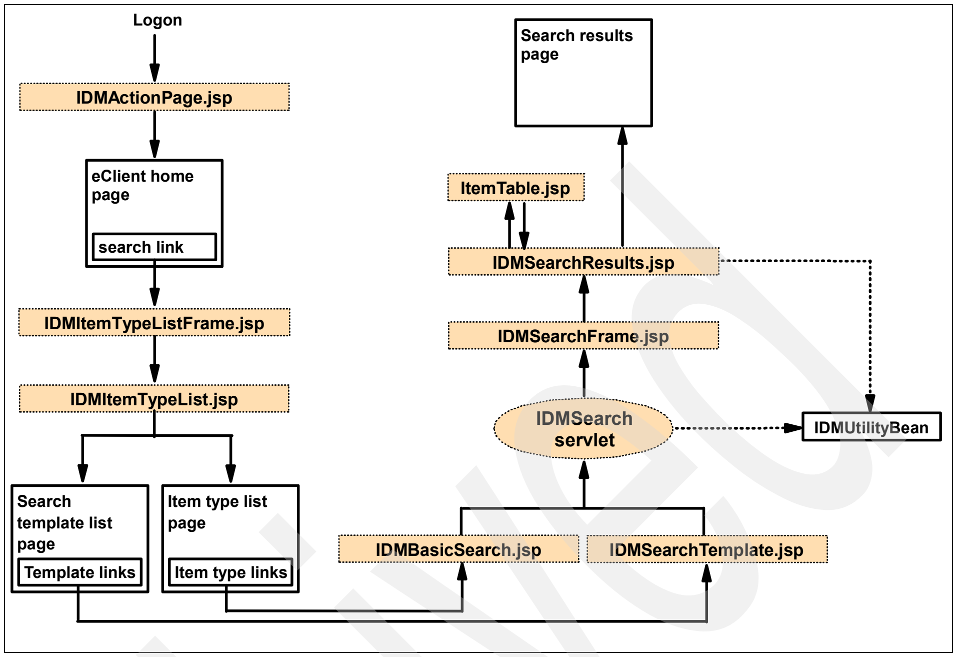


Figure 5-7 eClient search control flow

After logon, the eClient home page is generated by the IDMActionPage.jsp. If you click the search link, the IDMItemTypeListFrame.jsp is called:

```
<a href="<%= webAppName %>/IDMItemTypeListFrame.jsp" ...
```

The IDMItemTypeListFrame.jsp generates a frame set where the content for the frame that displays the item types list is created by a call to the IDMItemTypeList.jsp:

```
<frame name="ItemTypeList" title="<%= title %>" noresize src="<%=
webAppName %>/IDMItemTypeList.jsp?<%= key %>">
```

The item type list is built according to the connector that is used to log on. For the federated connector, a list of search templates needs to be created. If the Content Manager connector is used, you expect a list of item types. Before creating the list, the connector type is checked using a custom JSP tag:

```
<cme:if condition="<%=
connection.getDsType().equalsIgnoreCase(cvb.CMB_DSTYPE_FED) %>">
```

If the federated connector is used (cvb.CMB_DSTYPE_FED), a list of search templates is generated. Each search template name is a link to the IDMSearchTemplate.jsp where the template name is specified as a parameter:

```
<a href="<%= webAppName %>/IDMSearchTemplate.jsp?<%= key %>&entityName=<%=
java.net.URLEncoder.encode(searchtemplate) %>" ...
```

If the Content Manager connector is used, item types are listed. There is a link to IDMBasicSearch.jsp with the item type name as a parameter:

```
<a href="<%= webAppName %>/IDMBasicSearch.jsp?<%= key %>&entityName=<%=
com.ibm.idm.util.URLUTF8Encoder.encode(name) %>" ...
```

The IDMSearchTemplate.jsp is called when you click a search template; or if you have a list of item types, the IDMBasicSearch.jsp is called if you click one. Both JSPs create a HTML form for the search values. According to the specified form action, the IDMSearch servlet is called in both cases when you submit the form:

```
<FORM NAME="searchCriteria" action="<%= webAppName %>/IDMSearch"
Method="Get">
```

The servlet performs the search, and according to the IDM.properties file, it calls the IDMSearchFrame.jsp to display the results:

```
Output.IDMSearch=/IDMSearchFrame.jsp
```

The IDMSearchFrame.jsp generates a frame set where the content for the frame that displays the search results is created by a call to the IDMSearchResults.jsp:

```
<frame noresize title="ResultsBottom" name="ResultsBottom" src="<%=
webAppName %>/IDMSearchResults.jsp ...
```

As with most JSPs, the IDMSearchResults.jsp retrieves the IDMUtilityBean object from the session because the servlet has stored the search results in this object:

```
<jsp:useBean id="cub" scope="session"
class="com.ibm.idm.beans.IDMUtilityBean">
```

The IDMSearchResults.jsp then retrieves the results from this object:

```
CMBSearchResults results = cub.getSearchResults(srKey);
```

To create the result table, the IDMSearchResults.jsp calls another JSP that generates just the HTML fragment for the table, and it includes this table in the page. To share the search results object with the other JSP, it is first stored in the request object:

```
request.setAttribute("results",results);
```

The ItemTable.jsp is then called to create the table, which is included in the current page:

```
dispatcher =
application.getRequestDispatcher("/pageComponents/ItemTable.jsp");
dispatcher.include(request, response);
```

In the ItemTable.jsp, the search results object is first retrieved from the request:

```
CMBSearchResults results = (CMBSearchResults)
request.getAttribute("results");
```

The items are then retrieved from the search results object:

```
items = results.getItems();
```

Finally, the attribute values of each item are set in the appropriate column of a table row:

```
<TD class='" + rowType + "' align='left' " + colWidth + " nowrap>" +
item.getAttrValue(colName) + "</TD>"
```

The table is returned to the client as part of the result page.

5.3 Customization and integration

In the following section, we give an overview about the possibilities for adapting the eClient according to your business needs.

5.3.1 Overview

You can customize the eClient by:

- **Changing the JSPs**. The JSPs are located in the directory where you installed the eClient. To customize the look and feel of the eClient, you can modify these JSPs or substitute JSPs of your own. The complete list of JSPs can be found in 5.3.2, “eClient JSPs” on page 114. An example can be found in Chapter 10, “Customizing the edit attributes window” on page 225.
- **Customizing eClient graphics**. You can replace the artwork with your own to customize the graphics for your eClient. You can set the font, colors, and background colors of the eClient in the eclient81.css Cascading Style Sheet file. An introduction is provided in 5.3.3, “Customizing eClient graphics” on page 120.
- **Customizing eClient help**. As part of customizing the eClient for your users, you can provide customized online help or add your own. The help files are written in Hypertext Markup Language (HTML) and reside in the directory where the eClient is installed. You can change the style sheet as well as the

graphic for the background, bkgrd.gif, located in the icons directory. If you add your own help topics, you must modify the JSP for the pages or panels from which you want to display the new help topics.

- **Customizing the viewer applet**. The applet is used to display the content of a document. Because it is based on the generic document viewer of the viewer toolkit (see 6.1, "Programming interface overview" on page 124) you can customize the viewer applet by modifying the default configuration file, CMBViewerConfiguration.properties, located in the cmbview81.jar. It is documented in the Information Center; select **Enterprise Information Portal -> Programming -> Working with the Java document viewer toolkit -> Customizing the generic document viewer**.
- **Adding new servlets**. You can integrate new functionality into the eClient by adding new servlets to the application. A servlet can either be added to the control flow by changing the IDM.properties file (see 13.2, "Creating categories and summaries during document import" on page 296 for an example) or with a link that points to the servlet (see. 13.3, "Searching for related items" on page 306).

You can integrate the eClient with third-party software. To read about Siebel integration, refer to Chapter 15, "Siebel Integration" on page 347.

5.3.2 eClient JSPs

In this section, we provide a complete list of the available JSPs with a short description. You can modify these JSPs or substitute JSPs of your own to customize eClient.

eClient JSPs are divided into the following categories:

- Application control flow as in Table 5-2 on page 116
- Items as in Table 5-2 on page 116
- Search as in Table 5-3 on page 117
- Folders as in Table 5-4 on page 118
- Annotations as in Table 5-5 on page 118
- Workflow as in Table 5-6 on page 118
- Document routing as in Table 5-7 on page 119

Table 5-1 eClient JSPs that are related to the application control flow

JSP file	Purpose
ErrorPage.jsp	Displays when an error is encountered.
IDMActionPage.jsp	Opens the eClient home page, where your user can start to use the eClient functions.

JSP file	Purpose
IDMBlank.jsp	Displays a blank page.
IDMChangePassword.jsp	Displayed when the user wants to change the password.
IDMCloseSelfWindow.jsp	JSP that will close itself, used in IDM.properties as the out page.
IDMCloseWindow.jsp	Closes a window.
IDMLogon.jsp	Displays when the user first accesses the eClient and for logging on to the server. This page displays the banner graphic (banner.gif). To customize the banner, you can supply a different graphic and call it from this page.
IDMLogon2.jsp	Displays the Logon page.
IDMLogonNewPassword.jsp	Displays the Change Password page, where your users can change their passwords.
IDMMessageBox.jsp	Displays a message box.
IDMNoteLog.jsp	Displays the window where a user can view or add to the Notelog from a Content Manager Version 8 server.
IDMPrintControl.jsp	Allows the user to print HTML-based content from their browser.
IDMPrintFrameset.jsp	Displays the print options.
IDMQueryBuilder.jsp	Displays the query builder.
IDMUserIDMapping.jsp	Provides user with a logon window to modify stored user ID and password mapped into the EIP administrative database for a federated server.
IDMProcessing.jsp	Displays the “Processing” or “Please Wait”graphic when there is an ongoing process.
IDMProgressIndicator.jsp	Displays the progress indicator.
SessionErrorPage.jsp	Pops up an alert telling the user that the session has expired.

Table 5-2 eClient JSPs that are related to items

JSP file	Purpose
IDMAddedItem.jsp	Confirms that an item has been added to a folder or worklist.
IDMAddItemToFolder.jsp	Allows a document or folder to be added to a folder.
IDMDeleteItem.jsp	Allows an item to be deleted from the database.
IDMDeletedItem.jsp	Verifies that an item was deleted.
IDMClipboard.jsp	Allows items to be viewed in the clipboard.
IDMEditAttributes.jsp	Displays the item attributes and allows for updating the attributes. Used to change how an item is indexed.
IDMEmail.jsp	Displayed when the user wants to create an e-mail message with an object attached.
IDMItemTypeList.jsp	Displays the list of item types or search templates. This JSP is a part of the IDMItemTypeListFrame.jsp frameset.
IDMItemTypeListFrame.jsp	Displays the frameset that contains IDMItemTypeList.jsp and IDMItemTypeListTitlebar.jsp.
IDMItemTypeListTitlebar.jsp	The title bar that displays the information in the frameset for displaying the list of item types or search templates.
IDMItemVersions.jsp	Displays a list of all of the versions of an item.
ItemTable.jsp	Displays a collection of items.
ItemTableHeader.jsp	Displays table headers for a collection of items.
ItemTabs.jsp	Displays item type tabs for a collection of items.
mail.jsp	Enables e-mailing of a document.
SRItemTableHeader.jsp	Enables ICM display order.

Table 5-3 eClient JSP that are related to search

JSP file	Purpose
IDMAdvancedSearch.jsp	Displays the advanced search page in a frame controlled by IDMSearchFrame.jsp.
IDMBasicSearch.jsp	Displays the basic search page in a frame controlled by IDMSearchFrame.
IDMSearchFrame.jsp	Displays the main search page frameset.
IDMSearching.jsp	Displays the processing indicator on a search results page.
IDMSearchResults.jsp	Displays the search results.
IDMSearchTemplate.jsp	Displays the page that contains the list of valid search templates or item types that a user can use for searching.
IDMSearchToolbar.jsp	Displays the toolbar for the search.
IDMViewApplet.jsp	Opens the HTML page embedded with the viewer applet.
IDMViewFrames.jsp	Displays the View page; when the entire item is sent to the browser, this page writes the item to the browser.
IDMViewPage.jsp	Displays the pane containing the current page of the selected item in the View page; this page is displayed in the lower frame of IDMViewFrame using the current settings for size, rotations, and other parameters.
IDMViewToolbar.jsp	When an item type is viewed, displays the toolbar in the upper frame of IDMViewFrame.
IDMResultsFrameBottom.jsp	Displays the bottom frameset of the search results.
Heading.jsp	Displays heading of the search result page.

Table 5-4 eClient JSPs that are related to folders

JSP file	Purpose
IDMAddItem.jsp	Displays the Import Document, Create Folder, and Create Federated Folder pages.
IDMFolderContents.jsp	Displays the contents in a folder. This is within a worklist or search results. This is displayed within the same frame as the worklist of search results.
IDMFolderDeleteItem.jsp	Displayed when the user wants to remove an item from a folder.

Table 5-5 eClient JSPs that are related to annotations

JSP file	Purpose
IDMODAnnotationsBB.jsp	Displays the bottom area of the OnDemand annotations interface.
IDMODAnnotationsBS.jsp	Displays the frame for the search area of the OnDemand annotations interface.
IDMODAnnotationsBT.jsp	Displays the top area of the OnDemand annotations interface.
IDMODAnnotationsEntry.jsp	Displays a framed interface made up of several frames.
IDMODAnnotationsFrame.jsp	Displays the View Annotations page; this file contains the frameset for the page.
IDMODAnnotationsList.jsp	Displays the list of annotations for the selected document in a frame of IDMODAnnotationsFrame.
IDMODAnnotationsView.jsp	Displays the annotation content for the selected annotation within the OnDemand annotations interface.

Table 5-6 eClient JSPs that are related to workflow

JSP file	Purpose
IDMWorkLists.jsp	Lists worklists that the user can retrieve.
IDMWorkItems.jsp	Displays work items in a worklist.
IDMWorkflowCheckIn.jsp	Checks in work items.

JSP file	Purpose
IDMWorkflowFrames.jsp	Contains IDMWorkFlowToolbar.jsp and IDMWorkItems.jsp.
IDMWorkflowNotifications.jsp	Displays work notifications.
IDMWorkflowStartOnMultiple.jsp	Starts workflow for multiple items.
IDMWorkflowToolbar.jsp	Toolbar for IDMWorkItems.jsp.
IDMWorkflowUserVariables.jsp	Displays the process information.
IDMWorkflowChange.jsp	Moves selected item from current workflow to another workflow.
IDMWorkflowInfo.jsp	Displays the workflow or document routing information.
DMWorkflowPriority.jsp	Allows user to change the priority of a workflow item.
IDMWorkflowStart.jsp	Starts an item on an EIP workflow or Content Manager document routing process.
IDMWorkflowStrings.jsp	Displays workflow variables for the work item. Each item on an EIP workflow has up to five variables that can be displayed or edited.
IDMWorkflowSuspend.jsp	Allows the user to suspend the workflow on a chosen document for a specified time.
IDMWorkflowDelNotif.jsp	Deletes a workflow notification list.
IDMWorking.jsp	Displays a processing indicator for workflow.
WFPageHeading.jsp	Displays the column headers for the worklist.

Table 5-7 eClient JSPs that are related to document routing

JSP file	Purpose
IDMDocRoutingConfirmWindow.jsp	Opens a window and displays a page that confirms an item has been put on a document routing process.

JSP file	Purpose
IDMDocRoutingGetWork.jsp	Internal JSP that retrieves items from a document routing process or workflow and populates a worklist.
IDMDocRoutingInfo.jsp	Displays the document routing information for an individual item from the search results.
IDMDocRoutingSelectUser.jsp	Displays a window to select a user to assign the work to from a worklist.
IDMDocRoutingSetOwner.jsp	Set the owner for a document routing process.

5.3.3 Customizing eClient graphics

All of the graphics (including the icons) that are used by the JSPs and the help are located in the CMeClient\installedApp\IBM_eClient_82.ear\eclient82.war\icons directory.

The most common graphics to change are the background graphics. Four different background graphics are used in the eClient and each is specified with an individual CSS class. Replacing these background graphics with files of the same name will change the background of those pages that use the related CSS class. Consult Table 5-8 to determine which files and class to modify. All of the backgrounds can be specified to use the same graphic, or not use any graphics, but four different classes and backgrounds are used by default to provide flexibility within the interface.

Table 5-8 CSS classes and background graphics

CSS class or element	Background graphic	Description
BODYLOGON	icons/logon_bk.jpg	Used for the Logon and Change Password pages within the eClient.
BODYHOME	icons/home_bk.jpg	Used for the Home page within the eClient. This is the first page that is displayed after the Logon page.
BODYMINI	icons/mini_bk.jpg	Used for framed pages within the eClient. The Basic Search, Advanced Search, and Search Template viewer pages use this class.

CSS class or element	Background graphic	Description
BODYDIALOG	icons/dialog_bk.jpg	Used for windows within the eClient. The import, edit attributes, e-mail and many other windows use this class.
BODY	N/A	Used for all other pages in the eClient. The BODY element does not by default specify a background graphic, but specifies the background color as white.

You can replace the art work with your own to customize the graphics for your eClient. You can set the font, the colors, and the background colors of the eClient in the eclient81.css Cascading Style Sheet file. If the default location for the icons or graphics are changed, the CSS style that specifies a background image may also need to be modified.

The usage of style sheets is discussed in Chapter 9, “Customizing look and feel using style sheets” on page 213.

6

Creating applications with EIP

This chapter provides the essential information required to start creating applications with Information Integrator for Content (formerly known as EIP and is still referred to as EIP in this redbook). We cover the programming interface and go through an API test drive to help you get up and running as quickly as possible.

This chapter includes the following:

- ► Using Content Manager connector
- ► Using DB2 connector
- ► Using federated connector to access DB2
- ► Using federated connector to search across content servers
- ► Working with Information Mining Service
- ► Working with controller servlet
- ► Working with viewer toolkit

This is far from being a programming guide. For a complete programming guide, refer to the Information Center. Select **Enterprise Information Portal -> Administration -> Managing information mining**.

6.1 Programming interface overview

The application programming interfaces (APIs) are a set of classes that access and manipulate either local or remote data according to the architecture described in 1.3, “Information Integrator for Content Version 8” on page 6.

The APIs support:

- A common object model for data access
- Multiple searches and updates across a heterogeneous combination of content servers
- A flexible mechanism for using a combination of search engines such as the Content Manager text search feature
- Workflow capability
- Text analysis capability (Information Mining Service)
- Document viewing capability
- Administration functions
- Client/server implementation for Java applications

You can choose from the following interfaces:

- C++ classes
- Java classes
- JavaBeans
- Controller servlet and JSP tag library

Because we focus on the APIs that can directly be used in J2EE applications such as eClient, the C++ API is not discussed in detail; however, the C++ classes are aligned with the Java classes and can therefore be used in a similar way.

The Java APIs can reside on both the EIP server and the client (both provide the same interface). The client API communicates with the server to access data through the network via Java RMI (Remote Method Invocation). Communication between the client and the server is performed by classes; it is not necessary to add additional programs.

Java classes consist of the following packages:

- server
- client
- cs
- common

The client and server classes provide the same APIs, but have different implementations.

The server package is com.ibm.mm.sdk.server. The classes in the server package communicate directly with the federated or back-end content server.

The client package is com.ibm.mm.sdk.client. The classes in the client package communicate with the classes in the server package via RMI.

The common classes are shared by both the client and server.

Sometimes, an application does not know where the content resides. For example, an application can have content residing on the client at one time and on the server at another time. The cs package connects the client and server dynamically depending on .ini file settings.

The client application must import the client package, the dynamic application must import the cs package, and the server application must import the server package.

Although the same API is provided for the client and server, the client package has an additional exception item because it communicates with the server package.

The EIP JavaBeans can be divided into:

- Non-visual Java beans

  You can use the non-visual beans to build Java and Web client applications that require a customized user interface. The non-visual beans support the standard bean programming model by providing default constructors, properties, events and a serializable interface. You can use the non-visual beans in builder tools that support introspection.

- Visual Java beans

  The visual beans are customizable, Swing-based, graphical user interface components. Use the visual beans to build Java applications for Windows. You can place them within windows and dialogs of Java-based applications. Because the visual beans are built using the non-visual beans (as a data model), you must use them in conjunction with the non-visual beans when building an application.

The sample applications discussed in the following sections show how to create applications with the Java APIs and the JavaBeans.

An example of visual and non-visual JavaBeans is the viewer toolkit, which allows you to integrate document viewing capabilities in your application. It consists of a generic document viewer, a document viewer bean and a document services bean. Depending on the application architecture, you choose one of the three components. This is illustrated in Figure 6-1.

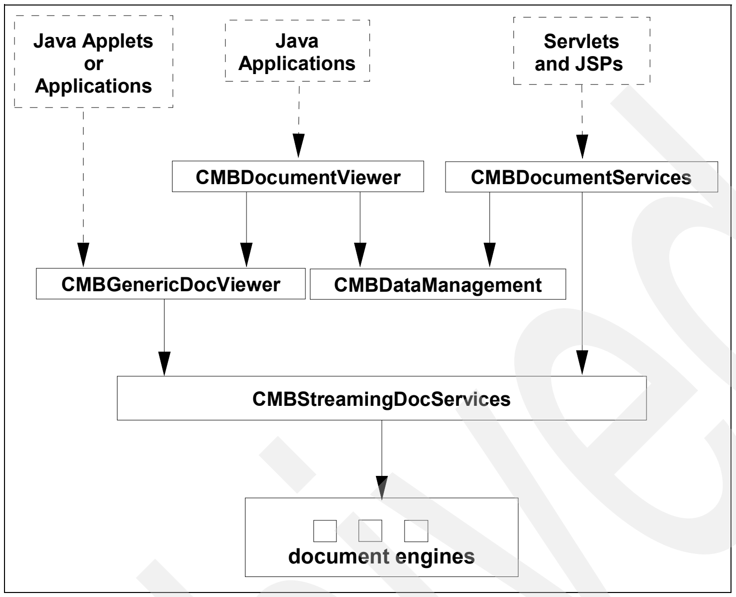


Figure 6-1 Viewer toolkit overview

The *generic document viewer* provides a graphical user interface to view the content of a document on a client. To parse the different document formats and render the content, *streaming document services* are used. The actual processing is done by a set of *document engines*. The generic document viewer allows you to customize the look and feel of many properties. It handles all page and document navigation, page viewing operations such as rotating, zooming and scaling, and annotations editing. The generic document viewer can be used in Java applets or stand-alone client applications.

The *document viewer bean* is based upon the generic document viewer but it can also launch external viewers to view documents. The viewer to be launched for a certain MIME type is specified in the EIP Administration Client. The document viewer bean can only be used in stand-alone client applications.

The *document services bean* performs server-side document conversions, such as in the case of a Web-based application. It can convert documents from content types that are not handled by the Web browser (documents that require a plug-in or native application launch) to content types that are handled by the Web browser natively, such as HTML, GIF, JPEG, or for which plug-ins are readily available, such as PDF. Like the generic document viewer, the document services bean calls the streaming document services to parse and render the documents. The document services bean can be used in servlets and JSPs. You can find a sample implementation in 6.2.8, “Working with viewer toolkit” on page 176.

EIP provides a *servlet* with pluggable actions that can be used when building Web applications. This servlet acts as a controller of a Model-View-Controller design Web application. It performs actions and initializes the beans (the model), which are then accessed in the JSPs (the views) either directly or indirectly by using JSP tags. This is illustrated in Figure 6-2.

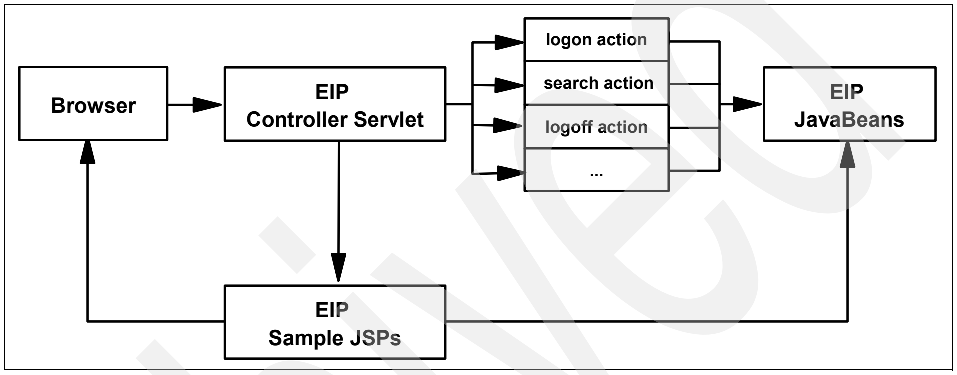


Figure 6-2 Controller servlet overview

Actions performed for typical application tasks include:

- Log on and log off.
- Search.
- Create, retrieve, modify, and delete documents.
- Create folders, and add documents to or remove documents from folders.
- Launch documents and document pages for viewing.

In addition, the servlet performs common tasks before and after the action, such as management of the connection to the content server. After every action, a JSP is invoked to format the results and send them back to the browser.

You can customize the servlet to add new actions and associate JSPs with the actions. This is shown in 6.2.7, “Working with controller servlet” on page 162.

The *JSP tag library* enables an application to dynamically create HTML files in JSPs without the use of Java code. The tag library consists of an XML file (taglib.tld) that defines each tag with its parameters and the name of the Java class that implements the tag behavior. The tag library supports the creation of HTML files with functions such as:

- Iterate through available data sources.
- Iterate through items in search results.

- Iterate through attributes of an item.
- Iterate through available search templates.
- Iterate through search criteria of a search template.
- Iterate through available entities.

6.2 Taking an API test drive

The best way to understand an API is to implement a simple application. The following samples help you to get some simple codes up and running as quickly as possible.

We first describe how you load the sample databases that come with Content Manager and EIP. We then discuss sample applications that work with these databases and demonstrate the usage of:

1. The connectors
2. The Information Mining Service
3. The controller servlet
4. The document viewer toolkit

Because we want to demonstrate the programming interfaces that can directly be used in J2EE applications such as the eClient, all sample applications use the Java programming language and the appropriate APIs.

6.2.1 Setting up sample data

Before we start to build applications that manage content, we first need to load data to the content servers.

The content servers used here are DB2 and Content Manager. In addition, we need Information Integrator for Content (EIP), and, of course, the eClient.

In the following section, we assume that Content Manager and EIP do not share the same database. The database names are:

- Content Manager: ICMNLSDB
- EIP: EIPDB

To load the sample data for Content Manager:

1. Select **Start -> Programs -> IBM Content Manager for Multiplatforms V8.2 -> First Steps**.
2. In the First Steps window, select **Load Sample Data** to start the creation of the sample data.

3. Use the System Administration Client to verify that the sample item types starting with `XYZ_` have ben created in database ICMNLSDB. To list the item types, expand category **ICMNLSDB**, then **Data Modeling**, and click **Item Types** as shown in Figure 6-3.

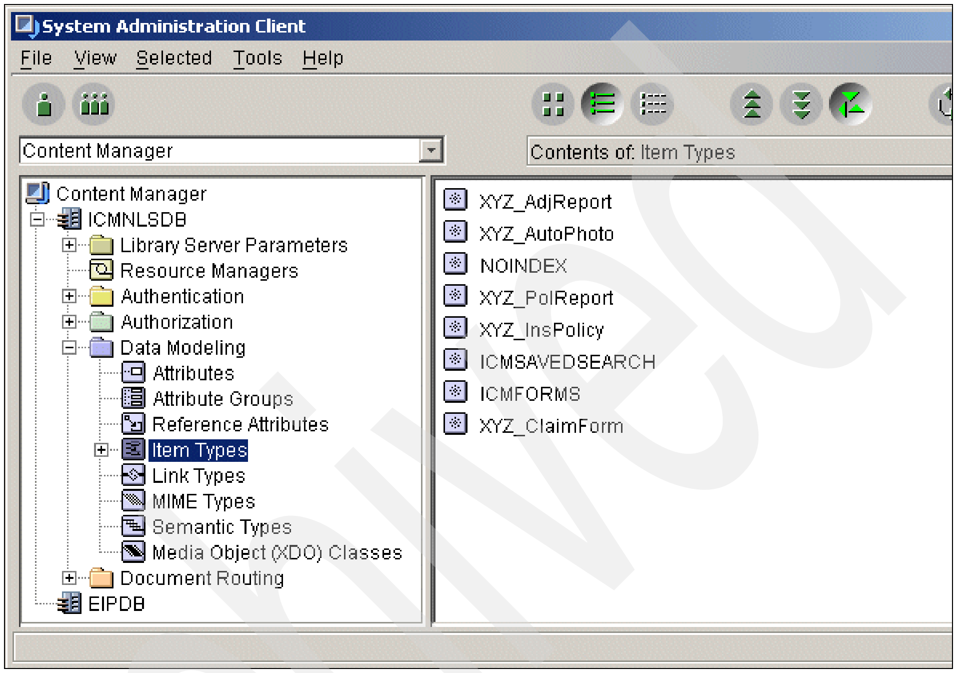


Figure 6-3 Content Manager Version 8 sample item types

To load sample data for DB2, we use a press article database that comes with the sample data for EIP:

1. Select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> EIP First Steps**.
2. In the First Steps window, select **Load Sample Data** to start the creation of the sample data.
3. Use the System Administration Client to verify that the sample database EIPSAMPL contains the search templates SearchXYZClaimForms and SearchLongBySource as shown in Figure 6-4 on page 130.

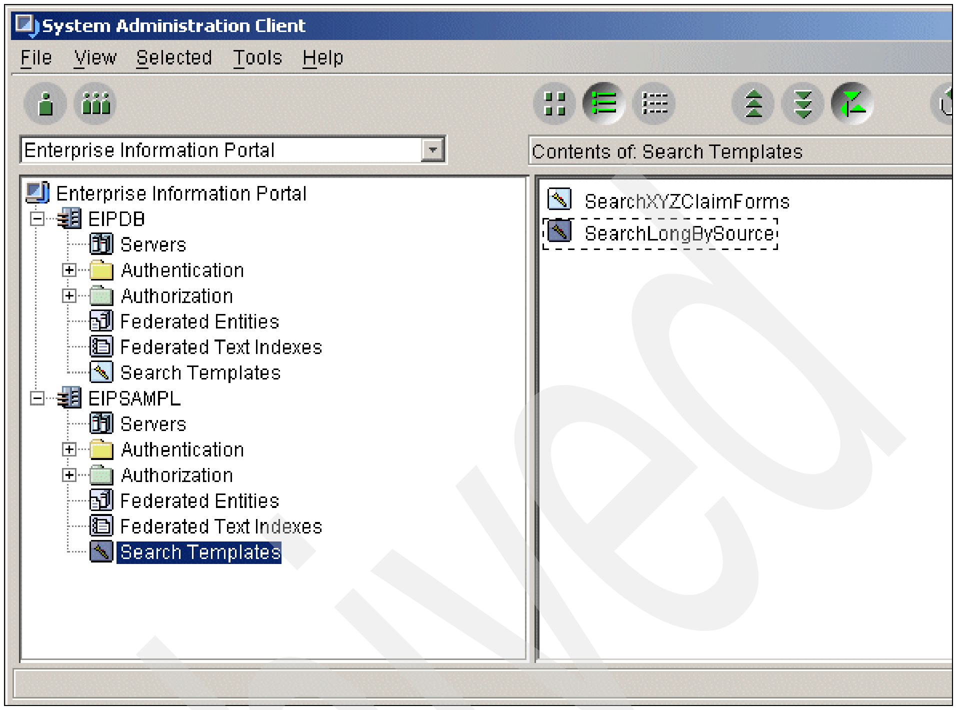


Figure 6-4 Information Integrator for Content Version 8 sample search templates

The structure of the sample databases is shown in Figure 6-5 on page 131.

The Content Manager database, ICMNLSDB, contains various item types. You can use the Content Manager connector to access the appropriate items. See 6.2.2, "Using Content Manager connector" on page 131.

The DB2 database, IBMPRESS, contains one table. You can use the DB2 connector to access the appropriate rows. See 6.2.3, "Using DB2 connector" on page 135.

In the sample administration database, EIPSAMPL, the columns of a DB2 table, also known as native attributes, are mapped to federated attributes. Two of the federated attributes became search criteria in a search template. This means you can use the federated connector to access the DB2 data. See 6.2.4, "Using federated connector to access DB2" on page 137.

Of course, you can also map the Content Manager attributes to federated attributes and use them together with the existing federated attributes in a federated entity to search across content servers. See 6.2.5, "Using federated connector to search across content servers" on page 143.

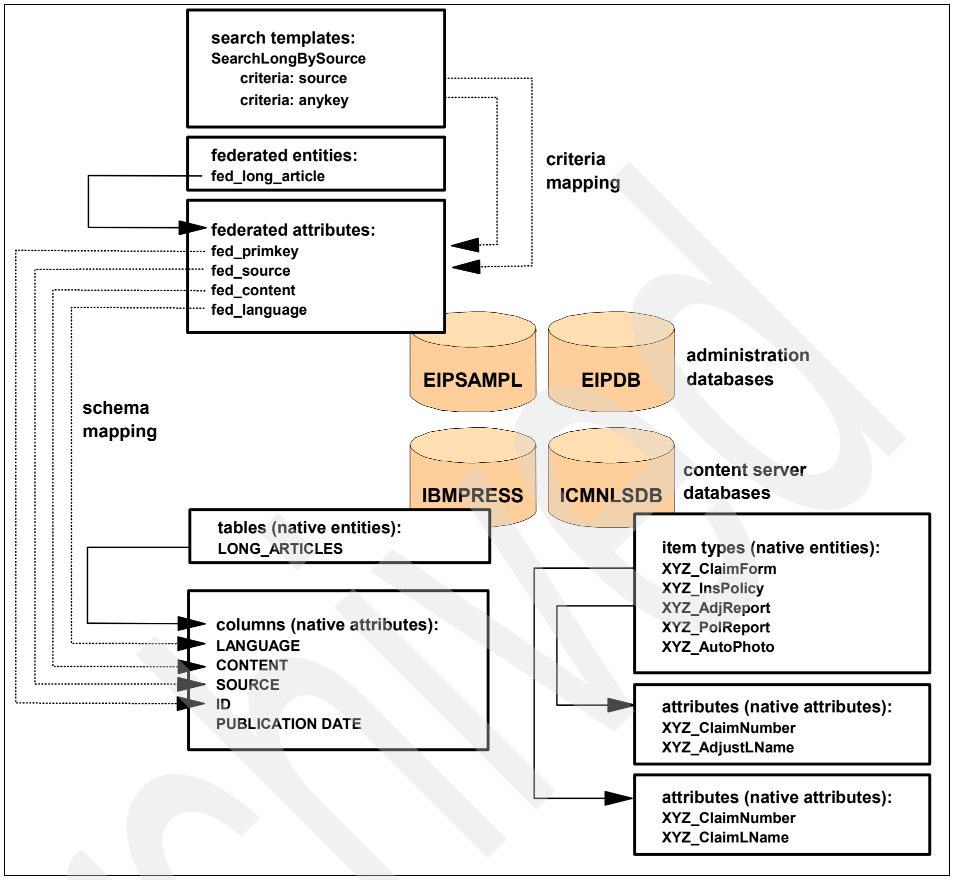


Figure 6-5 Sample databases structure

6.2.2 Using Content Manager connector

In this section, we demonstrate how to use a Content Manager connector through a sample Java application.

Java application

The sample application InformationAccessICM.java shows how to use the Content Manager connector to access the data stored in a Content Manager server.

The main class of the connector is DKDatastoreICM. You first have to call its connect method:

```
DKDatastoreICM dsICM = new DKDatastoreICM();
dsICM.connect(database,userName,password,"");
```

To be able to find objects in the datastore, the connector provides an XML-based query language that conforms to XQuery Path Expressions (XQPE), a subset of the W3C XML Query working draft. For details on query syntax, refer to the Information Center; select **Enterprise Information Portal -> Programming -> Working with Content Manager Version 8.2 -> Understanding the query language**. In this sample, we search for all items that belong either to item type XYZ_ClaimForm or XYZ_AdjReport.

The execute method of class DKDatastoreICM evaluates the query string command by executing it against the datastore and returning all results in a collection. You can also specify query options, for example to limit the number of returned results or the data that is retrieved for each result. The method returns a cursor that can be used to fetch the results:

```
String query = "(/XYZ_ClaimForm | /XYZ_AdjReport)";
DKNVPair options[] = new DKNVPair[3];
options[0] = new DKNVPair(DKConstant.DK_CM_PARM_MAX_RESULTS, "0");
options[1] = new DKNVPair(DKConstant.DK_CM_PARM_RETRIEVE,
new Integer(DKConstant.DK_CM_CONTENT_ATTRONLY));
options[2] = new DKNVPair(DKConstant.DK_CM_PARM_END, null);
dkResultSetCursor cursor = dsICM.execute(query,
DKConstantICM.DK_CM_XQPE_QL_TYPE, options);
```

A call to the fetchNext method of the cursor returns a Dynamic Data Object (DDO) of type DKDDO. The complete result set has been retrieved if the method returns null:

```
DKDDO  ddo;
while((ddo = cursor.fetchNext())!=null) {
...
}
```

Each DDO represents a persistent data object in the Content Manager server. You can use it to retrieve and update the appropriate object data. You can also delete the object in the server.

To run the sample:

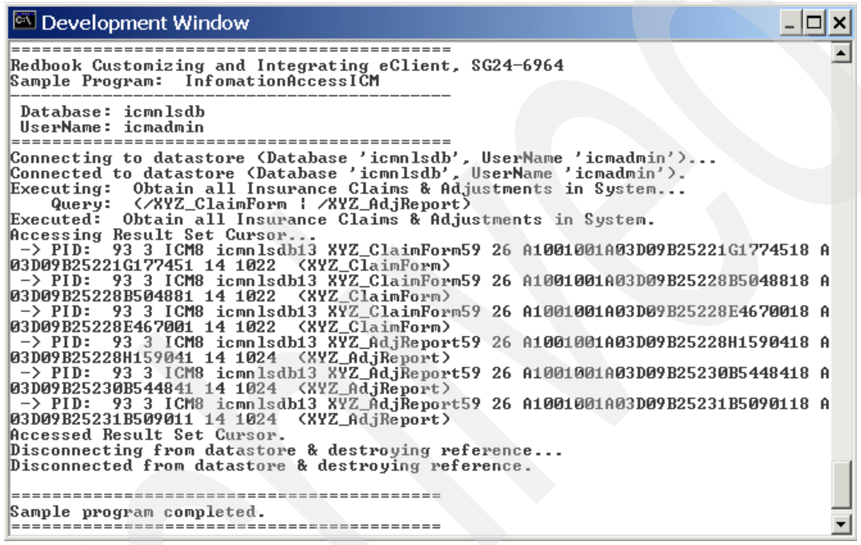
1. Select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window** to open a command window that has the required environment settings.
2. Change to the directory where you stored the redbook samples.
3. To compile the sample, run **`javac InformationAccessICM.java`**.
4. To run the sample, enter **`java InformationAccessICM`**.

Please note that the sample uses default values for the database (icmnlsdb), the user ID (icmadmin) and the password (password). To overwrite the defaults, specify these values on the command line as follows:

```
java InformationAccessICM <icmnlsdb> <icmadmin> <password>
```

Replace `<icmnlsdb>`, `<icmadmin>`, and `<password>` with your own values.

For each resulting data object, its Persistent Identifier (PID) is displayed. The output of the sample is shown in Figure 6-6.

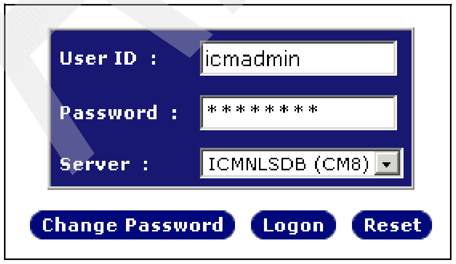


```
==========================================
Redbook Customizing and Integrating eClient, SG24-6964
Sample Program:  InfomationAccessICM
------------------------------------------
 Database: icmnlsdb
 UserName: icmadmin
==========================================
Connecting to datastore (Database 'icmnlsdb', UserName 'icmadmin')...
Connected to datastore (Database 'icmnlsdb', UserName 'icmadmin').
Executing:  Obtain all Insurance Claims & Adjustments in System...
    Query:  (/XYZ_ClaimForm | /XYZ_AdjReport)
Executed:  Obtain all Insurance Claims & Adjustments in System.
Accessing Result Set Cursor...
 -> PID:  93 3 ICM8 icmnlsdb13 XYZ_ClaimForm59 26 A1001001A03D09B25221G1774518 A
03D09B25221G177451 14 1022  (XYZ_ClaimForm)
 -> PID:  93 3 ICM8 icmnlsdb13 XYZ_ClaimForm59 26 A1001001A03D09B25228B5048818 A
03D09B25228B504881 14 1022  (XYZ_ClaimForm)
 -> PID:  93 3 ICM8 icmnlsdb13 XYZ_ClaimForm59 26 A1001001A03D09B25228E4670018 A
03D09B25228E467001 14 1022  (XYZ_ClaimForm)
 -> PID:  93 3 ICM8 icmnlsdb13 XYZ_AdjReport59 26 A1001001A03D09B25228H1590418 A
03D09B25228H159041 14 1024  (XYZ_AdjReport)
 -> PID:  93 3 ICM8 icmnlsdb13 XYZ_AdjReport59 26 A1001001A03D09B25230B5448418 A
03D09B25230B544841 14 1024  (XYZ_AdjReport)
 -> PID:  93 3 ICM8 icmnlsdb13 XYZ_AdjReport59 26 A1001001A03D09B25231B5090118 A
03D09B25231B509011 14 1024  (XYZ_AdjReport)
Accessed Result Set Cursor.
Disconnecting from datastore & destroying reference...
Disconnected from datastore & destroying reference.

==========================================
Sample program completed.
==========================================
```

Figure 6-6 Output of sample InformationAccessICM.java

eClient

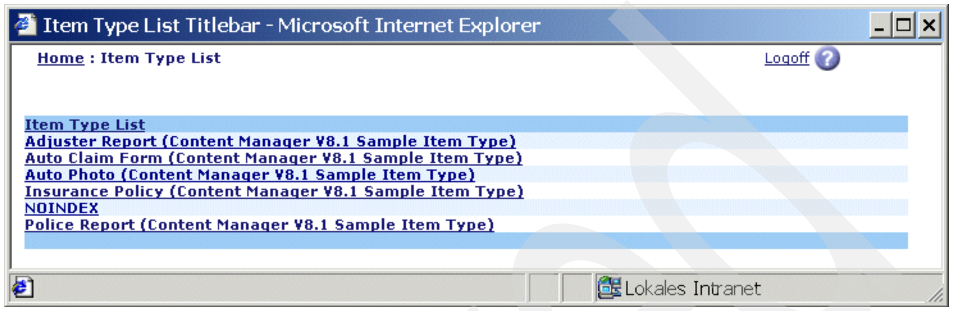
You can also use the eClient to access the sample data through the Content Manager connector. On the logon page, specify `ICMNLSDB(CM8)` for the server as shown in Figure 6-7.



User ID : icmadmin
Password : ********
Server : ICMNLSDB (CM8)

Figure 6-7 eClient logon with the Content Manager connector

After you click the **Search** link on the welcome page, the available item types are listed on the Item Type List page as shown in Figure 6-8. The item type description is displayed for each item type in the list rather than just the item type name. To search for the same items as in the sample application above, select one of the item types, **Auto Claim Form** or **Adjuster Report**.


Item Type List Titlebar - Microsoft Internet Explorer

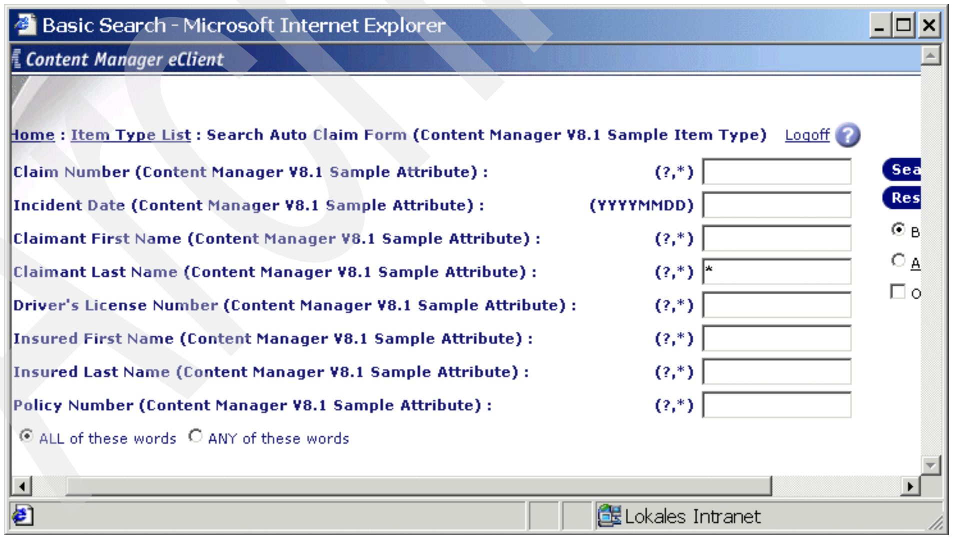
Home : Item Type List   Logoff

Item Type List
Adjuster Report (Content Manager V8.1 Sample Item Type)
Auto Claim Form (Content Manager V8.1 Sample Item Type)
Auto Photo (Content Manager V8.1 Sample Item Type)
Insurance Policy (Content Manager V8.1 Sample Item Type)
NOINDEX
Police Report (Content Manager V8.1 Sample Item Type)

Lokales Intranet

Figure 6-8 List of available item types

Figure 6-9 shows the search form that comes up if you select item type **Auto Claim Form**. To get all items of this type, specify the wildcard character in one of the fields and click the **Search** button. Because we search for Auto Claim Form items only, three results are returned as shown in Figure 6-10 on page 135. If you also want to see the other three results that we have seen in the sample application, you need to search for Adjuster Report items.


Basic Search - Microsoft Internet Explorer

Content Manager eClient

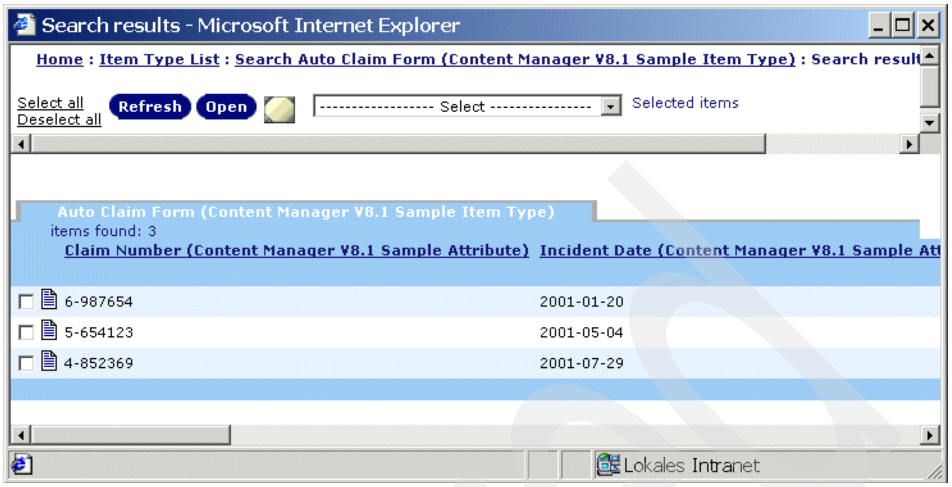
Home : Item Type List : Search Auto Claim Form (Content Manager V8.1 Sample Item Type)   Logoff

Claim Number (Content Manager V8.1 Sample Attribute) :	(?,*)	
Incident Date (Content Manager V8.1 Sample Attribute) :	(YYYYMMDD)	
Claimant First Name (Content Manager V8.1 Sample Attribute) :	(?,*)	
Claimant Last Name (Content Manager V8.1 Sample Attribute) :	(?,*)	*
Driver's License Number (Content Manager V8.1 Sample Attribute) :	(?,*)	
Insured First Name (Content Manager V8.1 Sample Attribute) :	(?,*)	
Insured Last Name (Content Manager V8.1 Sample Attribute) :	(?,*)	
Policy Number (Content Manager V8.1 Sample Attribute) :	(?,*)	

[✓ ALL of these words] ANY of these words

Lokales Intranet

Figure 6-9 Basic search page


Search results - Microsoft Internet Explorer

Home : Item Type List : Search Auto Claim Form (Content Manager V8.1 Sample Item Type) : Search result

Select all
Deselect all

Auto Claim Form (Content Manager V8.1 Sample Item Type)
items found: 3

Claim Number (Content Manager V8.1 Sample Attribute)	Incident Date (Content Manager V8.1 Sample Att
6-987654	2001-01-20
5-654123	2001-05-04
4-852369	2001-07-29

Figure 6-10 Search results page

6.2.3 Using DB2 connector

In this section, we demonstrate how to use a DB2 connector to access data through a sample Java application.

Java application

The sample application InformationAccessDB2.java shows how to use the DB2 connector to access the data, stored in a DB2 server. In our example, the database is IBMPRESS.

The main class of the connector is DKDatastoreDB2. You first have to call its connect method:

```
DKDatastoreDB2 dsDB2 = new DKDatastoreDB2();
dsDB2.connect(database,userName,password,"");
```

To find objects in the datastore, you can use SQL. In this sample, we search for all items where the source attribute is set to IBMPRESS.

The execute method of class DKDatastoreDB2 evaluates the query string command by executing it against the datastore and returning all results in a collection. You can also specify query options, for example to limit the number of returned results or the data that is retrieved for each result. The method returns a cursor that can be used to fetch the results:

```
String query = "SELECT SOURCE,ID FROM ICMADMIN.LONG_ARTICLES WHERE SOURCE =
'ibmpress'";
DKNVPair options[] = new DKNVPair[2];
```

```
options[0] = new DKNVPair(DKConstantDB2.DK_CM_PARM_MAX_RESULTS,"100");
options[1] = new DKNVPair(DKConstantDB2.DK_CM_PARM_END,null);
dkResultSetCursor cursor = dsDB2.execute(query,
DKConstantDB2.DK_CM_SQL_QL_TYPE, options);
```

The cardinality method returns the number of search results. Each call to the fetchNext method of the cursor returns a Dynamic Data Object (DDO) of type DKDDO:

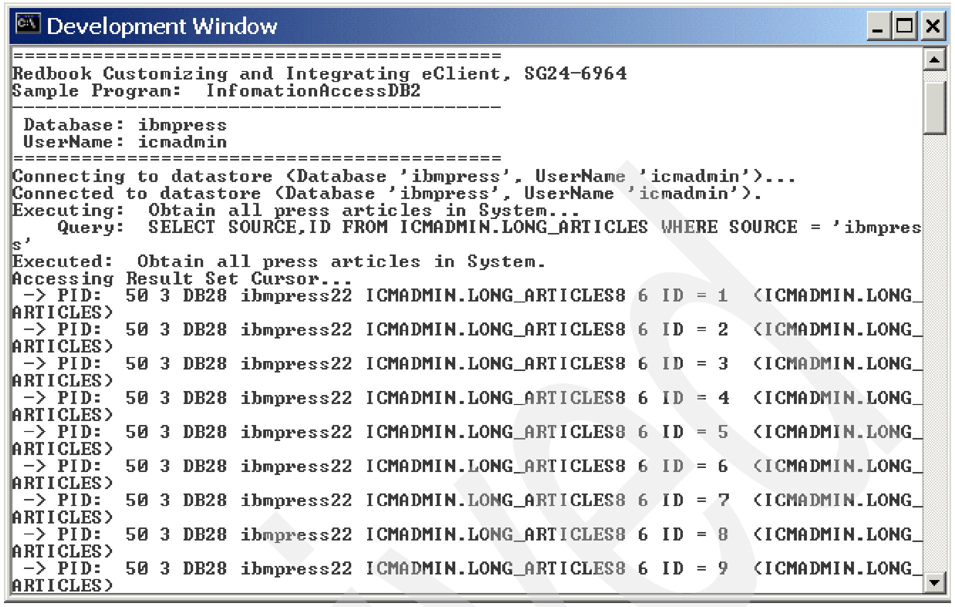
```
int resultCount = cursor.cardinality();
while(resultCount-- > 0) {
DKDDO ddo = cursor.fetchNext();
...
}
```

A DDO represents a persistent data object in the DB2 server. You can use it to retrieve and update the appropriate object data. You can also delete the object in the server.

To run the sample:

1. Select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window** to open a command window that has the required environment settings.
2. Change to the directory where you stored the redbook samples.
3. To compile the sample, run **`javac InformationAccessDB2.java`**.
4. To run the sample, enter **`java InformationAccessDB2`**.

For each resulting data object, its Persistent Identifier (PID) is displayed. The output of the sample is shown in Figure 6-11 on page 137.



```
==========================================
Redbook Customizing and Integrating eClient, SG24-6964
Sample Program:  InfomationAccessDB2
------------------------------------------
 Database: ibmpress
 UserName: icmadmin
==========================================
Connecting to datastore (Database 'ibmpress', UserName 'icmadmin')...
Connected to datastore (Database 'ibmpress', UserName 'icmadmin').
Executing:  Obtain all press articles in System...
    Query:  SELECT SOURCE,ID FROM ICMADMIN.LONG_ARTICLES WHERE SOURCE = 'ibmpres
s'
Executed:  Obtain all press articles in System.
Accessing Result Set Cursor...
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 1  (ICMADMIN.LONG_
ARTICLES)
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 2  (ICMADMIN.LONG_
ARTICLES)
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 3  (ICMADMIN.LONG_
ARTICLES)
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 4  (ICMADMIN.LONG_
ARTICLES)
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 5  (ICMADMIN.LONG_
ARTICLES)
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 6  (ICMADMIN.LONG_
ARTICLES)
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 7  (ICMADMIN.LONG_
ARTICLES)
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 8  (ICMADMIN.LONG_
ARTICLES)
 -> PID:  50 3 DB28 ibmpress22 ICMADMIN.LONG_ARTICLES8 6 ID = 9  (ICMADMIN.LONG_
ARTICLES)
```

Figure 6-11 Output of sample InformationAccessDB2.java

eClient

The DB2 connector is not available in the eClient. Hence, direct access to DB2 databases is not possible.

6.2.4 Using federated connector to access DB2

In this section, we demonstrate how to use the federated connector to access DB2 through sample Java applications and eClient.

Java application

The native attributes of the database IBMPRESS are mapped to the federated attributes in the sample administration database EIPSAMPL. The federated connector can be used to access the DB2 server using a federated query string. This is shown in the sample InformationAccessFedDB2.java.

Because there is also a search template (SearchLongBySource) where each of the federated attributes is a search criteria, the search template APIs can also be used. This is shown in the sample InformationAccessFedDB2Beans.java.

We also show how you can use the search template in the eClient.

Let's start with the federated query string sample.

The main class of the federated connector is DKDatastoreFed. You first have to call its connect method:

```
DKDatastoreFed dsFed = new DKDatastoreFed();
dsFed.connect(database,userName,password,"");
```

To find objects in the datastore, you can use a federated query string. This is a content server neutral query. It is documented in the Information Center. Select **Enterprise Information Portal -> Programming -> Working with a federated content server and federated searching -> Running federated queries**.

In this sample, we perform the same search as in 6.2.3, “Using DB2 connector” on page 135, but this time we use the federated attributes mapped to the DB2 native attributes.

A convenient way to execute the query string is to use the execute method of class DKFederatedQuery. The query is translated into a native query against the DB2 server. The translation information is obtained from the schema mapping. The datastore returns all results in a collection. You can also specify query options, for example to limit the number of returned results or the data that is retrieved for each result. The execute method returns a cursor that can be used to fetch the results:

```
String query = "PARAMETRIC_SEARCH = ( ENTITY = fed_long_article,
MAX_RESULTS=0, COND = ((fed_source == 'ibmpress')));
OPTION = (CONTENT = YES)";
DKFederatedQuery fedQuery = new DKFederatedQuery(dsFed, query);
fedQuery.execute(null);
dkResultSetCursor cursor = fedQuery.resultSetCursor();
```

The cardinality method returns the number of search results. Each call to the fetchNext method of the cursor returns a Dynamic Data Object (DDO) of type DKDDO:

```
int resultCount = cursor.cardinality();
while(resultCount-- > 0) {
DKDDO ddo = cursor.fetchNext();
...
}
```

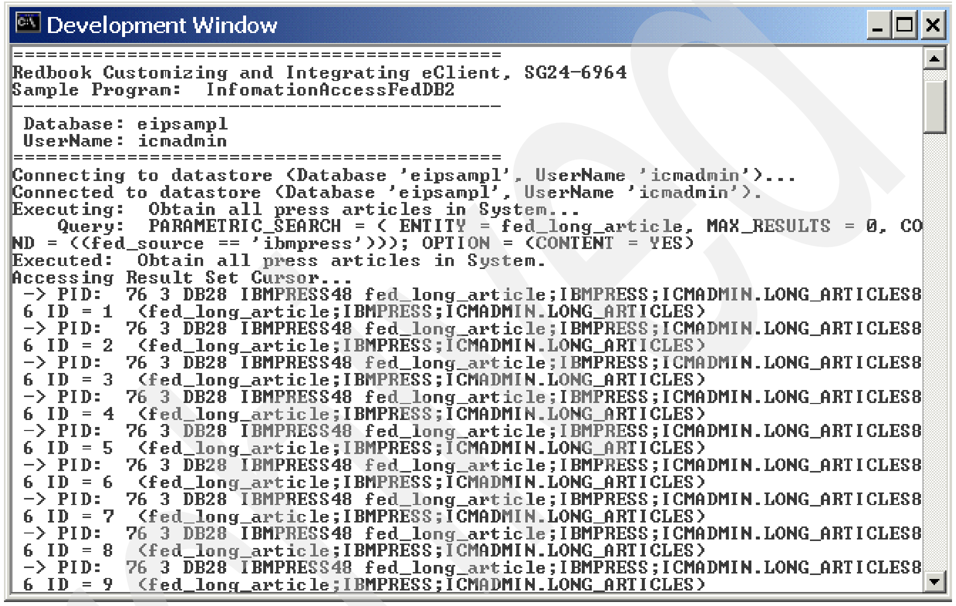
A DDO represents a persistent data object in the DB2 server. You can use it to retrieve and update the appropriate object data. You can also delete the object in the server.

To run the sample:

1. Select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window** to open a command window that has the required environment settings.

2. Change to the directory where you stored the redbook samples.
3. To compile the sample, run **`javac InformationAccessFedDB2.java`**.
4. To run the sample, enter: **`java InformationAccessFedDB2`**.

For each resulting data object, its Persistent Identifier (PID) is displayed. The PIDs now also contain the name of the federated entity. The output of the sample is shown in Figure 6-12.



```
===========================================
Redbook Customizing and Integrating eClient, SG24-6964
Sample Program:  InfomationAccessFedDB2
-------------------------------------------
 Database: eipsampl
 UserName: icmadmin
===========================================
Connecting to datastore (Database 'eipsampl', UserName 'icmadmin')...
Connected to datastore (Database 'eipsampl', UserName 'icmadmin').
Executing:  Obtain all press articles in System...
    Query:  PARAMETRIC_SEARCH = ( ENTITY = fed_long_article, MAX_RESULTS = 0, CO
ND = ((fed_source == 'ibmpress'))); OPTION = (CONTENT = YES)
Executed:  Obtain all press articles in System.
Accessing Result Set Cursor...
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 1  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 2  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 3  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 4  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 5  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 6  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 7  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 8  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 9  (fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES)
```

Figure 6-12 Output of sample InformationAccessFedDB2.java

The federated query string cannot be used in the eClient.

In the samples above, we create our queries manually. EIP, however, provides a convenient way to predefine a federated query using search templates. With a search template, you just have to provide the search values for certain search criteria and run the query. You do not have to deal with federated attributes and create a query string.

Creating an application that uses search templates is simple if you use the JavaBeans API. This is shown in the sample InformationAccessFedDB2Beans.java.

Using the JavaBeans API, a connection is established with an object of type CMBConnection:

```
CMBConnection connection = new CMBConnection();
connection.setServerName(database);
connection.setUserid(userName);
connection.setPassword(password);
connection.connect();
```

In this sample, we perform the same search as in the previous two samples but instead of using a query string, we use a search template.

The template can be retrieved by name. A search value and an operator need to be specified for each of the required search criteria. You can also specify query options, for example to limit the number of returned results or to run the query asynchronously. The runQueryWithCursor method returns a cursor that can be used to fetch the results:

```
CMBSchemaManagement schema = connection.getSchemaManagement();
CMBSearchTemplate searchTemplate = schema.getSearchTemplate(templateName);
String[] searchValues = { "ibmpress" };
searchTemplate.setSearchCriterion("source",
CMBBaseConstant.CMB_OP_EQUAL, searchValues);
searchTemplate.setAsynchSearch(false);
searchTemplate.setMaxResults(100);
CMBResultSetCursor cursor = searchTemplate.runQueryWithCursor();
```

The getResultCount method returns the number of search results. Each call to the fetchNext method of the cursor returns a CMBItem object:

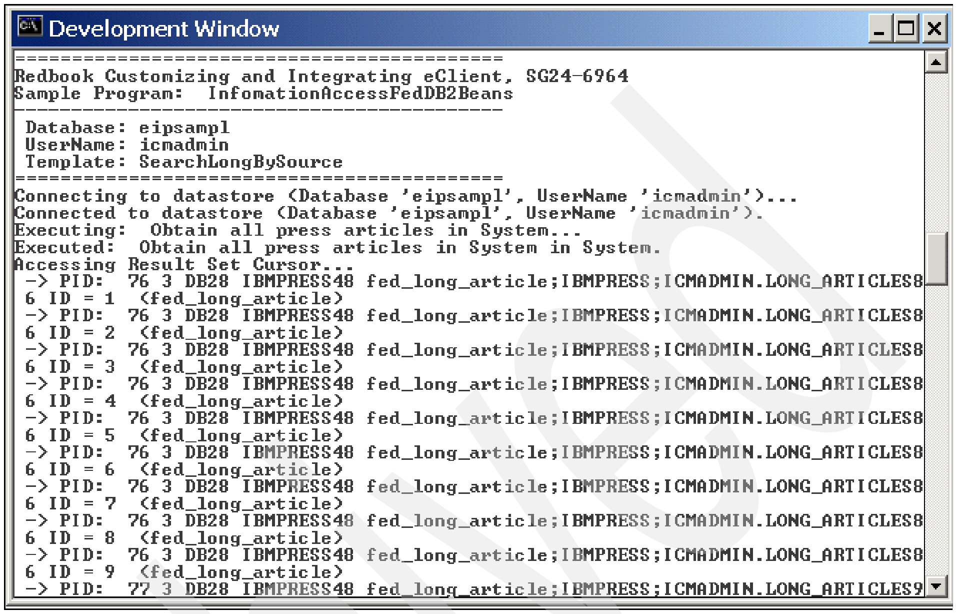
```
int resultCount = cursor.getResultCount();
while(resultCount-- > 0) {
CMBItem item = cursor.fetchNext();
...
}
```

An item represents a persistent data object in the content server (which is DB2 in this case). You can use it to retrieve and update the appropriate object data. You can also delete the object in the server.

To run the sample:

1. Select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window** to open a command window that has the required environment settings.
2. Change to the directory where you stored the redbook samples.
3. To compile the sample, run **`javac InformationAccessFedDB2Beans.java`**.
4. To run the sample, enter **`java InformationAccessFedDB2Beans`**.

For each resulting item, its Persistent Identifier (PID) is displayed. The output of the sample is shown in Figure 6-13.

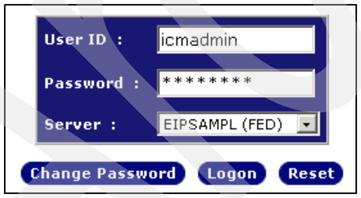


```
==========================================
Redbook Customizing and Integrating eClient, SG24-6964
Sample Program:  InfomationAccessFedDB2Beans
------------------------------------------
 Database: eipsampl
 UserName: icmadmin
 Template: SearchLongBySource
==========================================
Connecting to datastore (Database 'eipsampl', UserName 'icmadmin')...
Connected to datastore (Database 'eipsampl', UserName 'icmadmin').
Executing:  Obtain all press articles in System...
Executed:  Obtain all press articles in System in System.
Accessing Result Set Cursor...
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 1  (fed_long_article)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 2  (fed_long_article)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 3  (fed_long_article)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 4  (fed_long_article)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 5  (fed_long_article)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 6  (fed_long_article)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 7  (fed_long_article)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 8  (fed_long_article)
 -> PID:  76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8
 6 ID = 9  (fed_long_article)
 -> PID:  77 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES9
```

Figure 6-13 Output of sample InformationAccessFedDB2Beans.java

eClient

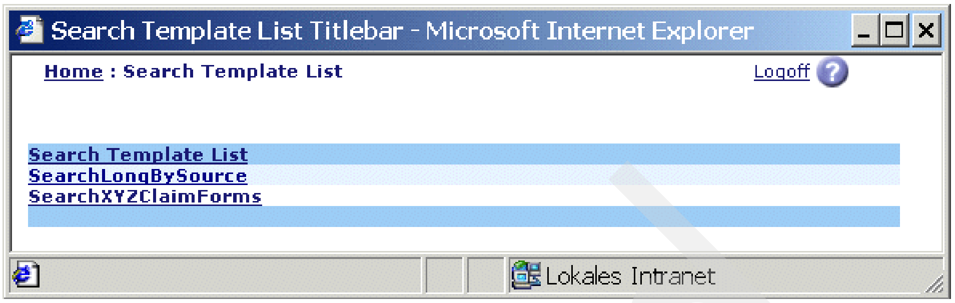
You can also use the eClient to search with this template. On the logon page, specify EIPSAMPL (FED) for the server as shown in Figure 6-14.



User ID : icmadmin
Password : ********
Server : EIPSAMPL (FED)

Figure 6-14 eClient logon with federated connector

The available search templates are listed on the Search Template List page as shown in Figure 6-15 on page 142. To search for the same items as in the sample application above, select the search template **SearchLongBySource**.


Search Template List Titlebar - Microsoft Internet Explorer

Home : Search Template List

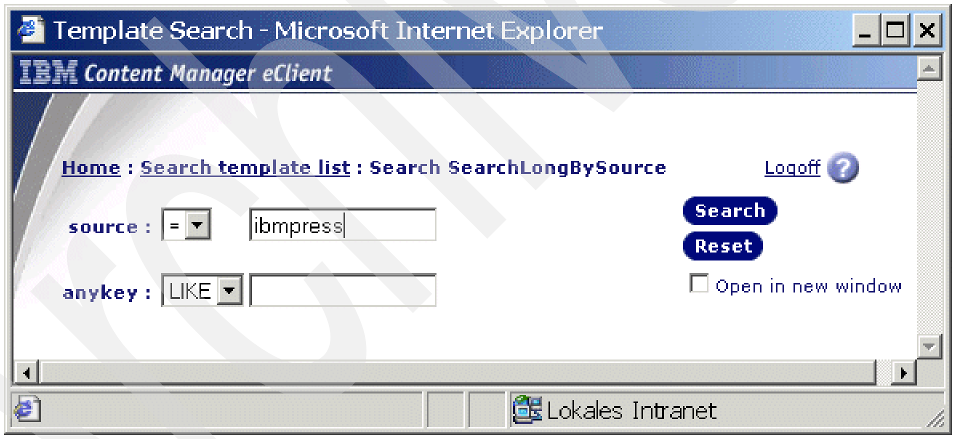
Logoff

Search Template List
SearchLongBySource
SearchXYZClaimForms

Lokales Intranet

Figure 6-15   List of available search templates

In the search form, make sure the search value for the source criteria is `ibmpress` (lowercase) and no value is specified for the anykey criteria as shown in Figure 6-16. Click the **Search** button and you will get a result list as in Figure 6-17 on page 143. Because such a result list does not really help you to find a certain document, we will discuss how you can organize the data and enhance the result list in 13.1, “Using categories and summaries in eClient searches” on page 282.


Template Search - Microsoft Internet Explorer

IBM Content Manager eClient

Home : Search template list : Search SearchLongBySource

Logoff

source : = ibmpress

anykey : LIKE

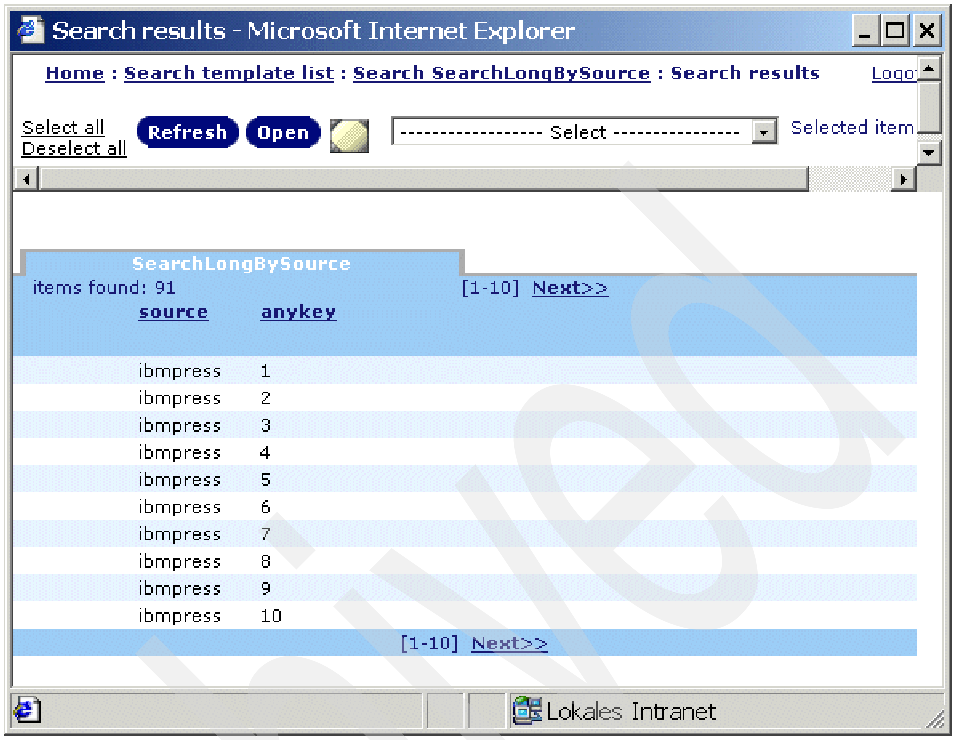
Search

Reset

Open in new window

Lokales Intranet

Figure 6-16   Template search page


Search results - Microsoft Internet Explorer

Home : Search template list : Search SearchLongBySource : Search results   Logo

Select all
Deselect all
Refresh Open
------------------ Select -----------------
Selected item

SearchLongBySource

items found: 91   [1-10] Next>>

source	anykey
ibmpress	1
ibmpress	2
ibmpress	3
ibmpress	4
ibmpress	5
ibmpress	6
ibmpress	7
ibmpress	8
ibmpress	9
ibmpress	10

[1-10] Next>>

Lokales Intranet

Figure 6-17 Search template result page

6.2.5 Using federated connector to search across content servers

In the examples before, we use EIP to access a single content server; however, one of the key features of EIP is the ability to search across multiple content servers.

This can be done using a federated query that uses federated attributes that are mapped to native attributes of different content servers.

System configuration

Figure 6-18 on page 144 shows how you can define a federated entity that allows cross-server search using the sample databases.

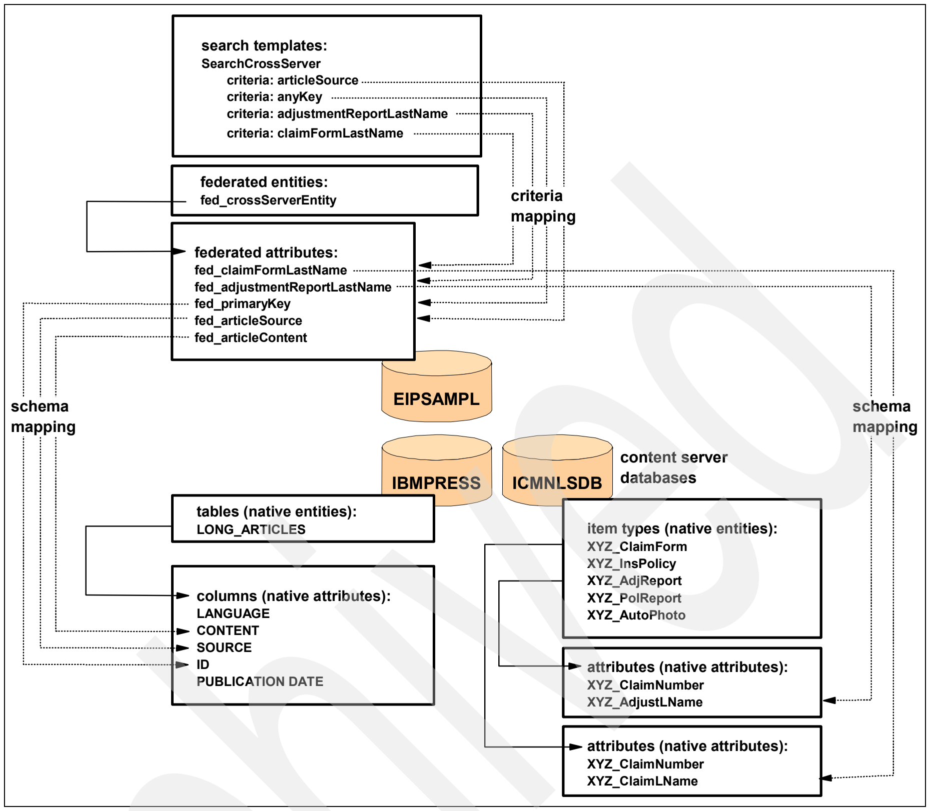


Figure 6-18 Cross-server search definitions

Three of the federated attributes of the entity are mapped to native attributes in the DB2 database IBMPRESS. The other two federated attributes are mapped to native attributes in the Content Manager database. The federated entity can then directly be used in a federated query string. Figure 6-18 shows also a search template that uses this entity to allow a cross-server search.

The only purpose of the federated entity described above is to retrieve documents from multiple content servers simultaneously. Beyond this purpose, it is hard to find a scenario where you have attributes of insurance documents and press articles in one entity. Imagine a federated entity that has an author attribute that is mapped to two native attributes, an author attribute in a DB2 database, and an originator attribute in Content Manager. Such a federated entity adds another dimension to the retrieval from multiple servers, because this way you bring two similar concepts together in one federated view.

In the following section, we configure the federated connector according to Figure 6-18 and run some searches.

Follow these steps to configure the connector:

1. Log on to the System Administration Client, using server type Enterprise Information Portal and server EIPSAMPL.

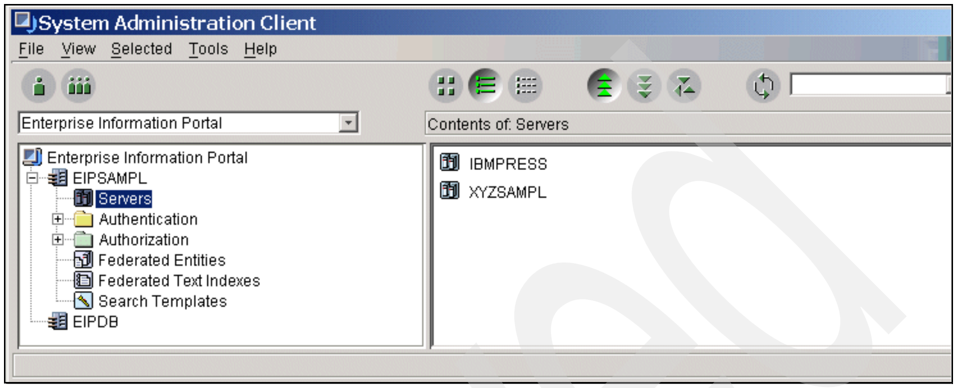


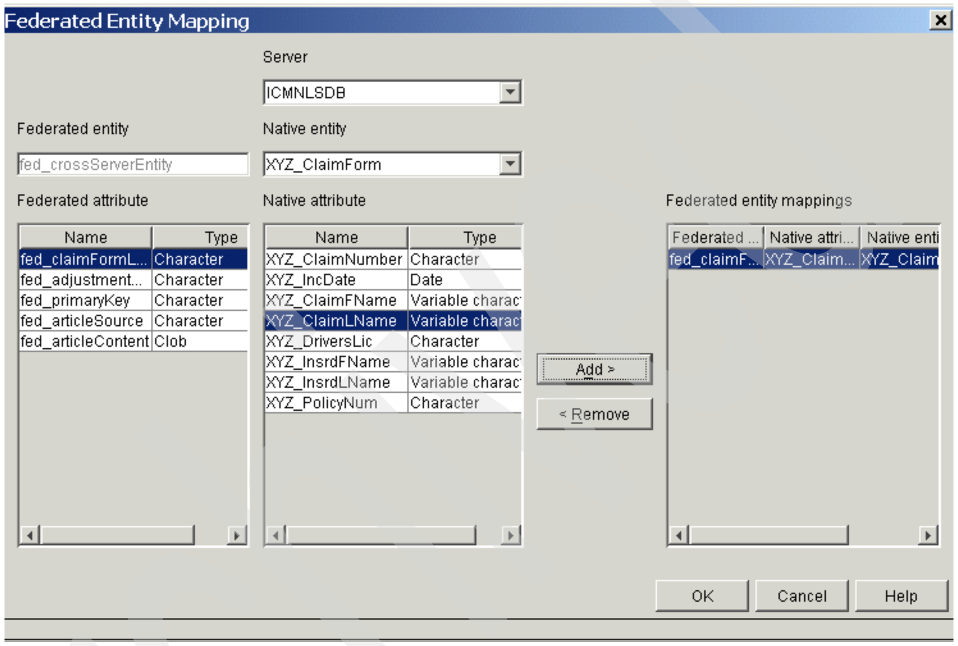
Figure 6-19 Configuring EIPSAMPL in System Administration Client

2. In the tree view for database EIPSAMPL, right-click **Servers** and select **New -> Content Manager v8**.
3. In the New Server window, enter `ICMNLSDB` as the server name and click **OK**. The new server should appear in the list of servers on the right-hand side, along with IBMPRESS and XYZSAMPL.
4. In the menu bar, select **Tools -> Server Inventory Viewer**.
5. In the Server Inventory Viewer, select **View -> Refresh Server -> ICMNLSDB**. Some lines for server ICMNLSDB will appear in the viewer. Close the viewer.
6. Back in the tree view for EIPSAMPL, right-click **Federated Entities** and select **New -> Nonwizard**.
7. In the New Federated Entity window, enter the name `fed_crossServerEntity` and click **Add**.
8. In the New Federated Attribute window, enter the name `fed_claimFormLastName`. Do not change any other value and click **Apply**. In the same way, add the following attributes:
   – `fed_adjustmentReportLastName`
   – `fed_primaryKey`
   – `fed_articleSource`
   – `fed_articleContent` (You must specify type `Clob` a length of `102400` for this attribute in the New Federated Attribute window)

   Click **Cancel** to close that window.

9. Back in the Federated Entity window, click **Map Federated Entity...**.

10. In the Federated Entity Mapping window, select server **ICMNLSDB** and the native entity **XYZ_ClaimForm**. Now select the federated attribute **fed_claimFormLastName** and the native attribute **XYZ_ClaimLName**. Finally, click **Add** to create the mapping. Now the window looks similar to Figure 6-20.



Server: ICMNLSDB

Federated entity: fed_crossServerEntity

Native entity: XYZ_ClaimForm

Federated attribute

Name	Type
fed_claimFormL...	Character
fed_adjustment...	Character
fed_primaryKey	Character
fed_articleSource	Character
fed_articleContent	Clob

Native attribute

Name	Type
XYZ_ClaimNumber	Character
XYZ_IncDate	Date
XYZ_ClaimFName	Variable charac
XYZ_ClaimLName	Variable charac
XYZ_DriversLic	Character
XYZ_InsrdFName	Variable charac
XYZ_InsrdLName	Variable charac
XYZ_PolicyNum	Character

Federated entity mappings

Federated ...	Native attri...	Native enti
fed_claimF...	XYZ_Claim...	XYZ_Claim

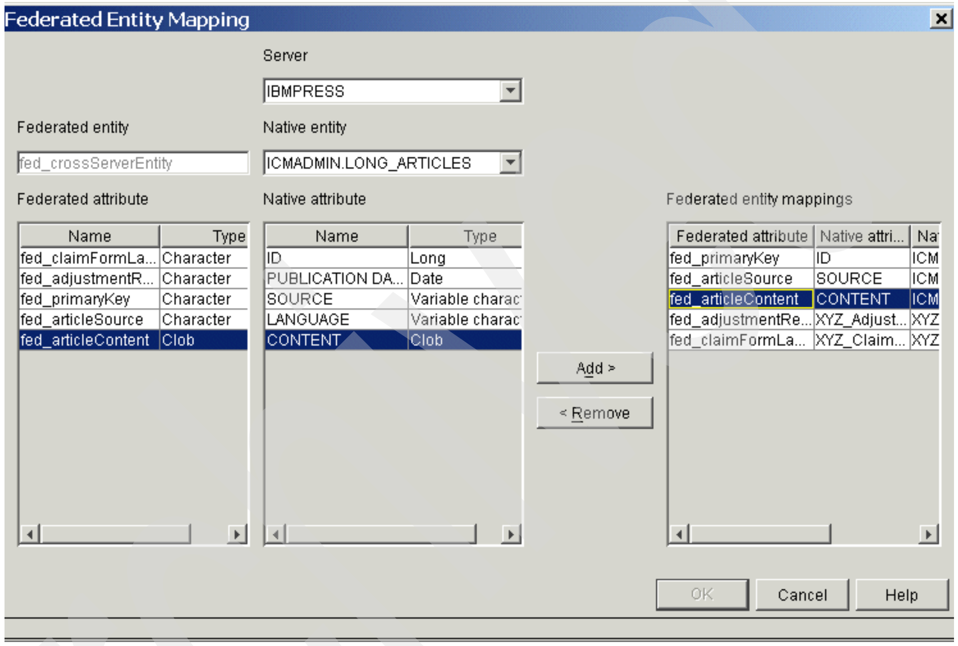
Figure 6-20 First federated entity mapping

In the same way, create the mappings for the remaining federated attributes according to Table 6-1. After this is done, your Federated Entity Mapping window should look similar to Figure 6-21 on page 147.

Table 6-1 Required mappings

Federated attribute	Native attribute	Native Entity	Server
fed_claimFormLast Name	XYZ_ClaimLName	XYZ_ClaimForm	ICMNLSDB
fed_adjustmentRe portLastName	XYZ_AdjustLName	XYZ_AdjReport	ICMNLSDB
fed_primaryKey	ID	ICMADMIN.LONG _ARTICLES	IBMPRESS

Federated attribute	Native attribute	Native Entity	Server
fed_articleSource	SOURCE	ICMADMIN.LONG_ARTICLES	IBMPRESS
fed_articleContent	CONTENT	ICMADMIN.LONG_ARTICLES	IBMPRESS



Server: IBMPRESS

Federated entity: fed_crossServerEntity

Native entity: ICMADMIN.LONG_ARTICLES

Federated attribute:

Name	Type
fed_claimFormLa...	Character
fed_adjustmentR...	Character
fed_primaryKey	Character
fed_articleSource	Character
fed_articleContent	Clob

Native attribute:

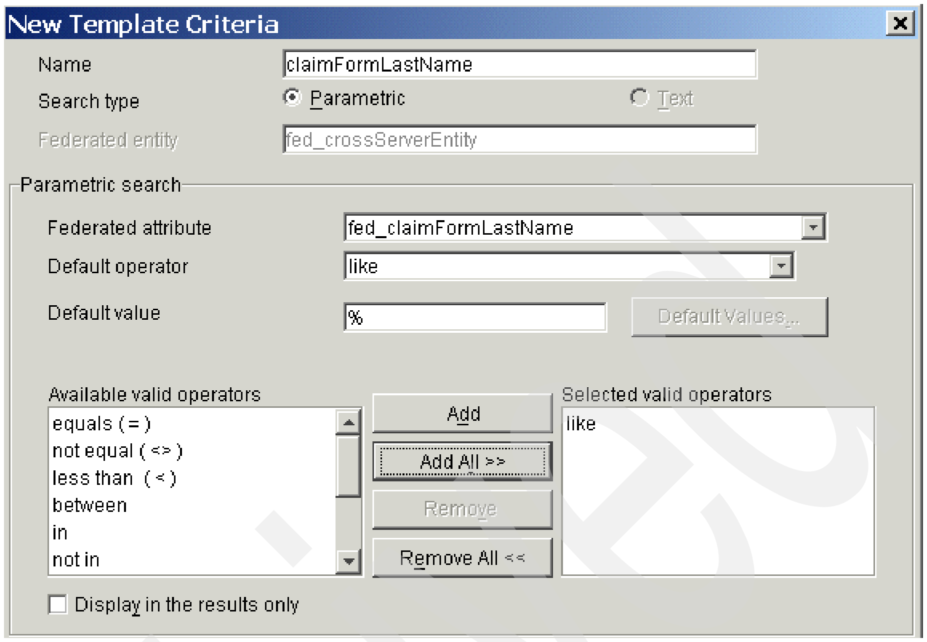
Name	Type
ID	Long
PUBLICATION DA...	Date
SOURCE	Variable charac
LANGUAGE	Variable charac
CONTENT	Clob

Federated entity mappings:

Federated attribute	Native attri...	Na
fed_primaryKey	ID	ICM
fed_articleSource	SOURCE	ICM
fed_articleContent	CONTENT	ICM
fed_adjustmentRe...	XYZ_Adjust...	XYZ
fed_claimFormLa...	XYZ_Claim...	XYZ

Figure 6-21 Federated Entity Mapping window

11. Click **OK** to close this window and again click **OK** to close the New Federated Entity window.
12. Back in the tree view for database EIPSAMPL, right-click **Search Templates** and select **New -> Nonwizard**.
13. Enter the name `SearchCrossServer`. Select the user **ICMADMIN** and click **Add** to move it to the list of selected users. Select the federated entity **fed_crossServerEntity** and click **Add...** to create template criteria.
14. In the New Template Criteria window, enter the name `claimFormLastName`, select the federated attribute **fed_claimFormLastName**, select the default operator **like**, specify `%` as the default value and make sure the Display in the results only box is not checked. Now the window looks like Figure 6-22 on page 148.


New Template Criteria

Name: claimFormLastName

Search type: [✓ Parametric] Text

Federated entity: fed_crossServerEntity

Parametric search

Federated attribute: fed_claimFormLastName

Default operator: like

Default value: %

Available valid operators: equals (=), not equal (<>), less than (<), between, in, not in

Selected valid operators: like

Display in the results only: not checked

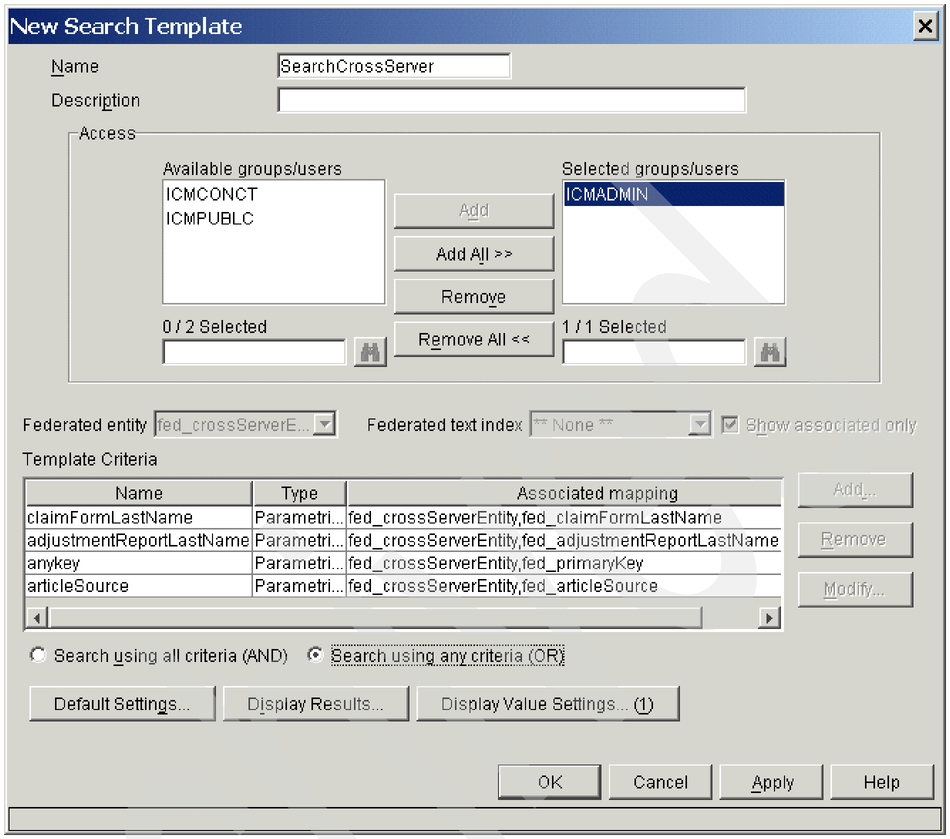
Figure 6-22 The first template criteria

Click **Apply** to add the new criteria. In the same way create the remaining template criteria according to Table 6-2. After this is done, close this window, by clicking **Cancel**.

Table 6-2 Required template criteria

Criteria name	**Federated attribute**	**Default operator**	**Default value**	**Display in the results only**
claimFormLastName	fed_claimFormLastName	like	%	not checked
adjustmentReportLastName	fed_adjustmentReportLastName	like	%	not checked
articleSource	fed_articleSource	equals (=)	ibmpress	not checked
anykey	fed_primaryKey			checked

15. Back in the new Search Template window, select **Search using any criteria (OR)** at the bottom of the window. The window should now look like Figure 6-23 on page 149.


New Search Template

Name: SearchCrossServer

Description:

Access

Available groups/users: ICMCONCT, ICMPUBLC — 0 / 2 Selected

Selected groups/users: ICMADMIN — 1 / 1 Selected

Federated entity: fed_crossServerE... Federated text index: ** None ** [✓ Show associated only]

Template Criteria

Name	Type	Associated mapping
claimFormLastName	Parametri...	fed_crossServerEntity,fed_claimFormLastName
adjustmentReportLastName	Parametri...	fed_crossServerEntity,fed_adjustmentReportLastName
anykey	Parametri...	fed_crossServerEntity,fed_primaryKey
articleSource	Parametri...	fed_crossServerEntity,fed_articleSource

Search using all criteria (AND) [✓ Search using any criteria (OR)]

Figure 6-23 New search template window

16. Click **OK** to close the window. The new search template appears in the list of available templates.

The federated connector is now configured according to Figure 6-18 on page 144.

Java application

To execute the cross-server search, the federated entity can directly be used in a federated query string. This is shown in the sample InformationAccessFed.java.

The sample InformationAccessFedBeans.java shows how to search with the search template.

Let's start with the federated query string sample. Note that we can follow the same procedure as in 6.2.4, "Using federated connector to access DB2" on page 137, as the federated query string is server agnostic.

The main class of the federated connector is DKDatastoreFed. You first have to call its connect method:

```
DKDatastoreFed dsFed = new DKDatastoreFed();
dsFed.connect(database,userName,password,"");
```

To find objects in the datastore, you can use a federated query string. This is a content server neutral query.

A convenient way to execute the query string is to use the execute method of class DKFederatedQuery. The query is translated into native queries against the DB2 server and the Content Manager server. The translation information is obtained from the schema mapping. The results of the native queries are merged into a federated collection. You can specify query options, for example to limit the number of returned results or the data that is retrieved for each result. The execute method returns a cursor that can be used to fetch the results:

```
String query = "PARAMETRIC_SEARCH = ( ENTITY = fed_crossServerEntity,
MAX_RESULTS = 0, COND = ((fed_claimFormLastName LIKE '%') OR
(fed_adjustmentReportLastName LIKE '%') OR (fed_articleSource LIKE
'%ibmpress%'))); OPTION = (CONTENT = YES)";
DKFederatedQuery fedQuery = new DKFederatedQuery(dsFed, query);
fedQuery.execute(null);
dkResultSetCursor cursor = fedQuery.resultSetCursor();
```

A call to the fetchNext method of the cursor returns a Dynamic Data Object (DDO) of type DKDDO. The complete result set has been retrieved if the method returns null:

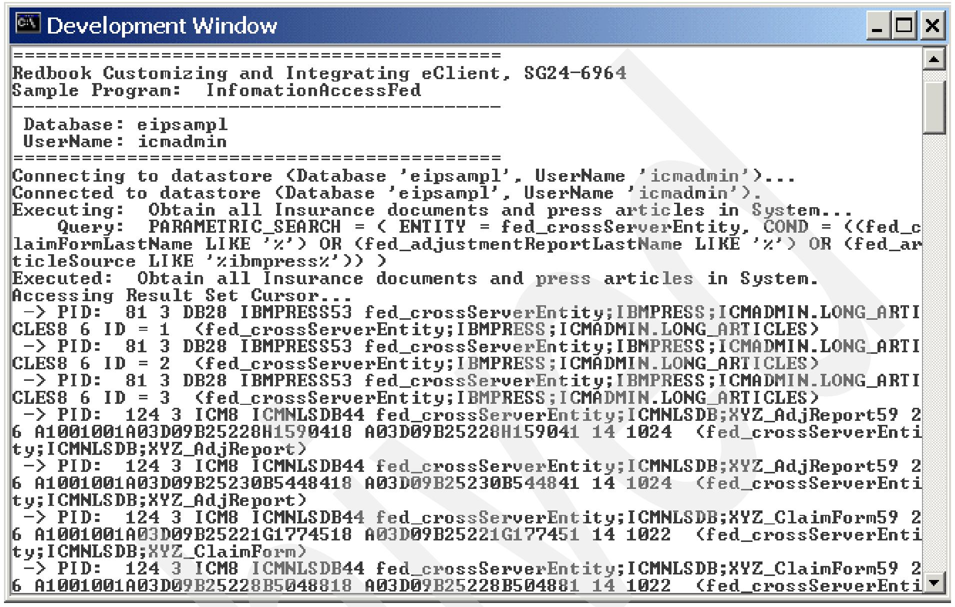
```
DKDDO  ddo;
while((ddo = cursor.fetchNext())!=null) {
...
}
```

A DDO represents a persistent data object either in the DB2 server or the Content Manager server. You can use it to retrieve and update the appropriate object data. You can also delete the object in the server.

To run the sample:

1. Select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window** to open a command window that has the required environment settings.
2. Change to the directory where you stored the redbook samples.
3. To compile the sample, run **`javac InformationAccessFed.java`**.
4. To run the sample, enter **`java InformationAccessFed`**.

For each resulting data object, its Persistent Identifier (PID) is displayed. The output of the sample is shown in Figure 6-24. As you can see, we get results from both content servers.



```
=========================================
Redbook Customizing and Integrating eClient, SG24-6964
Sample Program:  InfomationAccessFed
-----------------------------------------
 Database: eipsampl
 UserName: icmadmin
=========================================
Connecting to datastore (Database 'eipsampl', UserName 'icmadmin')...
Connected to datastore (Database 'eipsampl', UserName 'icmadmin').
Executing:  Obtain all Insurance documents and press articles in System...
    Query:  PARAMETRIC_SEARCH = ( ENTITY = fed_crossServerEntity, COND = ((fed_c
laimFormLastName LIKE '%') OR (fed_adjustmentReportLastName LIKE '%') OR (fed_ar
ticleSource LIKE '%ibmpress%')) )
Executed:  Obtain all Insurance documents and press articles in System.
Accessing Result Set Cursor...
 -> PID:  81 3 DB28 IBMPRESS53 fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTI
CLES8 6 ID = 1  (fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  81 3 DB28 IBMPRESS53 fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTI
CLES8 6 ID = 2  (fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  81 3 DB28 IBMPRESS53 fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTI
CLES8 6 ID = 3  (fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTICLES)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_AdjReport59 2
6 A1001001A03D09B25228H1590418 A03D09B25228H159041 14 1024  (fed_crossServerEnti
ty;ICMNLSDB;XYZ_AdjReport)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_AdjReport59 2
6 A1001001A03D09B25230B5448418 A03D09B25230B544841 14 1024  (fed_crossServerEnti
ty;ICMNLSDB;XYZ_AdjReport)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_ClaimForm59 2
6 A1001001A03D09B25221G1774518 A03D09B25221G177451 14 1022  (fed_crossServerEnti
ty;ICMNLSDB;XYZ_ClaimForm)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_ClaimForm59 2
6 A1001001A03D09B25228B5048818 A03D09B25228B504881 14 1022  (fed_crossServerEnti
```

Figure 6-24 Output of sample InformationAccessFed.java

The federated query string cannot be used in the eClient.

The sample InformationAccessFedDB2Beans.java performs the same search with the search template and the JavaBeans API.

Using the JavaBeans API a connection is established with an object of type CMBConnection:

```
CMBConnection connection = new CMBConnection();
connection.setServerName(database);
connection.setUserid(userName);
connection.setPassword(password);
connection.connect();
```

The template can be retrieved by name. A search value and an operator need to be specified for each of the required search criteria. You can also specify query options, for example to limit the number of returned results or to run the query asynchronously. The runQuery method does not return the result collection directly, but an object of type CMBSearchResults collect all the files and can be used to fetch the following results:

```
CMBSchemaManagement schema = connection.getSchemaManagement();
CMBSearchTemplate searchTemplate = schema.getSearchTemplate(templateName);
String[] searchValues = { "ibmpress" };
searchTemplate.setSearchCriterion("articleSource",
CMBBaseConstant.CMB_OP_EQUAL, searchValues);
searchValues = new String[] { "%" };
searchTemplate.setSearchCriterion("claimFormLastName",
CMBBaseConstant.CMB_OP_LIKE, searchValues);
searchTemplate.setSearchCriterion("adjustmentReportLastName",
CMBBaseConstant.CMB_OP_LIKE, searchValues);
searchTemplate.setMaxResults(100);
searchTemplate.setAsynchSearch(false);
searchTemplate.runQuery();
CMBSearchResults result = new CMBSearchResults();
result.setConnection(connection);
result.newResults(searchTemplate.getResults());
```

The getResultCount method returns the number of search results. Each call to the fetchNext method of the cursor returns a CMBItem object:

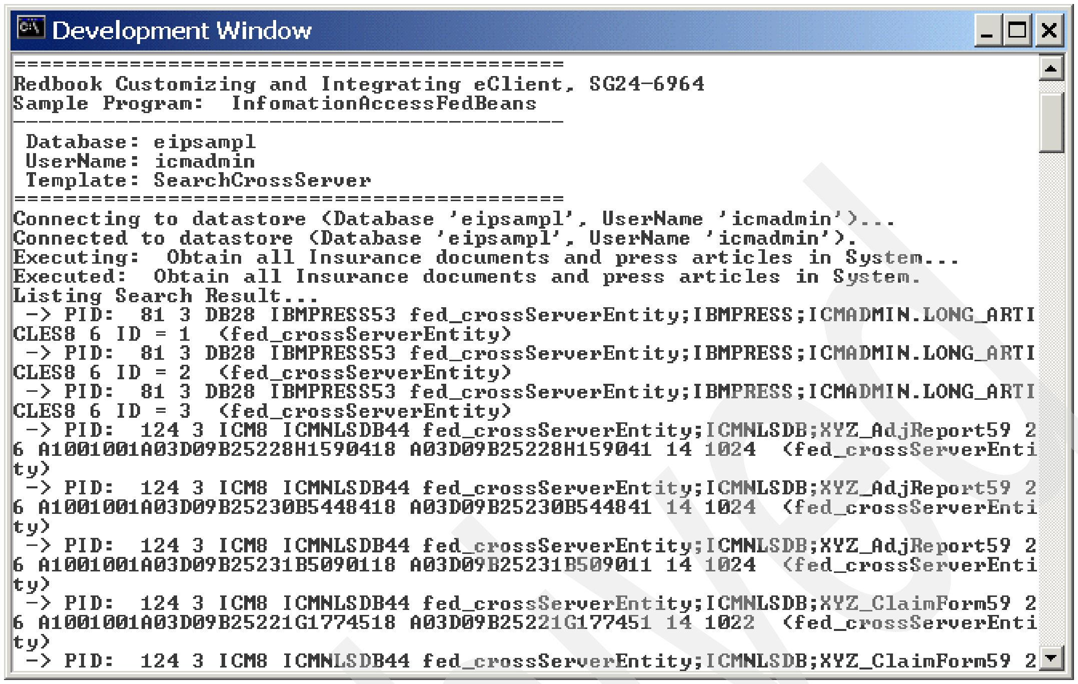
```
int resultCount = cursor.getResultCount();
while(resultCount-- > 0) {
CMBItem item = cursor.fetchNext();
...
}
```

An item represents a persistent data object in either the DB2 server or the Content Manager server. You can use it to retrieve and update the appropriate object data. You can also delete the object in the server.

To run the sample:

1. Select **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window** to open a command window that has the required environment settings.
2. Change to the directory where you stored the redbook samples.
3. To compile the sample, run **`javac InformationAccessFedBeans.java`**.
4. To run the sample, enter **`java InformationAccessFedBeans`**.

For each resulting item, its Persistent Identifier (PID) is displayed. The output of the sample is shown in Figure 6-25 on page 153. Again, we get results from both content servers.

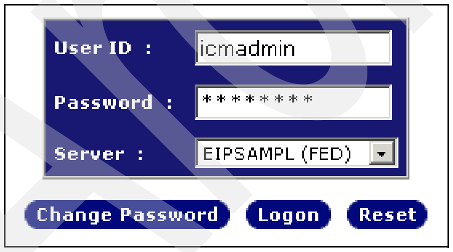


```
===========================================
Redbook Customizing and Integrating eClient, SG24-6964
Sample Program:  InfomationAccessFedBeans
-------------------------------------------
 Database: eipsampl
 UserName: icmadmin
 Template: SearchCrossServer
===========================================
Connecting to datastore (Database 'eipsampl', UserName 'icmadmin')...
Connected to datastore (Database 'eipsampl', UserName 'icmadmin').
Executing:  Obtain all Insurance documents and press articles in System...
Executed:  Obtain all Insurance documents and press articles in System.
Listing Search Result...
 -> PID:  81 3 DB28 IBMPRESS53 fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTI
CLES8 6 ID = 1  (fed_crossServerEntity)
 -> PID:  81 3 DB28 IBMPRESS53 fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTI
CLES8 6 ID = 2  (fed_crossServerEntity)
 -> PID:  81 3 DB28 IBMPRESS53 fed_crossServerEntity;IBMPRESS;ICMADMIN.LONG_ARTI
CLES8 6 ID = 3  (fed_crossServerEntity)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_AdjReport59 2
6 A1001001A03D09B25228H1590418 A03D09B25228H159041 14 1024  (fed_crossServerEnti
ty)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_AdjReport59 2
6 A1001001A03D09B25230B5448418 A03D09B25230B544841 14 1024  (fed_crossServerEnti
ty)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_AdjReport59 2
6 A1001001A03D09B25231B5090118 A03D09B25231B509011 14 1024  (fed_crossServerEnti
ty)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_ClaimForm59 2
6 A1001001A03D09B25221G1774518 A03D09B25221G177451 14 1022  (fed_crossServerEnti
ty)
 -> PID:  124 3 ICM8 ICMNLSDB44 fed_crossServerEntity;ICMNLSDB;XYZ_ClaimForm59 2
```

Figure 6-25   Output of sample InformationAccessFedBeans.java

eClient

The eClient is also able to search with this template. On the logon page, specify `EIPSAMPL (FED)` for the server, as shown in Figure 6-26.



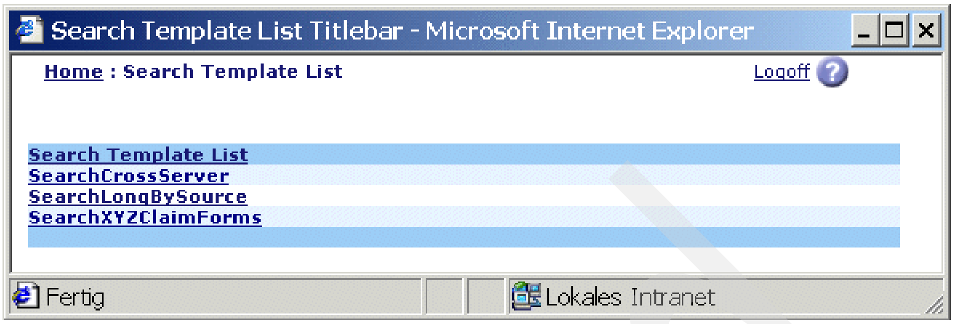
User ID : icmadmin

Password : ********

Server : EIPSAMPL (FED)

Figure 6-26   eClient logon with federated connector

The available search templates are listed on the Search Template List page as shown in Figure 6-27 on page 154. Select the search template **SearchCrossServer**.



Search Template List Titlebar - Microsoft Internet Explorer

Home : Search Template List

Logoff

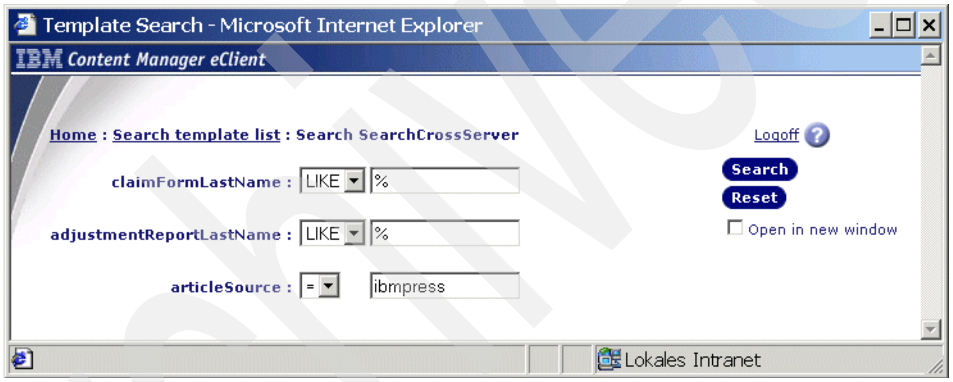
Search Template List
SearchCrossServer
SearchLongBySource
SearchXYZClaimForms

Fertig

Lokales Intranet

Figure 6-27 List of available search templates

In the search form, leave all default values and just click the **Search** button. You get a result list similar to Figure 6-28.



Template Search - Microsoft Internet Explorer

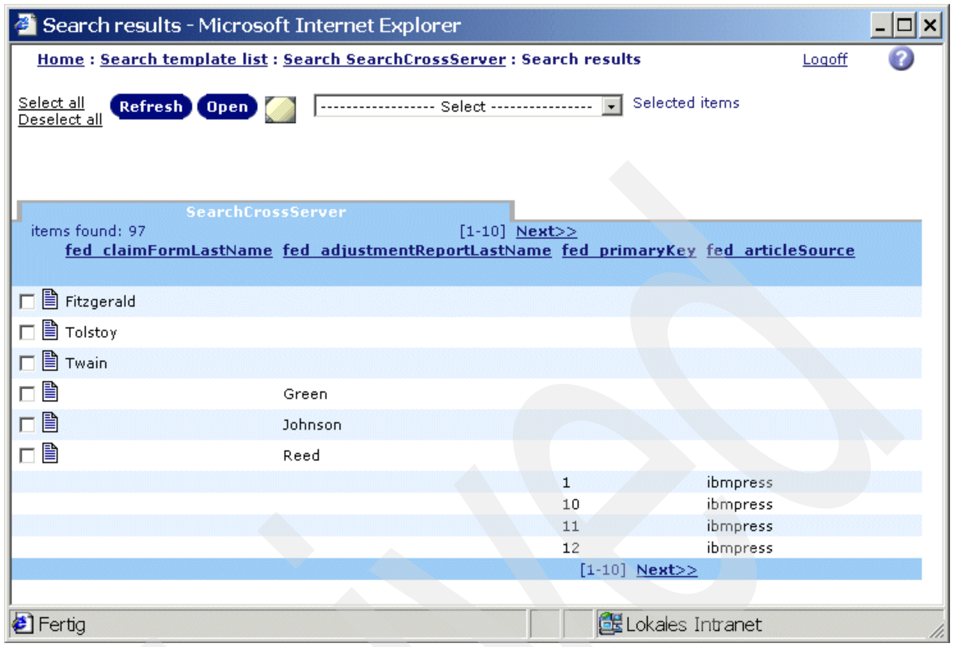
IBM Content Manager eClient

Home : Search template list : Search SearchCrossServer

Logoff

claimFormLastName :	LIKE	%
adjustmentReportLastName :	LIKE	%
articleSource :	=	ibmpress

Figure 6-28 Template Search window


Search results - Microsoft Internet Explorer

Home : Search template list : Search SearchCrossServer : Search results   Logoff

Select all
Deselect all   Refresh   Open   ------------------ Select ------------------   Selected items

SearchCrossServer

items found: 97   [1-10] Next>>

	fed_claimFormLastName	fed_adjustmentReportLastName	fed_primaryKey	fed_articleSource
☐	Fitzgerald			
☐	Tolstoy			
☐	Twain			
☐		Green		
☐		Johnson		
☐		Reed		
			1	ibmpress
			10	ibmpress
			11	ibmpress
			12	ibmpress

[1-10] Next>>

Fertig   Lokales Intranet

Figure 6-29 Search Template result window

6.2.6 Working with Information Mining Service

As discussed in 1.4, “Information Mining Service” on page 9, this feature of EIP provides functions for automatic text analysis. The analysis results, such as important terms, categories, and summaries can be used to organize the data. For example, you can structure your document collection by adding category information to each document, finding related documents by creating queries containing important terms, and providing a short summary for each document in result lists.

In the following sample, we analyze each article in the sample database IBMPRESS to find out to which topic (category) the article belongs. We then add another analysis step that creates a summary of each article.

If you want to know how you use this type of analysis to organize your data in Content Manager and restrict a search to a certain document topic, refer to 13.1, “Using categories and summaries in eClient searches” on page 282.

Running the categorization sample

Before you can start to determine the category for an article, you first need to:

1. Define the set of available categories, a taxonomy.

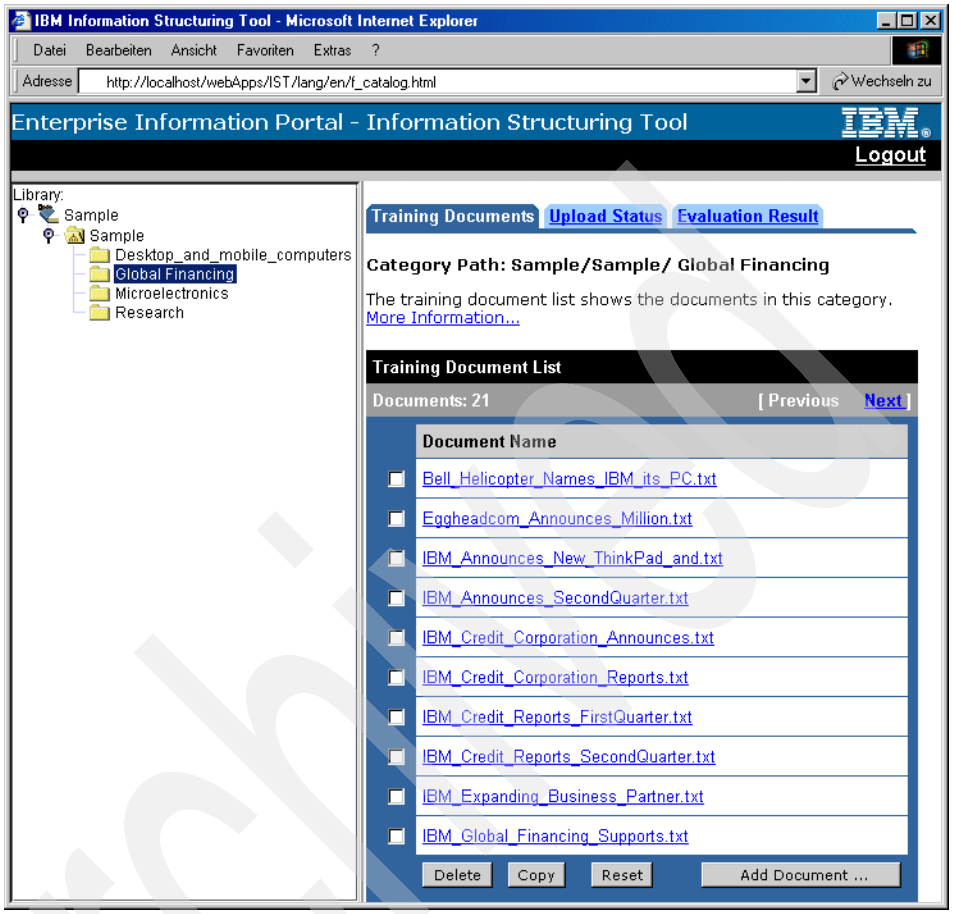
2. Assign typical text documents (training documents) to each category of the taxonomy.
3. Train the taxonomy to create a categorization model, which is then used by the categorizer to automatically determine the category of new documents.

You use the Information Structuring Tool (IST) of the Information Mining Service to define and train a taxonomy. The tool allows you to manually create each category and assign documents to it. It also provides the ability to upload a directory where each directory becomes a category and the documents in each directory are the training documents for the respective category.

Tip: Before continuing, you should make sure the DB2EXT service has been started. To start it, open a command window and run **`db2text start`**.

To create and train the taxonomy for the sample:

1. Open your browser, enter `http://localhost/webApps/IST/login.html`, and log on with the user name icmadmin.
2. In the left-hand frame, right-click **Library** and select **New catalog**. Name this new catalog “Sample”. The catalog will hold the taxonomy and the categorization model once the taxonomy has been trained. The root category of the taxonomy is already created and is also named Sample.
3. Expand the tree for catalog Sample (displayed with a book icon). Select the root category **Sample** of this catalog, and an empty Training Document List is displayed.
4. To create some categories and appropriate training documents from a directory, select **Add Document**, click **Browse**, go to directory C:\cmbroot\ikf\firststeps and select the directory **IBM Press Releases**. Click **Open**, then click **Submit** to begin the file upload process.
5. When the upload process is finished, close the window. Expand the root category Sample to see the whole taxonomy in the left-hand pane of your IST window. If you click a category, the assigned training documents will be listed as shown in Figure 6-30 on page 157.


IBM Information Structuring Tool - Microsoft Internet Explorer

Datei Bearbeiten Ansicht Favoriten Extras ?

Adresse http://localhost/webApps/IST/lang/en/f_catalog.html

Enterprise Information Portal - Information Structuring Tool

Logout

Library:
- Sample
  - Sample
    - Desktop_and_mobile_computers
    - Global Financing
    - Microelectronics
    - Research

Training Documents | Upload Status | Evaluation Result

Category Path: Sample/Sample/ Global Financing

The training document list shows the documents in this category.
More Information...

Training Document List

Documents: 21 [Previous Next]

	Document Name
☐	Bell_Helicopter_Names_IBM_its_PC.txt
☐	Eggheadcom_Announces_Million.txt
☐	IBM_Announces_New_ThinkPad_and.txt
☐	IBM_Announces_SecondQuarter.txt
☐	IBM_Credit_Corporation_Announces.txt
☐	IBM_Credit_Corporation_Reports.txt
☐	IBM_Credit_Reports_FirstQuarter.txt
☐	IBM_Credit_Reports_SecondQuarter.txt
☐	IBM_Expanding_Business_Partner.txt
☐	IBM_Global_Financing_Supports.txt

Delete Copy Reset Add Document ...

Figure 6-30 Information Structuring Tool training documents

6. Select the catalog **Sample** (displayed with a book icon), click the **Training** tab and select **Start Training**.
7. The message “`The catalog is up-to-date, no training is required`” shows that the training is finished.

Based on the training documents that are assigned to each category, a categorization model has been created that can now be used to analyze an arbitrary text and assign one of the four categories as shown in Figure 6-30.

To analyze all articles of the sample database IBMPRESS, we perform a federated search using the search template SearchLongBySource defined in database EIPSAMPL. We then conduct the text analysis on each search result item and simply print the results.

The sample Categorization.java can be found in the samples directory for the Information Mining Service: C:\cmbroot\samples\java\beans\infomining\categorization.

The sample application is implemented using the JavaBeans API. Figure 6-31 shows the beans and how they are connected to define the required event flow.

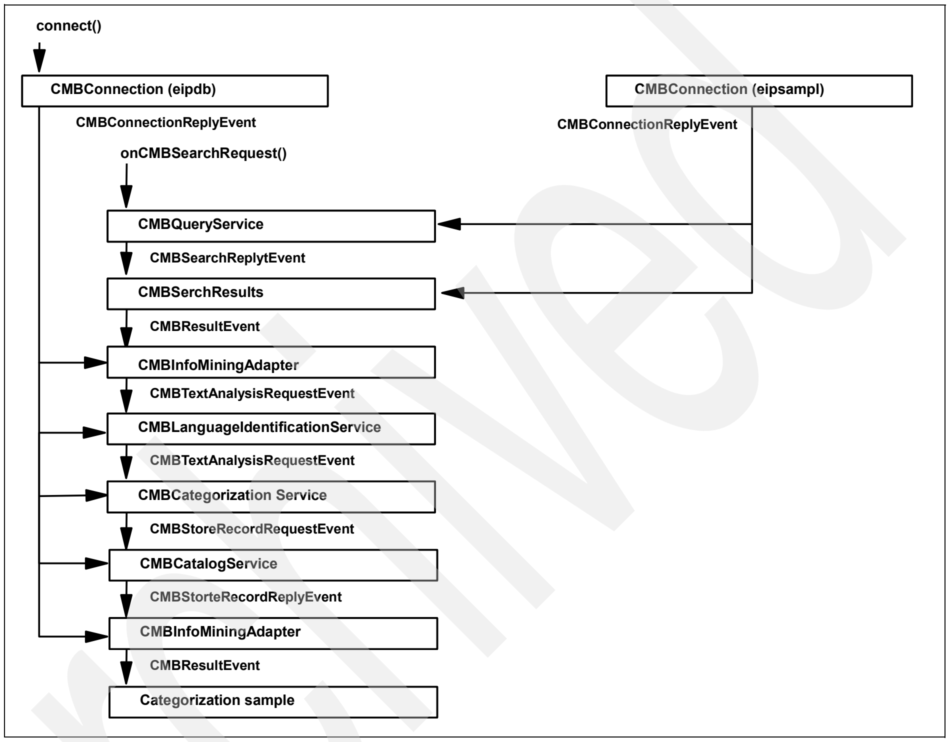


Figure 6-31 Categorization sample JavaBean event flow

The query bean and search results bean need a connection to the EIPSAMPL database because this is where the federated entity and the search template SearchLongBySource is defined. The beans of the Information Mining Service need to get connected to the EIP administration database. The categorization beans, for example, needs to look up the categorization model from this database.

The search results bean populates the result list and fires a result event, which contains the result items, to the adapter bean, which retrieves the items and extracts the text from the item parts. The adapter then initiates a text analysis request against the language identification bean. Language identification is a required analysis before you can run categorization because categorization is language specific. The categorization bean reads the taxonomy and categorization model from the catalog we just configured and uses this information to determine the category for each item. The category and the language get stored in the item but it is transient only.

The categorization bean then invokes the catalog bean to store the analysis results in the EIP administration database. This is optional and is only required if you do not want to store the results on a content server.

Finally the adapter bean converts the reply event back to a search result event, where each contained item is now enriched with the category and the language. The sample prints both results together with the Persistent Identifier of the item.

The event flow for the text analysis is defined with the following code snippet:

```
queryService.addCMBSearchReplyListener(searchResults);
searchResults.addCMBResultListener(adapter1);
adapter1.addCMBTextAnalysisRequestListener(languageIdentificationService);
languageIdentificationService.addCMBTextAnalysisRequestListener(categorizat
ionService);
categorizationService.addCMBStoreRecordRequestListener(catalogService);
catalogService.addCMBStoreRecordReplyListener(adapter2);
adapter2.addCMBResultListener(this);
```

The search template can be retrieved by name (SearchLongBySource). A value needs to be specified for the search criteria (source). To start the search, a search request is created with the template and fired against the query bean:

```
CMBSearchTemplate searchTemplate =
schema.getSearchTemplate(SEARCH_TEMPLATE);
String[] searchValues = { SEARCH_VALUE };
searchTemplate.setSearchCriterion(SEARCH_CRITERION,
CMBBaseConstant.CMB_OP_EQUAL, searchValues);
CMBSearchRequestEvent searchRequest = new CMBSearchRequestEvent(this,
CMBSearchRequestEvent.CMB_REQUEST_SEARCH_SYNCH, searchTemplate);
queryService.onCMBSearchRequest(searchRequest);
```

The category information of a result item can be obtained from the record object contained in the item as shown in the following code snippet.

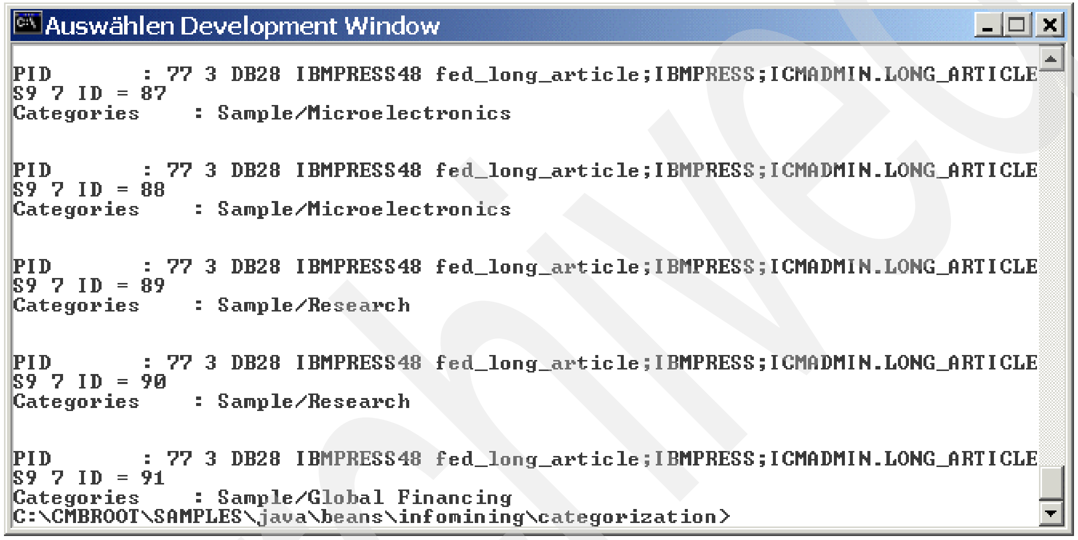
```
CMBRecord currentRecord = currentItem.getInfoMiningRecord();
System.out.println("\n\nPID        : " + currentRecord.getPID());
System.out.println("Categories    : " +
currentRecord.getValue("IKF_CATEGORIES"));
```

To compile and run the categorization sample:

1. Open the EIP development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**.

2. Change to the sample directory C:\cmbroot\samples\java\beans\infomining\categorization.
3. To compile the sample, enter **`compile`**.
4. Enter **`run`** to start the sample. You are prompted for the database names, user IDs and the catalog name. You should be able to go with the defaults except for the EIP database. Enter `EIPDB`. Use the same DB name - case sensitive - as in IST web.xml.

In the search result list, the Persistent Identifier of each item is displayed along with the category information for this item as shown in Figure 6-32.



```
PID        : 77 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLE
S9 7 ID = 87
Categories    : Sample/Microelectronics


PID        : 77 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLE
S9 7 ID = 88
Categories    : Sample/Microelectronics


PID        : 77 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLE
S9 7 ID = 89
Categories    : Sample/Research


PID        : 77 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLE
S9 7 ID = 90
Categories    : Sample/Research


PID        : 77 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLE
S9 7 ID = 91
Categories    : Sample/Global Financing
C:\CMBROOT\SAMPLES\java\beans\infomining\categorization>
```

Figure 6-32 Categorization sample output

Adding summarization

Adding a new analysis step to the application, such as summarization, means you need to add a new bean, the summarization bean, to the event flow as shown in Figure 6-33 on page 161.

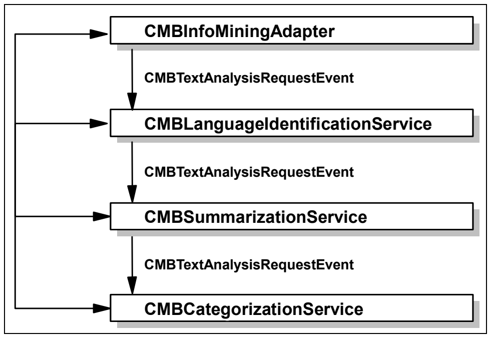

CMBInfoMiningAdapter
CMBTextAnalysisRequestEvent
CMBLanguageIdentificationService
CMBTextAnalysisRequestEvent
CMBSummarizationService
CMBTextAnalysisRequestEvent
CMBCategorizationService

Figure 6-33 Adding summarization to the beans event flow

The implementation involves editing the Categorization.java file as follows:

1. Add a new import statement for the summarizer bean:

   ```
   import com.ibm.mm.beans.infomining.CMBSummarizationService;
   ```

2. In the constructor, create an object reference and an object:

   ```
   CMBSummarizationService summarizationService = new
   CMBSummarizationService();
   ```

3. Add the summarizer bean to the connection flow:

   ```
   eipConnection.addCMBConnectionReplyListener(summarizationService);
   ```

4. Currently the language identification bean is configured to send analysis request events to the categorization bean. Change the appropriate line so that the event is now sent to the summarization bean. Add a new line for the summarization bean so that it sends request events to the categorization service. Both lines should now look as follows:

   ```
   languageIdentificationService.addCMBTextAnalysisRequestListener(summarizati
   onService);
   summarizationService.addCMBTextAnalysisRequestListener(categorizationServic
   e);
   ```

5. Add this line to make sure the summarization bean can send exception events:

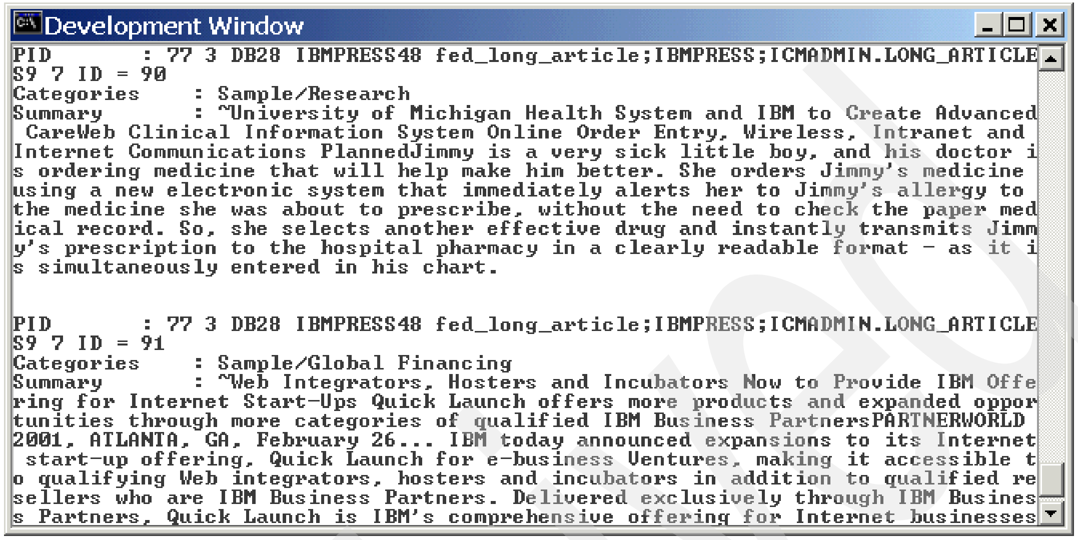
   ```
   summarizationService.addCMBExceptionListener(this);
   ```

6. In the onCMBResult method, add this line to print the summary for each item:

   ```
   System.out.println("Summary: " + currentRecord.getValue("IKF_SUMMARY"));
   ```

The complete sample is also provided with the redbook samples.

After you have recompiled and run the application again, the results should look similar to Figure 6-34. In addition to the category information, a summary is printed for each item.



```
PID        : 77 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLE
S9 7 ID = 90
Categories     : Sample/Research
Summary        : ~University of Michigan Health System and IBM to Create Advanced
 CareWeb Clinical Information System Online Order Entry, Wireless, Intranet and
Internet Communications PlannedJimmy is a very sick little boy, and his doctor i
s ordering medicine that will help make him better. She orders Jimmy's medicine
using a new electronic system that immediately alerts her to Jimmy's allergy to
the medicine she was about to prescribe, without the need to check the paper med
ical record. So, she selects another effective drug and instantly transmits Jimm
y's prescription to the hospital pharmacy in a clearly readable format - as it i
s simultaneously entered in his chart.


PID        : 77 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLE
S9 7 ID = 91
Categories     : Sample/Global Financing
Summary        : ~Web Integrators, Hosters and Incubators Now to Provide IBM Offe
ring for Internet Start-Ups Quick Launch offers more products and expanded oppor
tunities through more categories of qualified IBM Business PartnersPARTNERWORLD
2001, ATLANTA, GA, February 26... IBM today announced expansions to its Internet
 start-up offering, Quick Launch for e-business Ventures, making it accessible t
o qualifying Web integrators, hosters and incubators in addition to qualified re
sellers who are IBM Business Partners. Delivered exclusively through IBM Busines
s Partners, Quick Launch is IBM's comprehensive offering for Internet businesses
```

Figure 6-34 Changed categorization sample output

A scenario that uses the Information Mining Service to implement a category-based search in Content Manager with the eClient is shown in 13.1, “Using categories and summaries in eClient searches” on page 282. In 13.2, “Creating categories and summaries during document import” on page 296, we demonstrate how to create an application with the Information Mining Service Java API instead of JavaBeans.

6.2.7 Working with controller servlet

The following section shows how to install, use and extend the controller servlet.

Installing the servlet

To install the servlet, complete the steps in the Information Center. Select **Enterprise Information Portal -> Planning and Install -> Configuring Enterprise Information Portal components -> Configuring the Web Application Server for the EIP tag library and servlet**. (Please read the comments below first.)

It describes how to build the Web application archive (eip.war) file and the enterprise archive (eip.ear) file. Furthermore, it describes how to install the application in WebSphere Application Server.

Important:

1. Other than described in the Information Center, start the Application Assembly Tool by selecting **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Application Assembly Tool**.

- To avoid duplicates in the CLASSPATH and to speed up the installation process in WebSphere Application Server, we recommend that you not add all the JAR files listed the information Center when adding the JAR files to the WAR file. You just need:
  - cmbservlets81.jar
  - cmbtag81.jar

2. When adding the tag library, you cannot select **tld** as described because there is no such directory. Select the file **taglib.tld** instead.
3. When defining an alias for the tag library, specify `/taglib.tld` (with a leading slash) for the tag library location.
4. To avoid configuration problems, we also recommend that you install the servlet on the eClient server (eClient_Server) rather than on an arbitrary server as stated in the Information Center. The eClient server has most of the required environment settings (classpath and native library path) to run the servlet.
5. You need to follow the additional configuration steps below before continuing.

To install the application on the eClient server, make sure you specify the right server (eClient_Server) in step 3 of the installation process in the WebSphere Application Server Administrative Console. Restart the eClient server after installation. Also, do not forget to update the Web server plug-in after installation. To do that, select **Environment -> Update Web Server Plugin** in the WebSphere Application Server Administrative Console and click the **OK** button.

After installing the servlet, you need to complete the environment setup for the application server (eClient_Server) by extending its classpath and native library path:

1. Open the WebSphere Application Server Administrative Console and click **Environment -> Shared Libraries**.
2. In the scope fields, specify the name of your node and eClient_Server as the server. Click **Apply**. Library eClientLib appears in the list. Click **eClientLib** to open the configuration window.

3. Add the following four lines to the classpath (assuming EIP is installed in C:\cmbroot):

   ```
   C:/CMBROOT/lib/cmbcm81.jar
   C:/CMBROOT/lib/essrv.jar
   C:/CMBROOT/ikf/lib/ikf.jar
   C:/CMBROOT/ikf/lib
   ```

4. Add the following line to the native library path (assuming EIP is installed in C:\cmbroot):

   ```
   C:/CMBROOT/ikf/bin
   ```

5. Click **Apply** and save the changes to the master configuration.
6. Restart the application server.

Preparing the EIP environment

To enable the EIP development console for servlet development, edit the file C:\cmbroot\cmbenv81.bat (assuming EIP is installed in C:\cmbroot):

1. Change the value of variable WAS_HOME according to your WebSphere Application Server installation.
2. Add the following line to the CLASSPATH definition:

   ```
   set CLASSPATH=%CLASSPATH%;%JARDIR%\cmbservlets81.jar
   ```

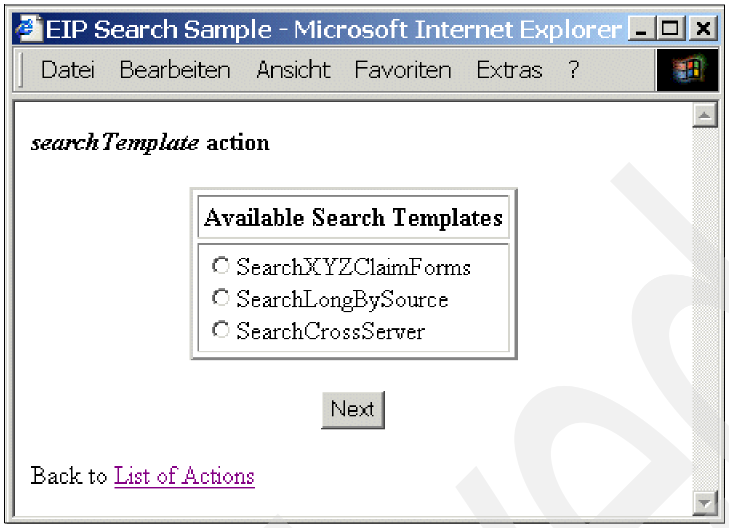
Running the servlet

To perform a federated search using the controller servlet, do the following:

1. Open your browser, enter `http://localhost/eip/jsp/main.html`, and follow the link to the servlet actions.
2. Click **Logon** and log on with:

   ```
   userid: icmadmin
   server: eipsampl
   serverType: Fed
   ```

3. After you have successfully logged on, go back to the list of servlet actions and click **Search template**. You get a list of available search templates as shown in Figure 6-35 on page 165. Select the template **SearchLongBySource** and click **Next**.



searchTemplate action

Available Search Templates

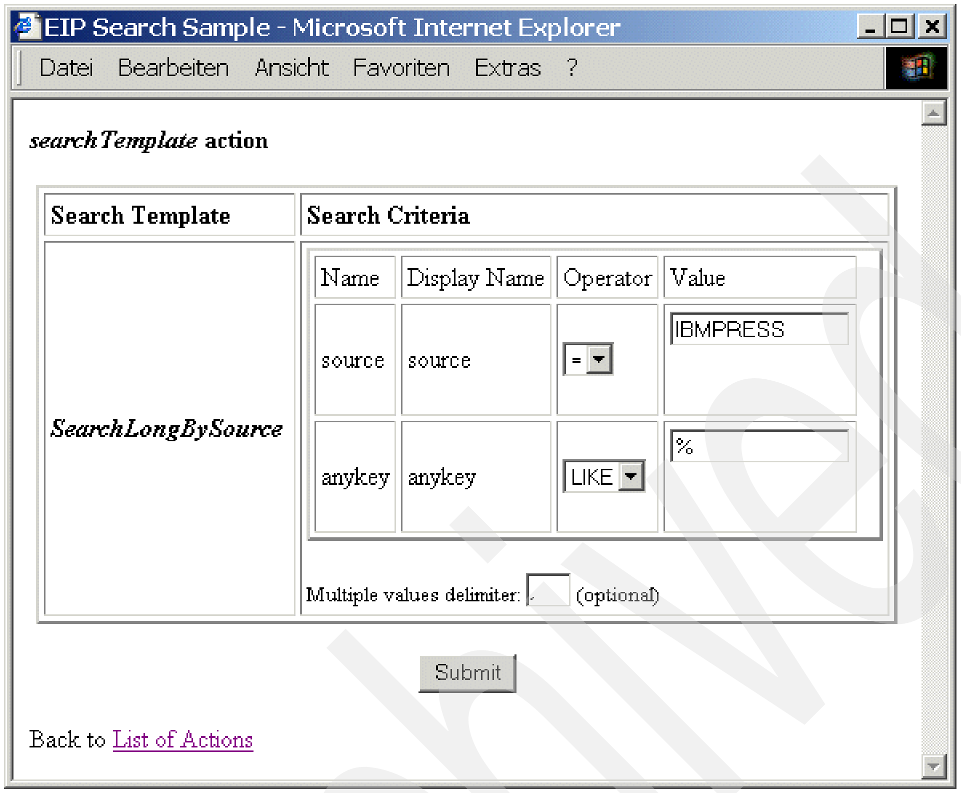
- ○ SearchXYZClaimForms
- ○ SearchLongBySource
- ○ SearchCrossServer

Next

Back to List of Actions

Figure 6-35 Search template list from controller servlet

4. In the search form, enter `ibmpress` (lowercase) as the search value for the criteria search and do not specify a search value for the criteria anykey. The search form should now look like Figure 6-36 on page 166. Click **Submit** to initiate the search.


EIP Search Sample - Microsoft Internet Explorer

searchTemplate action

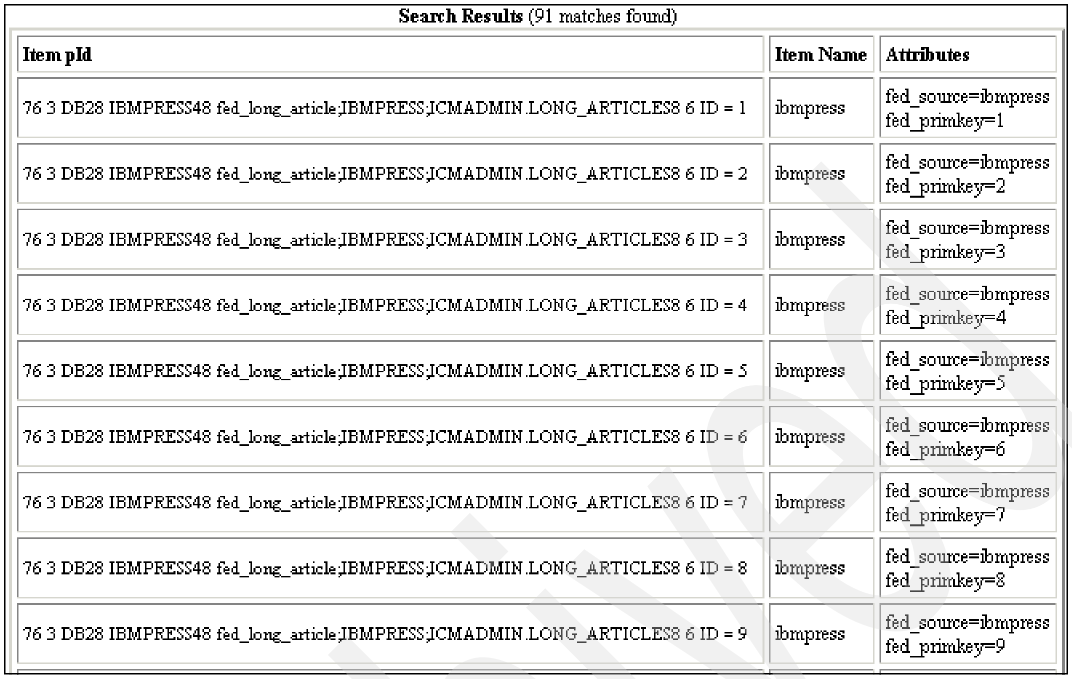
Search Template	Search Criteria			
	Name	Display Name	Operator	Value
SearchLongBySource	source	source	=	IBMPRESS
	anykey	anykey	LIKE	%

Multiple values delimiter: , (optional)

Back to List of Actions

Figure 6-36 Search values for controller servlet

5. The search result list is shown in Figure 6-37 on page 167; however, it is not very useful yet. Beside the fact that you cannot click a document to view its content, it is also not possible to distinguish between the important and the unimportant documents.



Search Results (91 matches found)

Item pId	Item Name	Attributes
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 1	ibmpress	fed_source=ibmpress fed_primkey=1
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 2	ibmpress	fed_source=ibmpress fed_primkey=2
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 3	ibmpress	fed_source=ibmpress fed_primkey=3
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 4	ibmpress	fed_source=ibmpress fed_primkey=4
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 5	ibmpress	fed_source=ibmpress fed_primkey=5
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 6	ibmpress	fed_source=ibmpress fed_primkey=6
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 7	ibmpress	fed_source=ibmpress fed_primkey=7
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 8	ibmpress	fed_source=ibmpress fed_primkey=8
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 9	ibmpress	fed_source=ibmpress fed_primkey=9

Figure 6-37 Search results from controller servlet

We show how to create new servlet actions and adapt the JSPs to make the result list more useful in the following section.

Creating new servlet actions

All actions known to the servlet are defined in a properties file named cmbservlet.properties. You can add, modify, or delete servlet actions by changing this file. To add a new action you need to:

- Implement a class to perform the action. The class must extend com.ibm.mm.servlets.CMBServletAction.
- Add the name of the class and the action name to the cmbservlet.properties file.

Class CMBServletAction declares one abstract method:

```
public abstract void perform(ServletContext app, HttpSession session,
HttpServletRequest request, HttpServletResponse res, Hashtable params)
throws...
```

Each new action needs to implement this method. The method parameters include the scope objects, the request and response object and a parameter object that contains all parameters specified when calling a servlet action.

In this example, we implement a new action that simply returns the whole textual content of a data object to the client. A second action creates and return a short summary of the textual content. We use the Information Mining Service of EIP to extract the textual content and create the summary. Because a data object is identified by its Persistent Identifier (PID), the new actions have a parameter for the PID. With a tiny change to the JSP that creates the search result page, we then add links to this page that invoke the new actions on the servlet.

The required steps are:

1. Implement two new servlet actions.
2. Register the new servlet actions in the cmbservlet.properties file.
3. Change the search result JSP to add links to the page.
4. Update the .ear file with the changed application files.

The servlet action for the content retrieval is implemented in NewRetrieveDocumentAction.java. The action that creates a summary is implemented in NewSummarizeDocumentAction.java.

The implementation in NewRetrieveDocumentAction.java starts with a reconnect because the implementation of the default logon action does not configure the connection bean to connect to the Information Mining Service. Since we do not want to implement a new logon action here, we simply reconnect. The connection bean has been stored in the session object where it can be retrieved by name. After disconnecting, we call the method setConnectToIKF to get a connection to the service during the next connect. The service connection can then be obtained using the getIKFService method:

```
connection = (CMBConnection)session.getAttribute("connection");
connection.disconnect();
connection.setConnectToIKF(true);
connection.connect();
...
DKIKFService ikfService = connection.getIKFService();
```

Since the action expects a parameter for the PID, we retrieve it from the parameter object. The PID is used to create an item object and retrieve its content from the server. The implementation assumes all items contain a content object:

```
CMBItem item = new CMBItem((String)params.get("documentPid"));
CMBDataManagement dataManagement =  connection.getDataManagement();
dataManagement.setDataObject(item);
dataManagement.retrieveItem();
CMBObject object = dataManagement.getContent(0);
```

The document filter of the Information Mining Service is used to extract the textual content from the content object. The filter supports all important document formats:

```
DKIKFDocumentFilter ikfFilter = new DKIKFDocumentFilter(ikfService);
DKIKFTextDocument ikfDocument =
ikfFilter.getTextDocument(object.getDataStream());
String content = ikfDocument.getContent();
```

Finally we send an HTML page with the content to the client. According to the Model-View-Controller design, you would call a JSP to create the response page for the client. To simplify the sample, we create the page directly:

```
PrintWriter out = res.getWriter();
out.println("<HTML><BODY><table border=1 width=400>");
out.println("<tr><td>" + content + "</td></tr>");
out.println("</table></BODY></HTML>");
```

The implementation for the second action (NewSummarizeDocumentAction.java) looks similar; but it performs some text analysis on the extracted content before returning to the client. First, the language identifier is called to determine the language of the text. This information is required by the summarizer, which is then called to create a summary:

```
DKIKFLanguageIdentificationResult langResult[] =
ikfLangIdentifier.analyze(ikfDocument);
...
String language = langResult[0].getLanguage();
ikfDocument.setLanguage(language);
summary = summarizer.analyze(ikfDocument).getSummary();
```

This time, the page returned to the client contains the summary:

```
out.println("<tr><td>" + summary + "</td></tr>");
```

To compile the new servlet actions:

1. Open an EIP development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**.
2. Change to the directory where you stored the redbook samples.
3. Run the two commands:

```
javac NewRetrieveDocumentAction.java
javac NewSummarizeDocumentAction.java.
```

To register the new actions in the cmbservlet.properties file:

1. In the EIP development window, change to the directory where you stored the redbook samples.

2. To get the current file, run:

   ```
   jar xvf %CMBROOT%\lib\cmbservlets81.jar
   com\ibm\mm\servlets\cmbservlet.properties
   ```

   The file can be found in the directory com, which is created in the current directory.

3. Edit the file and add the following lines to the action mapping list:

   ```
   action.summarizeDocument.class=NewSummarizeDocumentAction
   action.retrieveDocument.class=NewRetrieveDocumentAction
   ```

4. Save and close the file.

In the following steps, we describe the changes made in the JSP file templateresults.jsp, which creates the search results window. You do not need to complete these steps because the changed JSP file is provided with the redbook samples for this chapter. As an exercise, you can compare it to the original version that is located in directory %CMBROOT%\SAMPLES\jsp\servlets.

The following has been changed in the JSP file:

1. In the table definition (<table ...> ... </table>) for the results on the bottom of the file, after line:

   ```
   <td><b>Attributes</b></td>
   ```

   we inserted the line:

   ```
   <td><b>Summary</b></td>
   ```

   This is the header of a new column in the table.

2. The following line:

   ```
   <td><font size=-1><%= item.getPidString() %></font></td>
   ```

   has been replaced with:

   ```
   <td><font size=-1><a
   href="CMBControlServlet?action=retrieveDocument&documentPid=<%=java.net.URL
   Encoder.encode(item.getPidString())%>"><%=item.getPidString()%></a></font><
   /td>
   ```

   The created link calls the controller servlet and specifies two arguments, the action (retrieveDocument) and the PID.

3. After the last occurrence of the tag:

   ```
   </td>
   ```

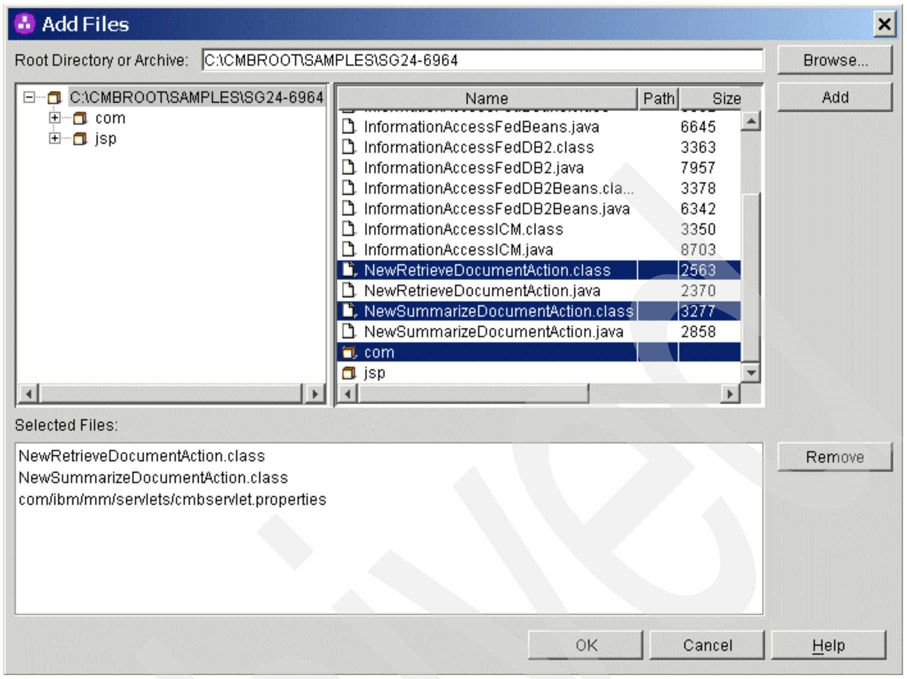
   we inserted the line:

   ```
   <td><a
   href="CMBControlServlet?action=summarizeDocument&documentPid=<%=java.net.UR
   LEncoder.encode(item.getPidString())%>">click to summarize</a></td>
   ```

This creates a new column in the table where each row contains a link. The created link calls the controller servlet and specifies two arguments, the action (summarizeDocument) and the PID.

Finally we need to update the eip.ear file:

1. Open the ear file using the WebSphere Application Server Application Assembly Tool and expand **Web Modules -> eip.war -> Files**. Class files, JAR files and resource files appear.
2. Right-click **Class Files** and select **Add Files**. The Add Files window appears.
3. Click **Browse**. Select the directory that contains the redbook samples (must appear in the File name field) and click **Select**.
4. From the upper-right box in the Add Files window, select the two class files files of the new servlet actions and the com directory containing the cmbservlet.properties file (hold down the Ctrl key):
   - NewRetrieveDocumentAction.class
   - NewSummarizeDocumentAction.class
   - com
5. Click **Add**. The Add Files window should now look like Figure 6-38 on page 172. Click **OK** and select **Yes** on the confirmation window to update the cmbservlet.properties file.



```
C:\CMBROOT\SAMPLES\SG24-6964
   com
   jsp
```

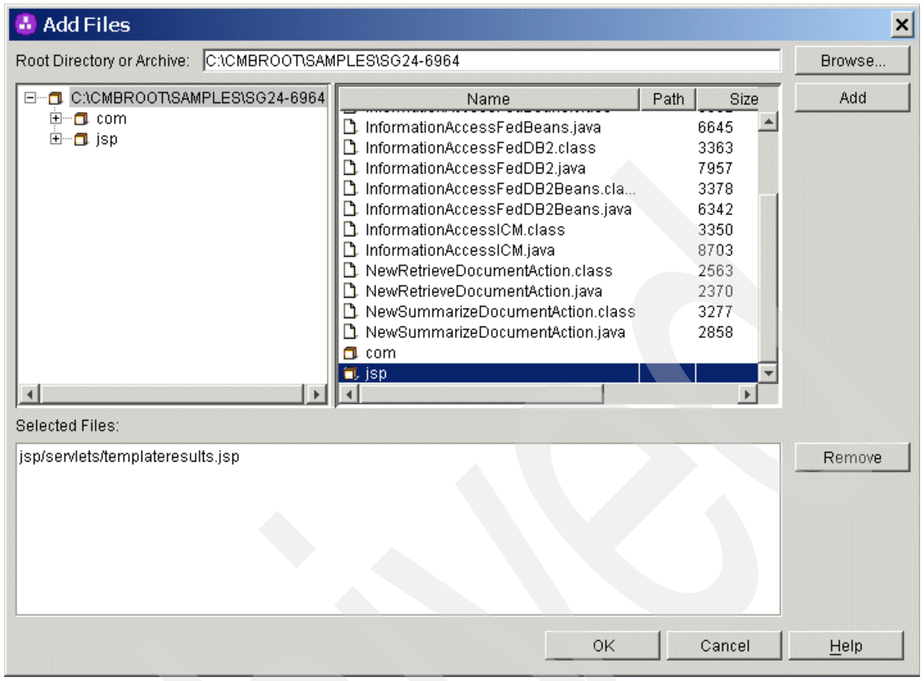
Name	Path	Size
InformationAccessFedBeans.java		6645
InformationAccessFedDB2.class		3363
InformationAccessFedDB2.java		7957
InformationAccessFedDB2Beans.cla...		3378
InformationAccessFedDB2Beans.java		6342
InformationAccessICM.class		3350
InformationAccessICM.java		8703
NewRetrieveDocumentAction.class		2563
NewRetrieveDocumentAction.java		2370
NewSummarizeDocumentAction.class		3277
NewSummarizeDocumentAction.java		2858
com		
jsp		

Selected Files:

```
NewRetrieveDocumentAction.class
NewSummarizeDocumentAction.class
com/ibm/mm/servlets/cmbservlet.properties
```

Figure 6-38 Adding new servlet actions to the WAR file

6. Right-click **Resource Files** and select **Add Files**. The Add Files window appears.
7. Click **Browse**. Select the directory that contains the redbook samples (must appear in the File name field) and click **Select**.
8. From the upper-right box in the Add Files window, select the **jsp** directory that contains the new JSP file.
9. Click **Add**. The Add Files window should now look like Figure 6-39 on page 173. Click **OK** and select **Yes** on the confirmation window to update the file.


Add Files

Root Directory or Archive: C:\CMBROOT\SAMPLES\SG24-6964

Name	Path	Size
InformationAccessFedBeans.java		6645
InformationAccessFedDB2.class		3363
InformationAccessFedDB2.java		7957
InformationAccessFedDB2Beans.cla...		3378
InformationAccessFedDB2Beans.java		6342
InformationAccessICM.class		3350
InformationAccessICM.java		8703
NewRetrieveDocumentAction.class		2563
NewRetrieveDocumentAction.java		2370
NewSummarizeDocumentAction.class		3277
NewSummarizeDocumentAction.java		2858
com		
jsp		

Selected Files:

jsp/servlets/templateresults.jsp

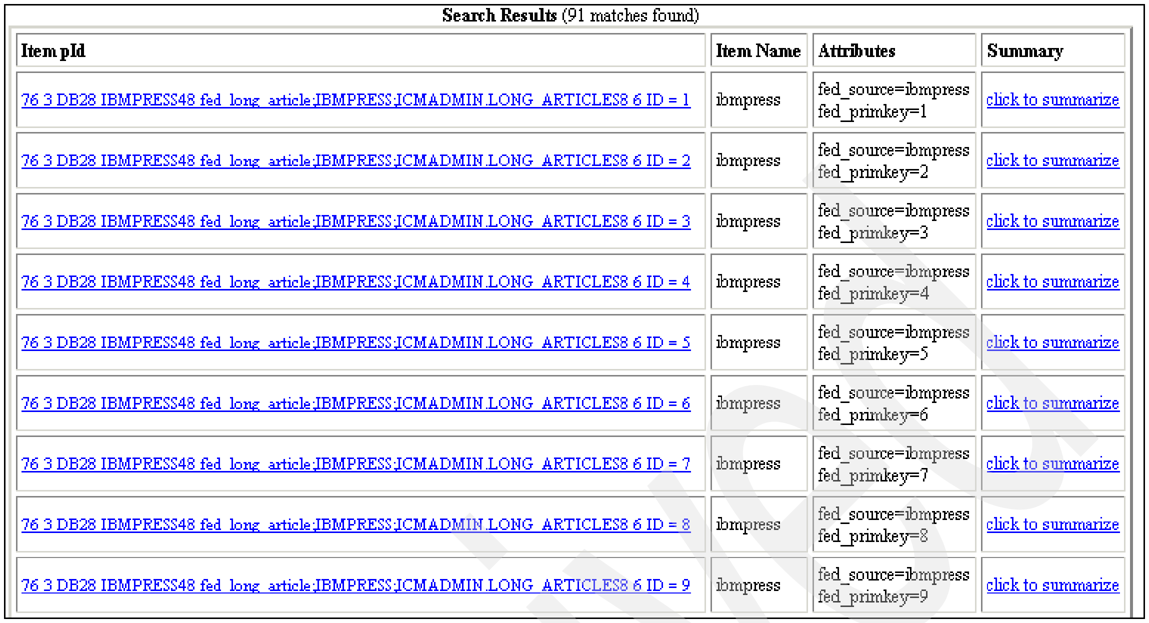
Figure 6-39 Adding resource files to the war file

10.Select **File -> Save** to save the EAR file and then select **File -> Close**.

Use the WebSphere Application Server Administrative Console to uninstall the old servlet and install the new version.

Now run the search template search again. Do you remember the previous result list?

The new search result window looks like Figure 6-40 on page 174.

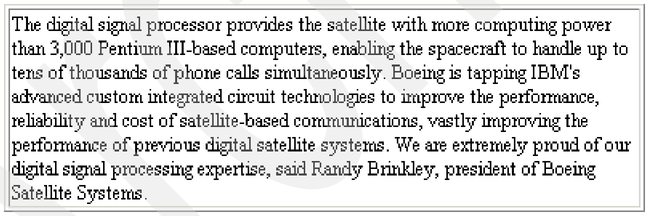


Search Results (91 matches found)

Item pId	Item Name	Attributes	Summary
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 1	ibmpress	fed_source=ibmpress fed_primkey=1	click to summarize
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 2	ibmpress	fed_source=ibmpress fed_primkey=2	click to summarize
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 3	ibmpress	fed_source=ibmpress fed_primkey=3	click to summarize
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 4	ibmpress	fed_source=ibmpress fed_primkey=4	click to summarize
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 5	ibmpress	fed_source=ibmpress fed_primkey=5	click to summarize
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 6	ibmpress	fed_source=ibmpress fed_primkey=6	click to summarize
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 7	ibmpress	fed_source=ibmpress fed_primkey=7	click to summarize
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 8	ibmpress	fed_source=ibmpress fed_primkey=8	click to summarize
76 3 DB28 IBMPRESS48 fed_long_article;IBMPRESS;ICMADMIN.LONG_ARTICLES8 6 ID = 9	ibmpress	fed_source=ibmpress fed_primkey=9	click to summarize

Figure 6-40 Enhanced search result of controller servlet

If you click the links in the new column on the right, a summary as shown in Figure 6-41 is displayed for the appropriate result. This should give you a quick idea of the content of the document.



The digital signal processor provides the satellite with more computing power than 3,000 Pentium III-based computers, enabling the spacecraft to handle up to tens of thousands of phone calls simultaneously. Boeing is tapping IBM's advanced custom integrated circuit technologies to improve the performance, reliability and cost of satellite-based communications, vastly improving the performance of previous digital satellite systems. We are extremely proud of our digital signal processing expertise, said Randy Brinkley, president of Boeing Satellite Systems.

Figure 6-41 Summary returned by new servlet action

If the summary of the document seems to be interesting, you can go back to the result list and click the Persistent Identifier to read the complete text as shown in Figure 6-42 on page 175.

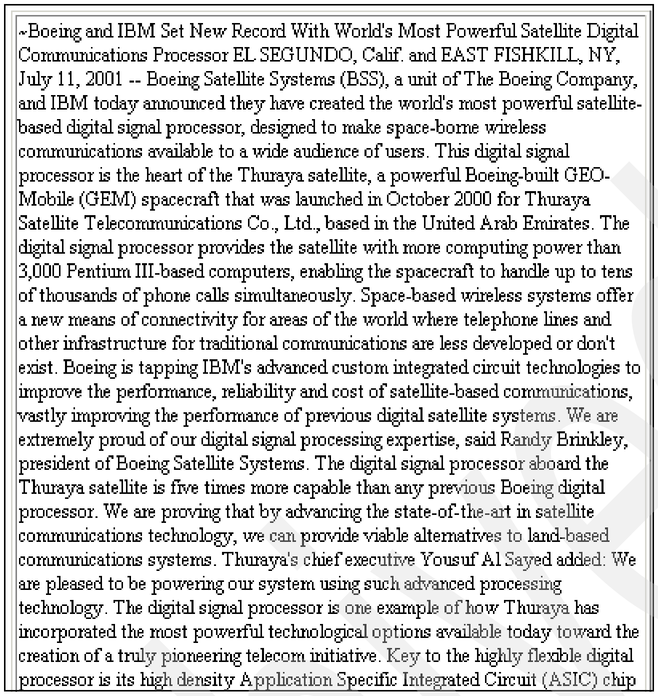

~Boeing and IBM Set New Record With World's Most Powerful Satellite Digital Communications Processor EL SEGUNDO, Calif. and EAST FISHKILL, NY, July 11, 2001 -- Boeing Satellite Systems (BSS), a unit of The Boeing Company, and IBM today announced they have created the world's most powerful satellite-based digital signal processor, designed to make space-borne wireless communications available to a wide audience of users. This digital signal processor is the heart of the Thuraya satellite, a powerful Boeing-built GEO-Mobile (GEM) spacecraft that was launched in October 2000 for Thuraya Satellite Telecommunications Co., Ltd., based in the United Arab Emirates. The digital signal processor provides the satellite with more computing power than 3,000 Pentium III-based computers, enabling the spacecraft to handle up to tens of thousands of phone calls simultaneously. Space-based wireless systems offer a new means of connectivity for areas of the world where telephone lines and other infrastructure for traditional communications are less developed or don't exist. Boeing is tapping IBM's advanced custom integrated circuit technologies to improve the performance, reliability and cost of satellite-based communications, vastly improving the performance of previous digital satellite systems. We are extremely proud of our digital signal processing expertise, said Randy Brinkley, president of Boeing Satellite Systems. The digital signal processor aboard the Thuraya satellite is five times more capable than any previous Boeing digital processor. We are proving that by advancing the state-of-the-art in satellite communications technology, we can provide viable alternatives to land-based communications systems. Thuraya's chief executive Yousuf Al Sayed added: We are pleased to be powering our system using such advanced processing technology. The digital signal processor is one example of how Thuraya has incorporated the most powerful technological options available today toward the creation of a truly pioneering telecom initiative. Key to the highly flexible digital processor is its high density Application Specific Integrated Circuit (ASIC) chip

Figure 6-42 Document content returned by new servlet action

The next step optionally is to precalculate the summary and store it together with the original object in the content server. This would allow a fast lookup of all summaries and you could create a result window that has the summaries already included. We discuss how to implement this scenario using Content Manager and the eClient in 13.1, “Using categories and summaries in eClient searches” on page 282.

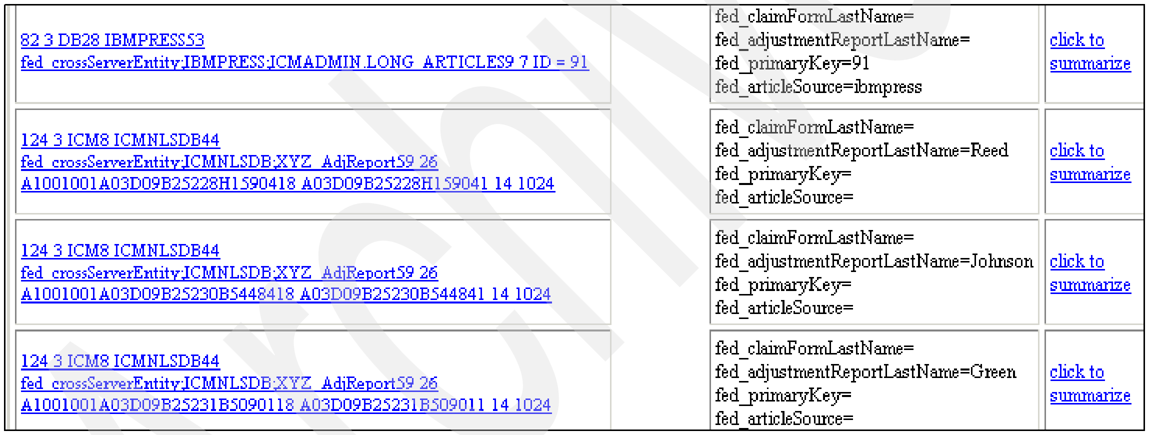
If you run the sample in 6.2.5, “Using federated connector to search across content servers” on page 143, you can now run the same search with the controller servlet. Just select the search template **SearchCrossServer**. Again you get the results from both content servers, Content Manager and DB2. You can use the new links to get a summary and the content of the press articles that come from DB2. You cannot see the content of the documents that come from Content Manager, because the documents in Content Manager are images and the document filter used to extract the textual content cannot perform optical character recognition (OCR). You need to add additional viewing capabilities to see the content. We add that additional viewing capability in 6.2.8, “Working with viewer toolkit” on page 176.

6.2.8 Working with viewer toolkit

In this sample, we show how to implement server-side document conversion to allow viewing of documents that are not directly supported by the browser. It demonstrates the usage of the document services bean, which is part of the EIP viewer toolkit. An overview of the toolkit can be found in 6.1, “Programming interface overview” on page 124.

This sample requires the extensions to the controller servlet described in 6.2.7, “Working with controller servlet” on page 162. Please make sure the Web application is working before continuing. It also requires the search template SearchCrossServer created in 6.2.5, “Using federated connector to search across content servers” on page 143.

Using the controller servlet and searching with search template SearchCrossServer, you get results from DB2 and Content Manager as shown in Figure 6-43. You can find the results that come from Content Manager by looking at the displayed identifier. It contains the database name ICMNLSDB.



82 3 DB28 IBMPRESS53
fed_crossServerEntity:IBMPRESS:ICMADMIN.LONG_ARTICLES9 7 ID = 91
	fed_claimFormLastName=
	fed_adjustmentReportLastName=
	fed_primaryKey=91
	fed_articleSource=ibmpress
		click to summarize

124 3 ICM8 ICMNLSDB44 fed_crossServerEntity:ICMNLSDB:XYZ_AdjReport59 26 A1001001A03D09B25228H1590418 A03D09B25228H159041 14 1024	fed_claimFormLastName= fed_adjustmentReportLastName=Reed fed_primaryKey= fed_articleSource=	click to summarize
124 3 ICM8 ICMNLSDB44 fed_crossServerEntity:ICMNLSDB:XYZ_AdjReport59 26 A1001001A03D09B25230B5448418 A03D09B25230B544841 14 1024	fed_claimFormLastName= fed_adjustmentReportLastName=Johnson fed_primaryKey= fed_articleSource=	click to summarize
124 3 ICM8 ICMNLSDB44 fed_crossServerEntity:ICMNLSDB:XYZ_AdjReport59 26 A1001001A03D09B25231B5090118 A03D09B25231B509011 14 1024	fed_claimFormLastName= fed_adjustmentReportLastName=Green fed_primaryKey= fed_articleSource=	click to summarize

Figure 6-43 Search results from Content Manager and DB2

When you click the identifier, you get an empty window, such as shown in Figure 6-44 on page 177; this is because the actual document is a TIFF image and the document filter used to extract the textual content does not conduct optical character recognition (OCR).

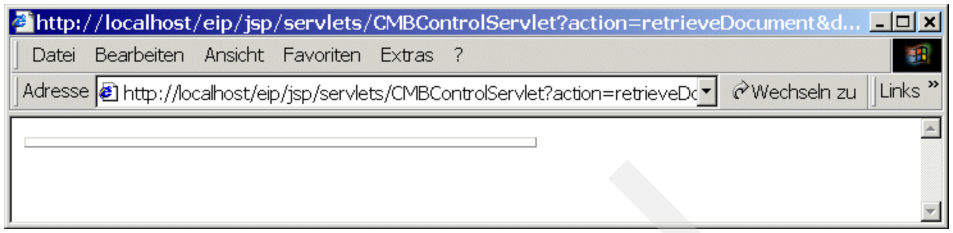

http://localhost/eip/jsp/servlets/CMBControlServlet?action=retrieveDocument&d...
Datei Bearbeiten Ansicht Favoriten Extras ?
Adresse http://localhost/eip/jsp/servlets/CMBControlServlet?action=retrieveDc
Wechseln zu
Links

Figure 6-44 Empty content window for Content Manager document

We update the servlet action NewRetrieveDocumentAction, which is responsible for returning the document content so that it uses the document services bean of the viewer toolkit to return the content in a format the browser can display.

The updated servlet action is provided with the redbook samples. Open the file NewRetrieveDocumentAction.java in the viewer directory.

The implementation has a special behavior if the retrieved data object is a TIFF image. In this case, the document services bean is created and the data object is loaded into a viewer document:

```
if(object.getMimeType().equals("image/tiff")) {
CMBDocumentServices documentServices = new CMBDocumentServices();
documentServices.setDataManagement(connection.getDataManagement());
CMBDocument document = documentServices.loadDocument(item);
```

The servlet output stream, which is retrieved from the response object, is required to send binary data to the client. The MIME type needs to be specified so that the browser knows how to display the binary data:

```
ServletOutputStream outStream = res.getOutputStream();
res.setStatus(HttpServletResponse.SC_OK);
res.setContentType(document.getWriteMimeType());
```

Finally, the rendered document is sent to the client. If the document has pages, we return just the first page to keep the sample small:

```
if(document.getCanPaginate()) {
CMBPage page = document.getPages(0);
page.write(outStream);
}
else {
document.write(outStream);
}
```

For documents other than TIFF images, the implementation still extracts the textual content with the document filter provided by the Information Mining Service.

To compile the new action:

1. Open an EIP development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**.
2. Change to the viewer directory contained in your redbook samples directory.
3. Run:

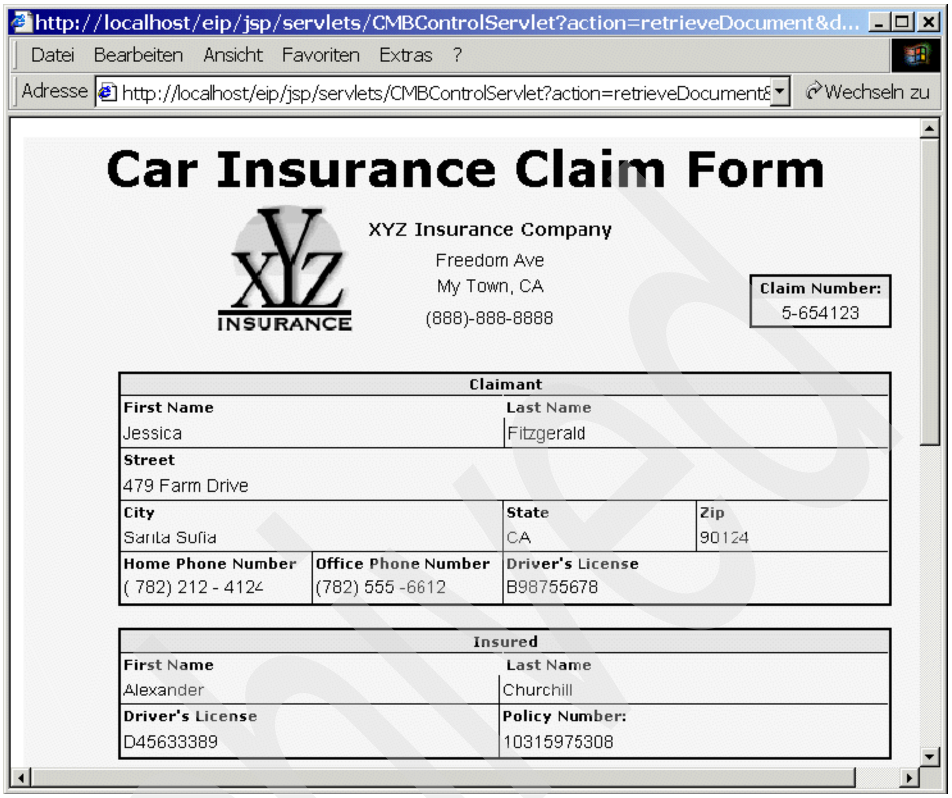
   ```
   javac NewRetrieveDocumentAction.java
   ```

To update the eip.ear file:

1. Open the EAR file using the WebSphere Application Server Application Assembly Tool and expand **Web Modules -> eip.war -> Files**. Class files, JAR files and resource files appear.
2. Right-click **Class Files** and select **Add Files**. The Add Files window appears.
3. Click **Browse**. Select the viewer directory in your redbook samples directory (viewer must appear in the File name field) and click **Select**.
4. From the upper-right box in the Add Files window, select **NewRetrieveDocumentAction.class** and then click **Add**.
5. Click **OK** and select **Yes** in the confirmation window.
6. Select **File -> Save** to save the EAR file and then select **File -> Close**.

Use the WebSphere Application Server Administrative Console to uninstall the old servlet and install the new version.

Now search again and click a Content Manager result. The document is displayed in the browser as shown in Figure 6-45 on page 179.


http://localhost/eip/jsp/servlets/CMBControlServlet?action=retrieveDocument&d...
Datei Bearbeiten Ansicht Favoriten Extras ?
Adresse http://localhost/eip/jsp/servlets/CMBControlServlet?action=retrieveDocument&
Wechseln zu

Car Insurance Claim Form

XYZ Insurance Company
Freedom Ave
My Town, CA
(888)-888-8888

Claim Number: 5-654123

Claimant

First Name: Jessica	Last Name: Fitzgerald	
Street: 479 Farm Drive		
City: Santa Sofia	State: CA	Zip: 90124
Home Phone Number: (782) 212 - 4124	Office Phone Number: (782) 555 -6612	Driver's License: B98755678

Insured

First Name: Alexander	Last Name: Churchill
Driver's License: D45633389	Policy Number: 10315975308

Figure 6-45 Rendered content for a Content Manager document

If you want to see more samples regarding the viewer toolkit, launch C:\cmbroot\samples\java\viewer\readme.html to get a list of samples that come with EIP. This includes an applet sample and stand-alone application.

7

Setting up an eClient development environment

This chapter discusses different options for an eClient development environment. It includes instructions on how to set up an eClient development environment, and provides pointers on how to use the WebSphere Studio Application Developer tools, including debugging your application.

This chapter covers the following topics:

- Development environment options
- Using simple editor and command-line utilities
- Configuring Studio Application Developer for use with eClient
- Deploying customized eClient to another system

7.1 Development environment options

There are many options for development tools that can be used to customize eClient. It is as simple as using a generic text editor or as complicated as using some J2EE software development tools. The following sections provide several options for a development environment. We include some of the reasons to choose one over another. A more detailed description of how to install, configure, and use some of the options is provided in the following sections of this chapter:

- Simple text editor and command-line tools
- IBM WebSphere Studio Application Developer
- Eclipse Software Development Kit

Simple text editor and command-line tools

If you are making a small number of simple changes, there is no need to install and configure a J2EE development system. You can simply modify the JSPs in the WebSphere directories with Notepad or your favorite editor. WebSphere will automatically recompile the JSP the next time the JSP is accessed. You can also use the WebSphere Application Assembly Tool to package your custom eClient. Many of the information mining examples in this redbook use this method. We recommend using text editor and command-line tools to get a quick start in development.

IBM WebSphere Studio Application Developer

IBM WebSphere Studio Application Developer Version 5.0 is the latest J2EE development platform from IBM. It is flexible and powerful. For serious software development, we recommend using WebSphere Studio Application Developer. The eClient customization examples for this redbook are developed using WebSphere Studio Application Developer, which can be downloaded (try or buy) from:

`http://www.ibm.com/software/awdtools/studioappdev`

Eclipse Software Development Kit

The Eclipse Software Development Kit (Eclipse SDK) Version 2.1 is an open source software development platform. The Eclipse organization was created by a consortium of software development tool providers comprised of IBM and other leading tool providers. Because it is free, we recommend using this if you do not want to invest in any serious development tools.

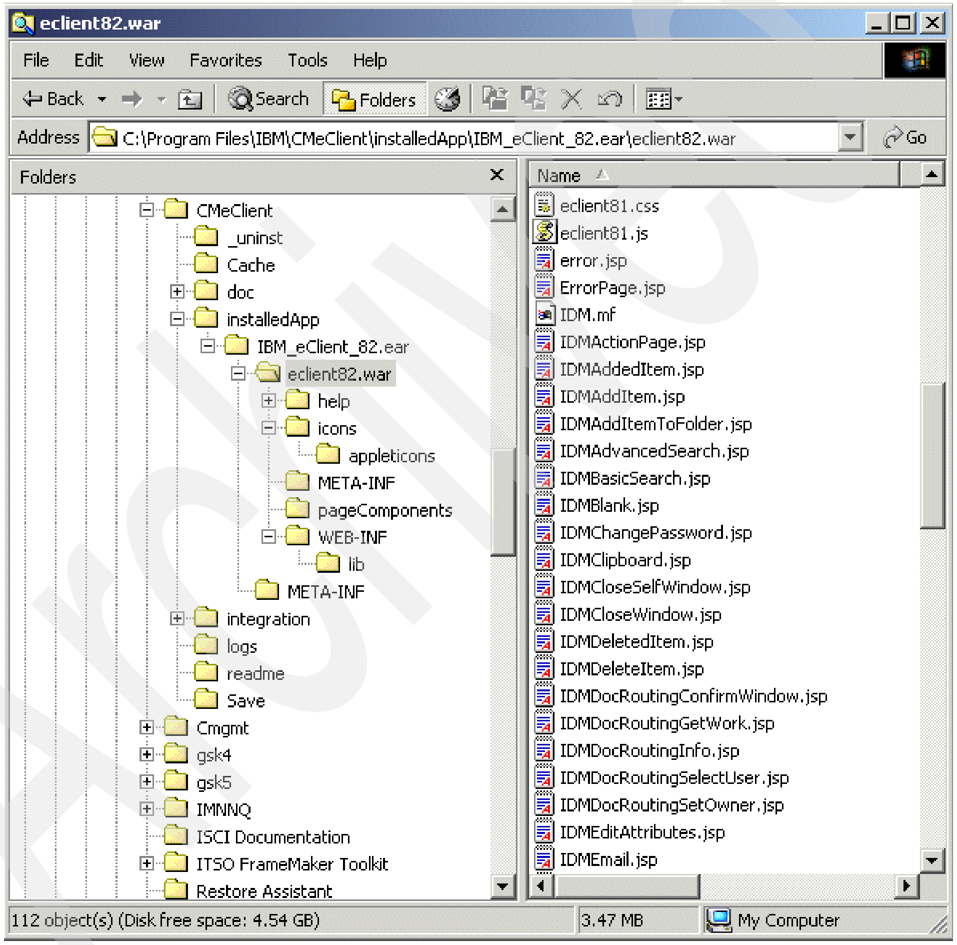
The Eclipse SDK Version 2.1 can be downloaded from:

`http://www.eclipse.org`

7.2 Using simple editor and command-line utilities

If you are making simple changes to the application, this can be done by simply using Notepad or your favorite source code editor to modify the JSP files in the eClient. The JSP files are located in two main directories in your WebSphere directory.

After installing the eClient, the JSPs and other components of the eClient are stored as shown in Figure 7-1.



Address C:\Program Files\IBM\CMeClient\installedApp\IBM_eClient_82.ear\eclient82.war

```
CMeClient
    _uninst
    Cache
    doc
    installedApp
        IBM_eClient_82.ear
            eclient82.war
                help
                icons
                    appleticons
                META-INF
                pageComponents
                WEB-INF
                    lib
            META-INF
    integration
    logs
    readme
    Save
Cmgmt
gsk4
gsk5
IMNNQ
ISCI Documentation
ITSO FrameMaker Toolkit
Restore Assistant
```

```
eclient81.css
eclient81.js
error.jsp
ErrorPage.jsp
IDM.mf
IDMActionPage.jsp
IDMAddedItem.jsp
IDMAddItem.jsp
IDMAddItemToFolder.jsp
IDMAdvancedSearch.jsp
IDMBasicSearch.jsp
IDMBlank.jsp
IDMChangePassword.jsp
IDMClipboard.jsp
IDMCloseSelfWindow.jsp
IDMCloseWindow.jsp
IDMDeletedItem.jsp
IDMDeleteItem.jsp
IDMDocRoutingConfirmWindow.jsp
IDMDocRoutingGetWork.jsp
IDMDocRoutingInfo.jsp
IDMDocRoutingSelectUser.jsp
IDMDocRoutingSetOwner.jsp
IDMEditAttributes.jsp
IDMEmail.jsp
```

112 object(s) (Disk free space: 4.54 GB) 3.47 MB My Computer

Figure 7-1 Standard location of eClient source files

You can edit these JSP files with an editor. The next time the JSP files are accessed by eClient, they are re-compiled on the fly by WebSphere.

If there are compilation errors, you get an eClient error page. You need to check the stdout file configured in WebSphere for the eClient application. The default location for this file is C:\Program Files\IBM\CMeClient\logs\eClient_Server_stdout.log.

To debug the Java code in your JSP, use the following to write debug messages to the stdout file:

```
System.out.println(“test”)
```

The Application Assembly Tool can be used if you want to add your own servlets to the eClient. This is described in Chapter 13, “Enabling metadata-based content retrieval” on page 281.

7.3 Configuring Studio Application Developer for use with eClient

This section describes how to set up your development environment using WebSphere Studio Application Developer.

You can download a trial version or buy a version of WebSphere Studio Application Developer Version 5.0 at:

```
http://www.ibm.com/software/awdtools/studioappdev
```

If you are not familiar with WebSphere Studio Application Developer, refer to the redbook, *WebSphere Studio Application Developer Version 5 Programming Guide*, SG24-6957 for detailed information. This redbook explains the architecture of WebSphere Studio Application Developer and how to set up your workbench and workspace preference. It covers WebSphere Studio Application Developer basic tooling and team environment along with the development and deployment of Web applications. This is where you can find out general information you need to know about programming and debugging your application in WebSphere Studio Application Developer.

System requirement

To use WebSphere Studio Application Developer, you need a workstation with at least 1 GB of RAM and preferably a Pentium IV with 1.8 GHz processor or better. While writing this redbook, we found that all of the software (DB2, Content Manager V8.2 Library Server and Resource Manager, EIP V8.2, eClient V8.2, WebSphere V5.0, and WebSphere Studio Application Developer V5.0) can run concurrently on an IBM ThinkPad® T30 (Pentium IV, 1.8 GHz with 1 GB of RAM). With this configuration, memory usage hovered between 1.8 GB and 2.5 GB. This configuration is usable but we recommend that you use more memory, if possible. If you are using a machine with 1 GB of RAM, be sure to set your virtual memory settings to provide a minimum of 1.0 GB of virtual memory and a maximum of 2.0 GB or more.

Important: If you install and run all products in one machine and you have only 1 GB of RAM, be sure to set the virtual memory to a minimum of 1.0 GB and a maximum of 2.0 GB or more.

While many WebSphere Studio Application Developer users install WebSphere on their development system in addition to WebSphere Studio Application Developer, the installation of WebSphere Application Server is not required because WebSphere Studio Application Developer includes the installation of a test environment that includes an HTTP server. In our scenario, we have standard WebSphere Application Server V5.0 installed in addition to WebSphere Studio Application Developer.

Installation notes

The following are some important notes on the installation of EIP and WebSphere Studio Application Developer:

- For EIP installation on a development workstation:

  In order to customize eClient, you must install the EIP development kit. This option is available on one of the first pages in the standard EIP setup.exe application.

- For WebSphere Studio Application Developer V5.0 installation on your workstation:

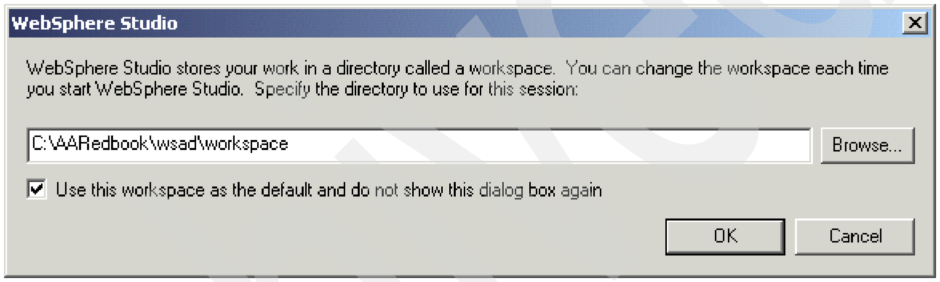
  When installing WebSphere Studio Application Developer, normally the "typical" install works just fine. If you plan to deploy your application with a version of WebSphere Application Server earlier than WebSphere Application Server V5.0, you need to install an optional WebSphere Studio Application Developer test environment for the version of WebSphere Application Server you will deploy with.

  If you download WebSphere Studio Application Developer from the Web, it comes in 12 "parts." Part 10 (Agent Controller), part 11 (ClearCase®), and part 12 (Embedded MQ) are optional and are not required for eClient development.

Set up eClient application in WebSphere Studio Application Developer

Once you have EIP V8.2, eClient V8.2, and WebSphere Studio Application Developer V5.0 installed and working (that is, you can log on with eClient, search, and display documents from your back-end server), follow these steps to set up the eClient application in WebSphere Studio Application Developer:

1. Start WebSphere Studio Application Developer:
   a. Select **Start -> Programs -> IBM WebSphere Studio -> Application Developer 5.0**.
   b. Enter a workspace location directory. The workspace is where WebSphere Studio Application Developer stores all of the source files and configuration information for your WebSphere Studio Application Developer workspace. Do not use special characters in the directory, such as c:\$user\data.
   c. Check **Use this workspace as the default and do not show this window again** as shown in Figure 7-2. Click **OK** to continue.


WebSphere Studio

WebSphere Studio stores your work in a directory called a workspace. You can change the workspace each time you start WebSphere Studio. Specify the directory to use for this session:

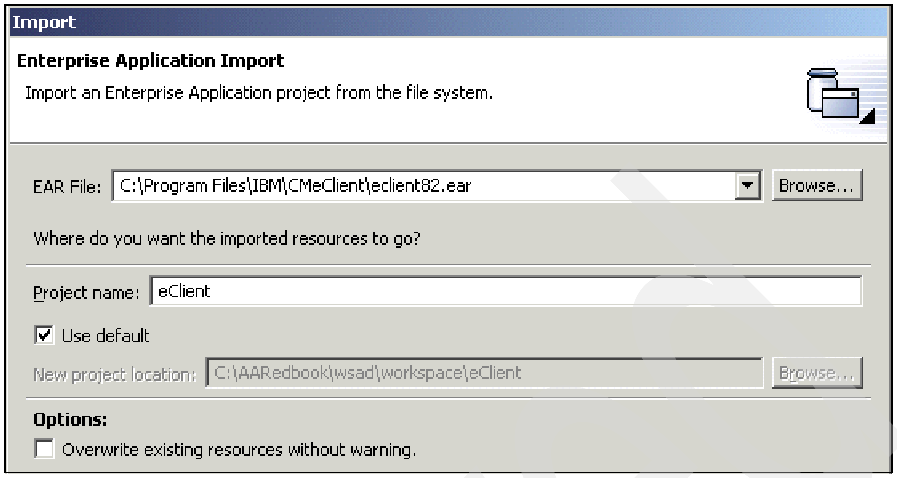
C:\AARedbook\wsad\workspace

☑ Use this workspace as the default and do not show this dialog box again

Figure 7-2 WebSphere Studio Application Developer - Set up workspace

2. Import the eClient.ear file into WebSphere Studio Application Developer:
   a. Select **File -> Import**. For Import Type, select **EAR file** and click **Next**.
   b. Use the Browse button to find the eclient82.ear file. By default, it is c:\Program Files\IBM\cmeClient\eclient82.ear. See Figure 7-3 on page 187.
   c. Enter `eClient` as the project name and click **Finish**.

Important: The project name must match the name of the application in the WebAppName field in the IDM.properties file. The IDM.properties file is normally located in c:\Program Files\IBM\cmEClient.


Import
Enterprise Application Import
Import an Enterprise Application project from the file system.
Browse...
Browse...
Options:

EAR File: C:\Program Files\IBM\CMeClient\eclient82.ear

Where do you want the imported resources to go?

Project name: eClient

[✓] Use default

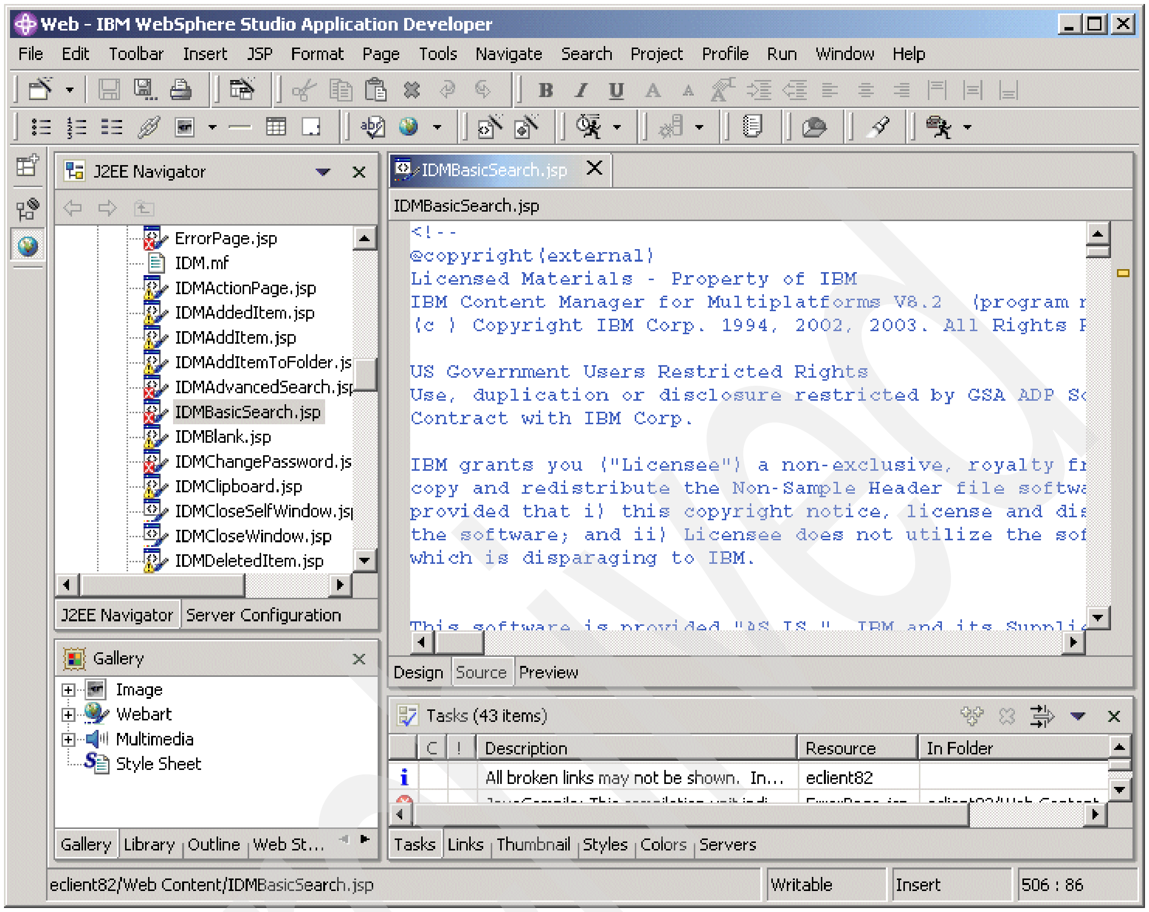
New project location: C:\AARedbook\wsad\workspace\eClient

[ ] Overwrite existing resources without warning.

Figure 7-3 WebSphere Studio Application Developer - Import eClient application

Note: This step may take from 5 to 10 minutes to complete. The import process creates directories in your workspace directory and extracts eClient source files from the EAR file to these directories. When using WebSphere Studio Application Developer to modify eClient, you must edit this set of files and not the files in the WebSphere directories.

3. Open the Web Perspective to view the project content:
   a. Select **Window -> Open Perspective -> Other...**.
   b. From the displayed list, select the Web perspective. It should look similar to Figure 7-4 on page 188.


Web - IBM WebSphere Studio Application Developer
File Edit Toolbar Insert JSP Format Page Tools Navigate Search Project Profile Run Window Help
J2EE Navigator
ErrorPage.jsp
IDM.mf
IDMActionPage.jsp
IDMAddedItem.jsp
IDMAddItem.jsp
IDMAddItemToFolder.js
IDMAdvancedSearch.jsp
IDMBasicSearch.jsp
IDMBlank.jsp
IDMChangePassword.js
IDMClipboard.jsp
IDMCloseSelfWindow.jsp
IDMCloseWindow.jsp
IDMDeletedItem.jsp
J2EE Navigator | Server Configuration
Gallery
Image
Webart
Multimedia
Style Sheet
Gallery | Library | Outline | Web St...
IDMBasicSearch.jsp
Design | Source | Preview
Tasks (43 items)
C ! Description | Resource | In Folder
Tasks | Links | Thumbnail | Styles | Colors | Servers

```
IDMBasicSearch.jsp
<!--
@copyright(external)
Licensed Materials - Property of IBM
IBM Content Manager for Multiplatforms V8.2  (program r
(c ) Copyright IBM Corp. 1994, 2002, 2003. All Rights R

US Government Users Restricted Rights
Use, duplication or disclosure restricted by GSA ADP Sc
Contract with IBM Corp.

IBM grants you ("Licensee") a non-exclusive, royalty fr
copy and redistribute the Non-Sample Header file softwa
provided that i) this copyright notice, license and dis
the software; and ii) Licensee does not utilize the sof
which is disparaging to IBM.

This software is provided "AS IS." IBM and its Suppli
```

Description	Resource	In Folder
All broken links may not be shown. In...	eclient82	

```
eclient82/Web Content/IDMBasicSearch.jsp    Writable    Insert    506 : 86
```

Figure 7-4 WebSphere Studio Application Developer - Web perspective

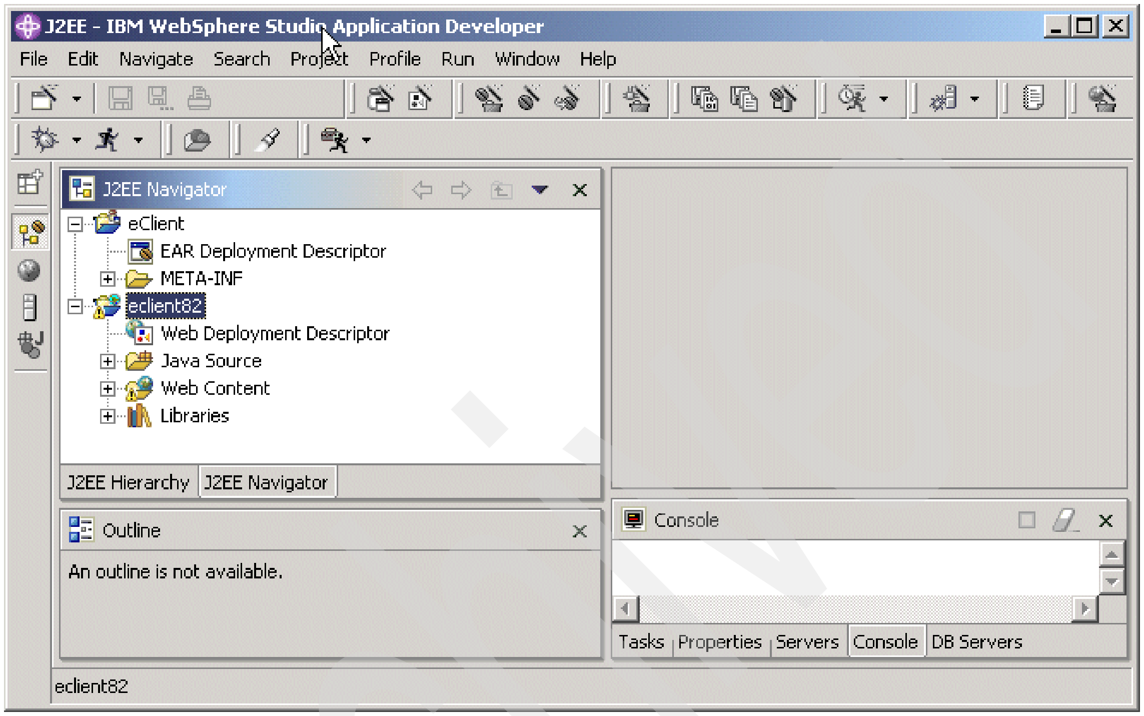
You may notice several errors. This is normal until the rest of the setup steps are complete.

The J2EE Navigator in the upper-left pane of the window (Figure 7-4) shows the content of the application. This is where you find the JSPs and any custom Java files that you add to your application. If you scroll down in the J2EE Navigator window, you see the JSPs that are included with the eClient. You can view and modify them from here.

This is where you work during development. You can access the various source files from the J2EE Navigator window and can edit the source in the editor. In the bottom-right pane, the Servers tab is where you start and stop the WebSphere test environment for testing your custom application.

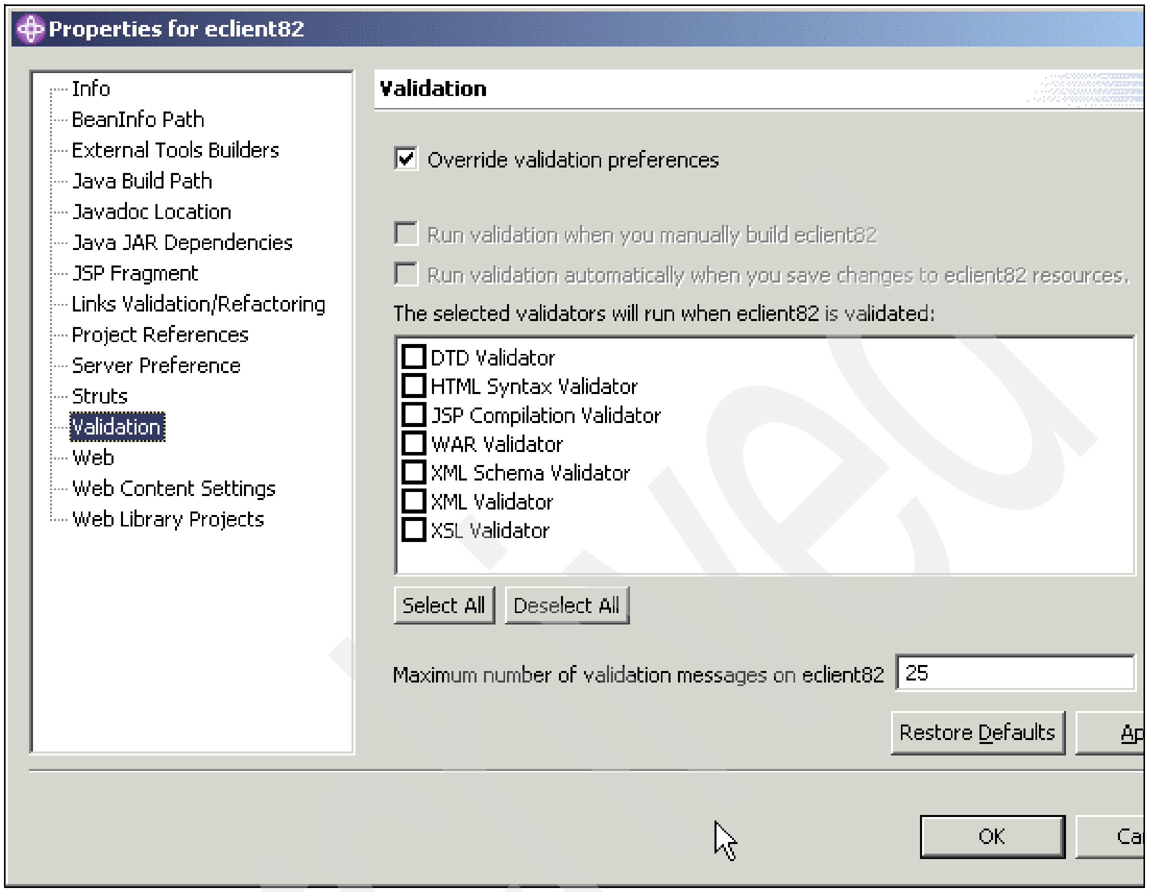
4. Turn off WebSphere Studio Application Developer validation:

   a. From the J2EE Navigator window (Figure 7-5), right-click the **eclient82** project and select **Properties**. Figure 7-6 on page 190 appears.



An outline is not available.

Figure 7-5 J2EE Navigator view



[✓] Override validation preferences

[ ] Run validation when you manually build eclient82

[ ] Run validation automatically when you save changes to eclient82 resources.

The selected validators will run when eclient82 is validated:

[ ] DTD Validator
[ ] HTML Syntax Validator
[ ] JSP Compilation Validator
[ ] WAR Validator
[ ] XML Schema Validator
[ ] XML Validator
[ ] XSL Validator

Maximum number of validation messages on eclient82: 25

Figure 7-6 WebSphere Studio Application Developer eClient properties window

b. Select **Validation** in the Properties for eclient82 window (Figure 7-6).

c. Check **Override validation preferences**.

d. Uncheck **Run validation automatically when you save changes** if it is checked.

e. Click **Deselect All**. Then click **Apply** and **OK**.

5. Delete unused help subdirectories.

This is optional but recommended. Performing this step saves a lot of time in the next step when changing the project properties. To do this:

a. Open the J2EE Navigator pane and expand the **eClient** project.

b. Expand **Web Content -> Help** folder.

c. Select all of the directories except your preferred language (in our case, it is en) and click the **Delete** key, or right-click and select **Delete** (see Figure 7-7). To select multiple directory entries, hold down the Ctrl key while selecting each directory. This may takes a couple of minutes to complete.

After completing this step, the J2EE Navigator window should include only one language. In our scenario, only the `en` (for English) directory still remains.

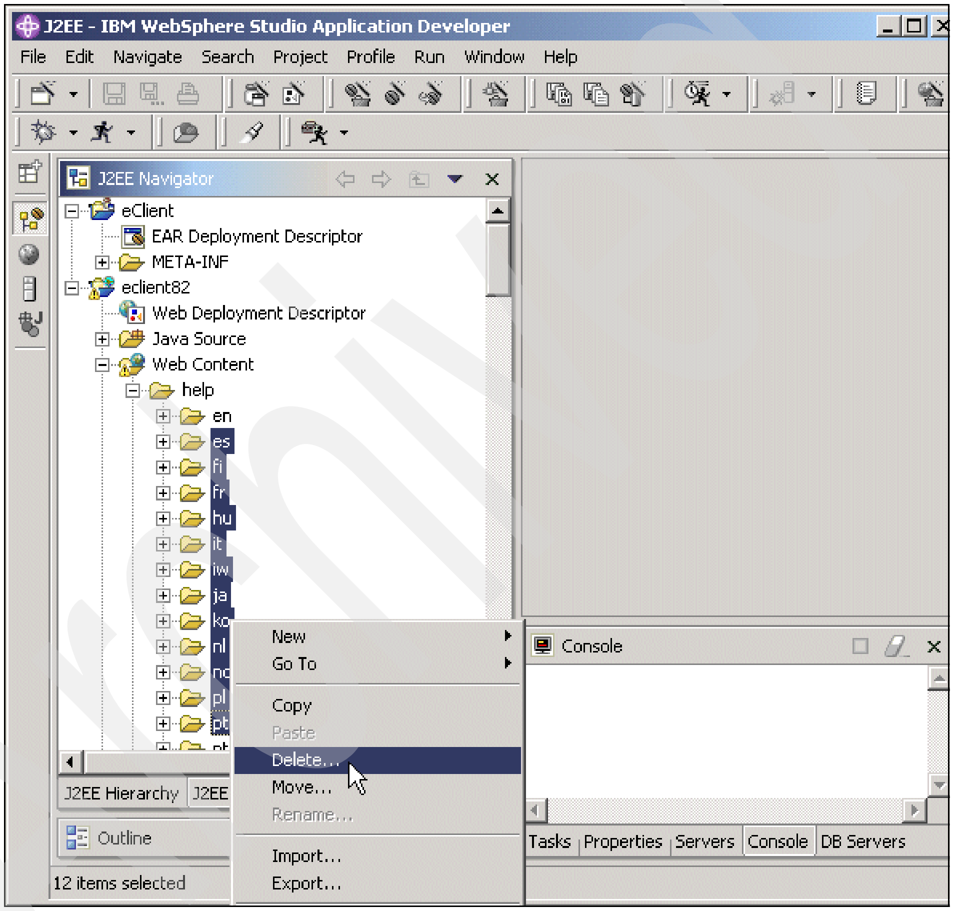
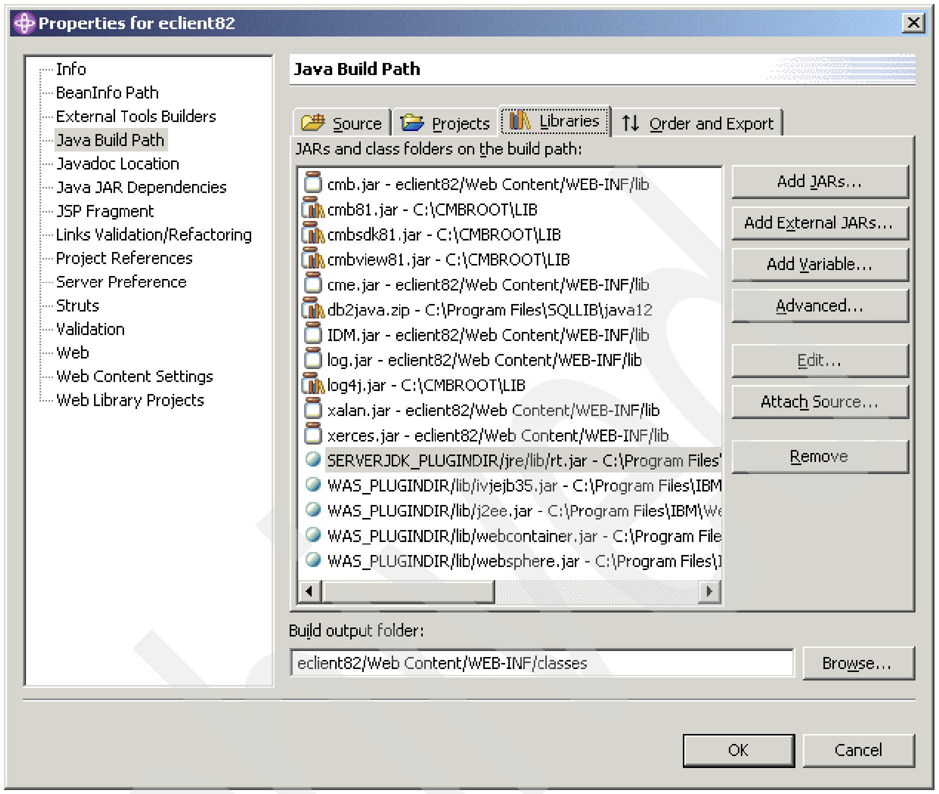


Figure 7-7 Deleting help folder from WebSphere Studio Application Developer

6. Specify the JAR files required by the eClient in the Properties page:

   a. From the J2EE Navigator view, right-click the **eclient82** project and select **Properties**.


Properties for eclient82

- Info
- BeanInfo Path
- External Tools Builders
- Java Build Path
- Javadoc Location
- Java JAR Dependencies
- JSP Fragment
- Links Validation/Refactoring
- Project References
- Server Preference
- Struts
- Validation
- Web
- Web Content Settings
- Web Library Projects

Java Build Path

JARs and class folders on the build path:

```
cmb.jar - eclient82/Web Content/WEB-INF/lib
cmb81.jar - C:\CMBROOT\LIB
cmbsdk81.jar - C:\CMBROOT\LIB
cmbview81.jar - C:\CMBROOT\LIB
cme.jar - eclient82/Web Content/WEB-INF/lib
db2java.zip - C:\Program Files\SQLLIB\java12
IDM.jar - eclient82/Web Content/WEB-INF/lib
log.jar - eclient82/Web Content/WEB-INF/lib
log4j.jar - C:\CMBROOT\LIB
xalan.jar - eclient82/Web Content/WEB-INF/lib
xerces.jar - eclient82/Web Content/WEB-INF/lib
SERVERJDK_PLUGINDIR/jre/lib/rt.jar - C:\Program Files'
WAS_PLUGINDIR/lib/ivjejb35.jar - C:\Program Files\IBM
WAS_PLUGINDIR/lib/j2ee.jar - C:\Program Files\IBM\We
WAS_PLUGINDIR/lib/webcontainer.jar - C:\Program File
WAS_PLUGINDIR/lib/websphere.jar - C:\Program Files\I
```

Build output folder:

eclient82/Web Content/WEB-INF/classes

Figure 7-8 Adding JAR files to WebSphere Studio Application Developer properties

b. Select **Java Build Path** in the list in the left-hand pane. See Figure 7-8.

c. Select the **Libraries** tab on the right.

d. Click **Add External JARS** and add the following JAR files from C:\CMBROOT\lib directory:

- cmb81.jar (Yes, the files are named 81 even though this is for 8.2.)
- cmbsdk81.jar
- cmbview81.jar
- log4j.jar (This is required only if you enable log4j logging.)
- cmblog4j81.jar (This is required only if you enable log4j logging.)

e. Click **Add External JARS** again and add the db2java.zip JAR file from the C:\Program Files\sqllib\java12 directory.

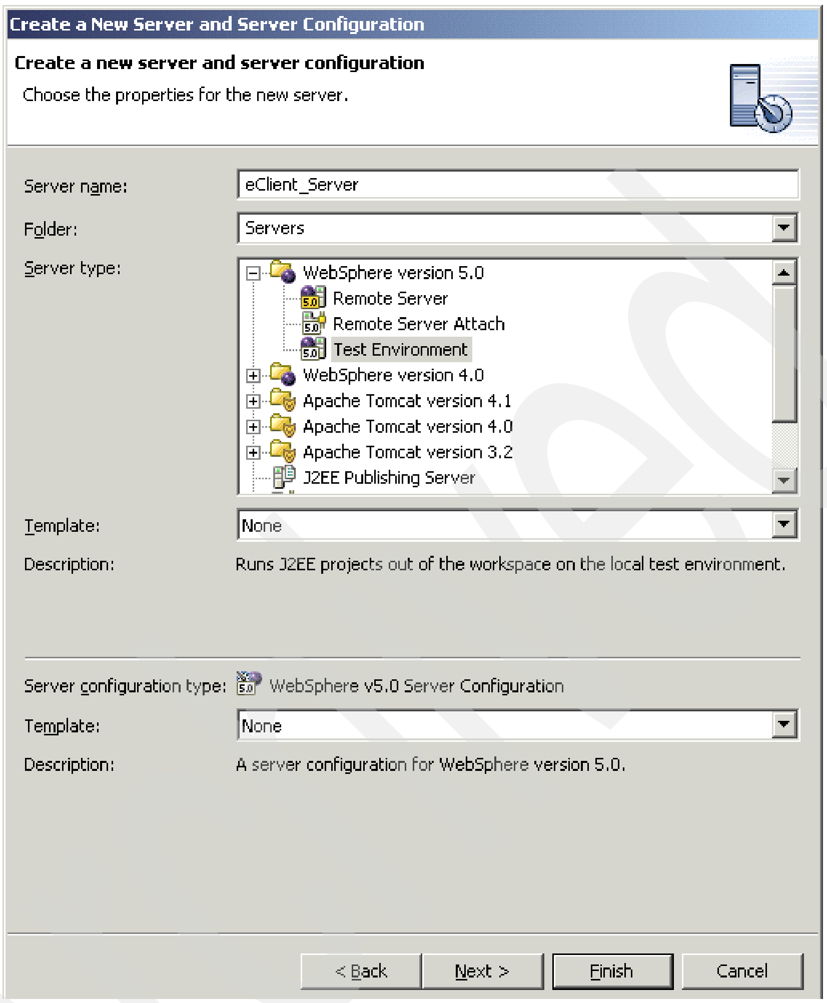
f. Click **OK** at the bottom of the Properties window.

7. Configure a test environment in WebSphere Studio Application Developer:

   When you execute an application in WebSphere Studio Application Developer, do not run it in your normal WebSphere environment. WebSphere Studio Application Developer creates a test environment in which you can run your application. The test environment includes an HTTP server and WebSphere Application Server environment. WebSphere Studio Application Developer can support multiple test environments for different levels of WebSphere Application Server, allowing you to test the application in the same environment you will deploy your application to.

   To create a WebSphere Studio Application Developer test environment, perform the following steps:

   a. Select **File -> New -> Other**. Select **Server** on the left, and **Server and Server Configuration** on the right, then click **Next**.


Create a New Server and Server Configuration

Create a new server and server configuration

Choose the properties for the new server.

Server name: eClient_Server

Folder: Servers

Server type:

```
⊟ WebSphere version 5.0
     Remote Server
     Remote Server Attach
     Test Environment
⊞ WebSphere version 4.0
⊞ Apache Tomcat version 4.1
⊞ Apache Tomcat version 4.0
⊞ Apache Tomcat version 3.2
  J2EE Publishing Server
```

Template: None

Description: Runs J2EE projects out of the workspace on the local test environment.

Server configuration type: WebSphere v5.0 Server Configuration

Template: None

Description: A server configuration for WebSphere version 5.0.

Figure 7-9 Creating test environment server configuration

b. For the server name, enter `eClient_Server`. For the default folder, select **Servers**. For the server type, expand **WebSphere version 5.0**, then select **Test Environment**. See Figure 7-9. If you plan to use a WebSphere Application Server version other than the default, select the version you plan to deploy here.

c. Click **Next** to get to the HTTP server port number.

d. Optional for WebSphere Application Server V4.0 test environment: If you would like to run your WebSphere Application Server version of the eClient at the same time as your WebSphere Studio Application Developer version of the eClient, you must ensure that the port numbers for your

WebSphere Studio Application Developer test environment are different from the port numbers for WebSphere Application Server. If you are using a WebSphere Application Server V5.0 test environment in WebSphere Studio Application Developer, the defaults should work fine.

Important: You can run both the standard out-of-the box eClient and your custom eClient on the same machine at the same time. This is useful for comparing the functionality of your custom eClient with that of the out-of-the box eClient.

If you use a WebSphere Application Server V5.0 test environment, the setup automatically uses different ports.

If you use a WebSphere Application Server V4.0 test environment, you must specify port numbers in your WebSphere Studio Application Developer test environment that do not interfere with the port numbers used by the standard HTTP server and WebSphere Application Server on your development workstation. If you use the default port numbers, you can still run both the WebSphere Application Server and WebSphere Studio Application Developer versions of the eClient, but you must have only one “environment” running at one time: either the standard WebSphere Application Server and HTTP server or your WebSphere Studio Application Developer test environment.

e. Click **Finish**.

f. Select **Window -> Open Perspective -> Server**.

g. Expand the **Servers** folder in the Server Configurations pane on the lower left of the window, right-click **eClient_Server** and select **Open**.

h. From the right pane in the window, select the **Paths** tab in the WebSphere-specific classpath section and perform the following:

   Click **Add External Folder...**. Navigate to c:\program files\IBM\WebSphere Studio\runtimes\base_v5\properties. Click **OK**.

i. Select the **Paths** tab in the lower classpath section (you may need to use the scroll bar to scroll down the upper-right pane to get to this section), and perform the following:

   i. Click **Add External JARS**. Add the corresponding files from various directories as specified in Table 7-1 on page 196. Click **Open**.

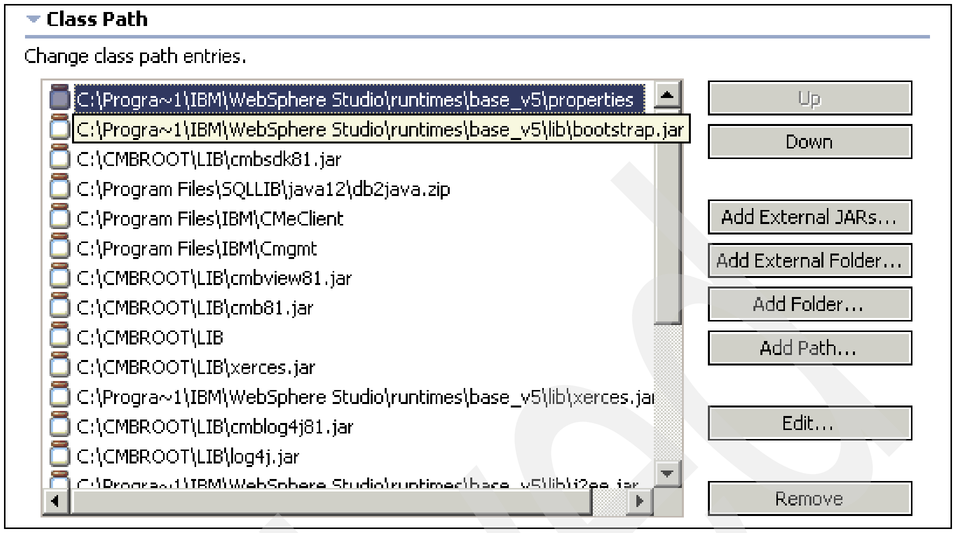
Table 7-1 External JARs needed to add to the class path

Directory	Files
C:\CMBROOT\lib	cmb81.jar cmbfed81.jar cmglog4j81.jar cmbsdk81.jar cmbservlets81.jar cmbtag81.jar cmbutil81.jar cmbutilfed81.jar cmbview81.jar log4j.jar xalan.jar xerces.jar
C:\Program Files\sqllib\java12	db2java.zip
C:\Program Files\IBM\WebSphere Studio\runtimes\base_v5\lib	bootstrap.jar j2ee.jar xerces.jar

ii. For each of the directories below, click **Add External Folder...**, select the directory, and click **OK**.

- C:\CMBROOT\lib
- C:\Program Files\IBM\CMeClient
- C:\Program Files\IBM\Cmgmt
- C:\Program Files\IBM\WebSphere Studio
- C:\Program Files\IBM\WebSphere Studio\runtimes\base_v5\lib

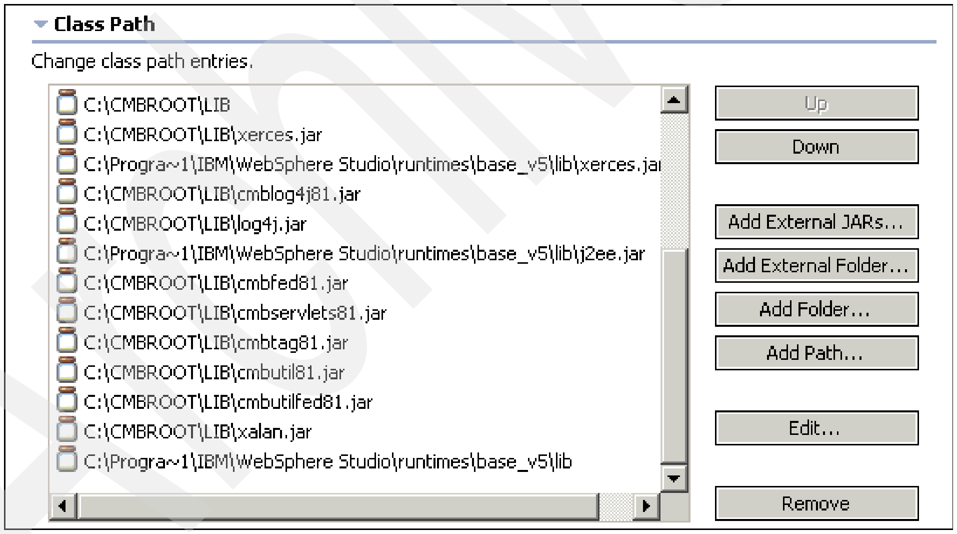
iii. Sort the entries in the lower Class Path entry field using the up and down buttons until the sequence matches the windows on Figure 7-10 on page 197 and Figure 7-11 on page 197. The sequence is important so that the correct files are pulled from the correct folders.


Class Path

Change class path entries.

```
C:\Progra~1\IBM\WebSphere Studio\runtimes\base_v5\properties
C:\Progra~1\IBM\WebSphere Studio\runtimes\base_v5\lib\bootstrap.jar
C:\CMBROOT\LIB\cmbsdk81.jar
C:\Program Files\SQLLIB\java12\db2java.zip
C:\Program Files\IBM\CMeClient
C:\Program Files\IBM\Cmgmt
C:\CMBROOT\LIB\cmbview81.jar
C:\CMBROOT\LIB\cmb81.jar
C:\CMBROOT\LIB
C:\CMBROOT\LIB\xerces.jar
C:\Progra~1\IBM\WebSphere Studio\runtimes\base_v5\lib\xerces.jar
C:\CMBROOT\LIB\cmblog4j81.jar
C:\CMBROOT\LIB\log4j.jar
```

Figure 7-10 WebSphere Studio Application Developer classpath configuration


Class Path

Change class path entries.

```
C:\CMBROOT\LIB
C:\CMBROOT\LIB\xerces.jar
C:\Progra~1\IBM\WebSphere Studio\runtimes\base_v5\lib\xerces.jar
C:\CMBROOT\LIB\cmblog4j81.jar
C:\CMBROOT\LIB\log4j.jar
C:\Progra~1\IBM\WebSphere Studio\runtimes\base_v5\lib\j2ee.jar
C:\CMBROOT\LIB\cmbfed81.jar
C:\CMBROOT\LIB\cmbservlets81.jar
C:\CMBROOT\LIB\cmbtag81.jar
C:\CMBROOT\LIB\cmbutil81.jar
C:\CMBROOT\LIB\cmbutilfed81.jar
C:\CMBROOT\LIB\xalan.jar
C:\Progra~1\IBM\WebSphere Studio\runtimes\base_v5\lib
```

Figure 7-11 WebSphere Studio Application Developer classpath configuration - Continued

j. Optional: Change the ORB bootstrap port so that your custom WebSphere Studio Application Developer eClient can run with your standard WebSphere Application Server eClient. With the eClient_Server Properties window still open, select the **Ports** tab and change the port for the ORB bootstrap from the default of `2809` to `2810`.

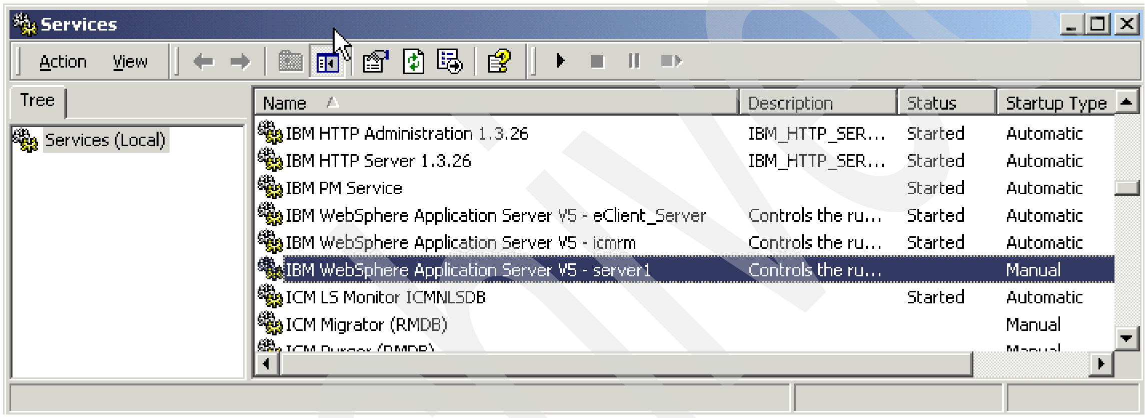
k. Press Ctrl+S to save the changes. Close the window by clicking the X on the **eClient_Server** tab.

l. Right-click **eClient_Server** again, and select **Add -> eClient**.

8. Test the eClient environment in WebSphere Studio Application Developer:

a. Select **Window -> Open Perspective -> Server**.

b. Stop the WebSphere Application Server service highlighted in Figure 7-12. With the service stopped, you cannot run the WebSphere Application Server Administration Console. This is because you cannot run the WebSphere Application Server Administration Console and the WebSphere Studio Application Developer test environment at the same time.



Name	Description	Status	Startup Type
IBM HTTP Administration 1.3.26	IBM_HTTP_SER...	Started	Automatic
IBM HTTP Server 1.3.26	IBM_HTTP_SER...	Started	Automatic
IBM PM Service		Started	Automatic
IBM WebSphere Application Server V5 - eClient_Server	Controls the ru...	Started	Automatic
IBM WebSphere Application Server V5 - icmrm	Controls the ru...	Started	Automatic
IBM WebSphere Application Server V5 - server1	Controls the ru...		Manual
ICM LS Monitor ICMNLSDB		Started	Automatic
ICM Migrator (RMDB)			Manual

Figure 7-12 Stop WebSphere Application Server service

c. Start the WebSphere Studio Application Developer test environment.

In the Server Configuration pane, right-click **eClient_Server** in the right pane, then select **Control -> Start** to start the test environment.

This causes WebSphere Studio Application Developer to start an HTTP server and WebSphere Application Server environment for your project. The Console window will open and you will see messages from the HTTP Server and WebSphere Studio Application Developer initialization. After about 15-30 seconds, it should display a message saying "`Server <server> open for e-business.`"

d. To execute the eClient in your WebSphere Studio Application Developer workspace, open a Web browser, and enter the following URL:

```
http://hostname/eClient82:9080/IDMInit
```

e. To execute the eClient in your standard WebSphere Application Server environment, enter the following URL:

```
http://hostname/eClient82:8080/IDMInit
```

Notes:

- Both WebSphere Application Server and WebSphere Studio Application Developer eClients use the same IDM.properties in c:\Program Files\IBM\CMgmt and cmEClient.
- The first time you run the eClient, it may run slowly since the JSPs will be compiled as they are used.
- If you are using a Content Manager V7.1 back-end and have 1 GB of RAM or more, set the FRNADDRON environment variable.

  Not doing so may result in the error FRN9255A in the WebSphere Studio Application Developer Console when logging on to EIP with the eClient.

  Use the following bat file to start WebSphere if you encounter this problem.

  ```
  set FRNADDRON=YES
  cd \Program Files\IBM\WebSphere Studio
  wsappdev.exe
  ```

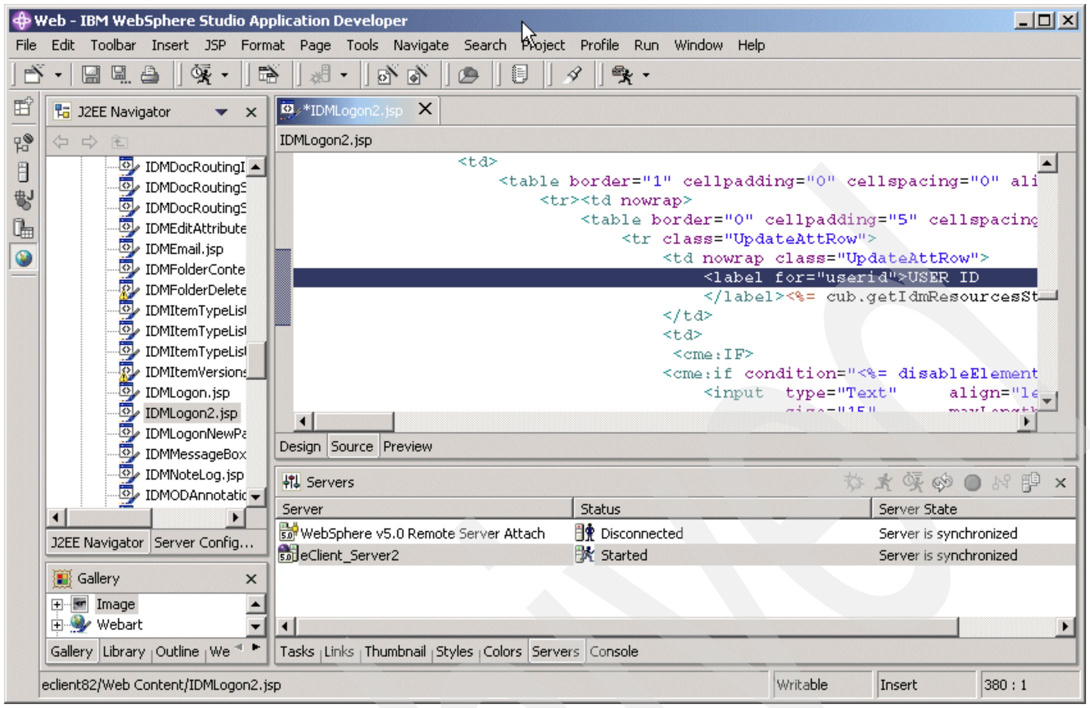
This completes the setup of WebSphere Studio Application Developer for the eClient.

Test changes to JSP files

To test this, we make a simple change to a JSP file.

In WebSphere Studio Application Developer, open the Web perspective. Scroll down in the J2EE Navigator window until you find IDMLogon2.jsp. This is the JSP that builds the logon window for the eClient.

In the following example, we change the label for the user ID field from **“**`User ID`” to “`USER ID`**”**. See Figure 7-13 on page 200.



```
<td>
    <table border="1" cellpadding="0" cellspacing="0" ali
        <tr><td nowrap>
            <table border="0" cellpadding="5" cellspacinc
                <tr class="UpdateAttRow">
                    <td nowrap class="UpdateAttRow">
                        <label for="userid">USER ID
                        </label><%= cub.getIdmResourcesSt
                    </td>
                    <td>
                     <cme:IF>
                    <cme:if condition="<%= disableElement
                        <input  type="Text"      align="le
```

Server	Status	Server State
WebSphere v5.0 Remote Server Attach	Disconnected	Server is synchronized
eClient_Server2	Started	Server is synchronized

Figure 7-13 IDMLogon2.jsp in WebSphere Studio Application Developer

To change the label, do the following

1. Locate the following line in IDMLogon2.jsp by searching for “`LogonJSP_UserIDPrompt`”:

   ```
   <label for="userid"><%=
   cub.getIdmResourcesString("LogonJSP_UserIDPrompt") %>
   ```

   and replace it with this:

   ```
   <label for="userid">USER ID
   ```

2. Press Ctrl+S from the editor window to save the changes. Check the servers in the bottom right pane of WebSphere Studio Application Developer to be sure your WebSphere test environment is started. Right-click the server and click **Start** if it is not started. Go back to the Web browser and enter the following URL to log on to the eClient:

   ```
   http://hostname/eClient82:9080/IDMInit
   ```

The original logon window is shown in Figure 7-14 on page 201.

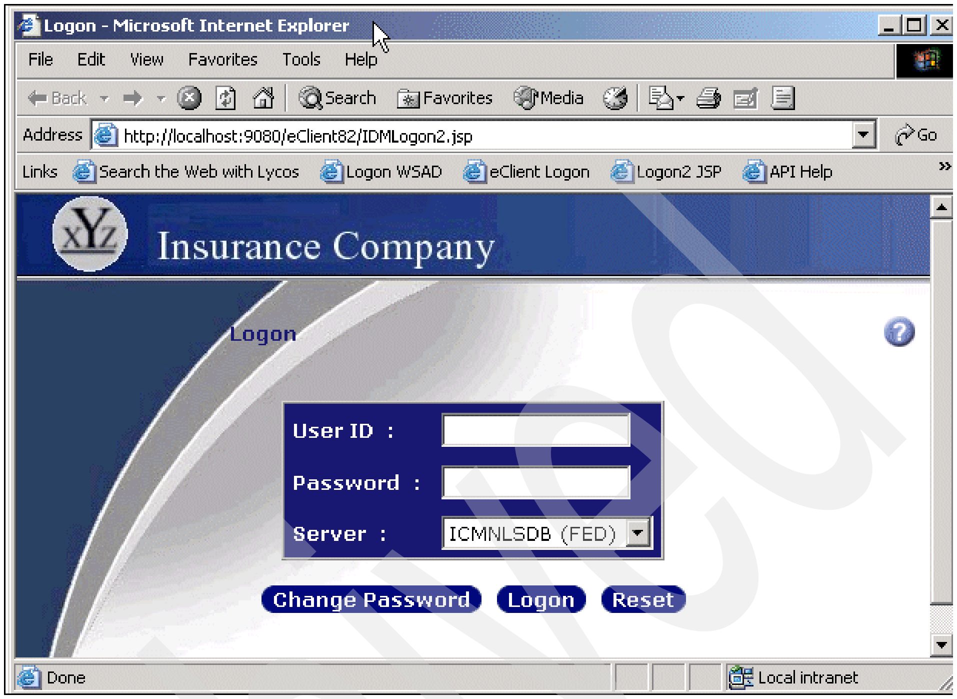

Logon - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:9080/eClient82/IDMLogon2.jsp
Go
Links Search the Web with Lycos Logon WSAD eClient Logon Logon2 JSP API Help
Insurance Company
Logon
User ID :
Password :
Server : ICMNLSDB (FED)
Change Password Logon Reset
Done
Local intranet

Figure 7-14 Standard eClient logon window with XYZ background image

The new logon window is shown in Figure 7-15 on page 202, with the label for the USER ID field all in uppercase. Note that the JSP is compiled by WebSphere when the new file is accessed.

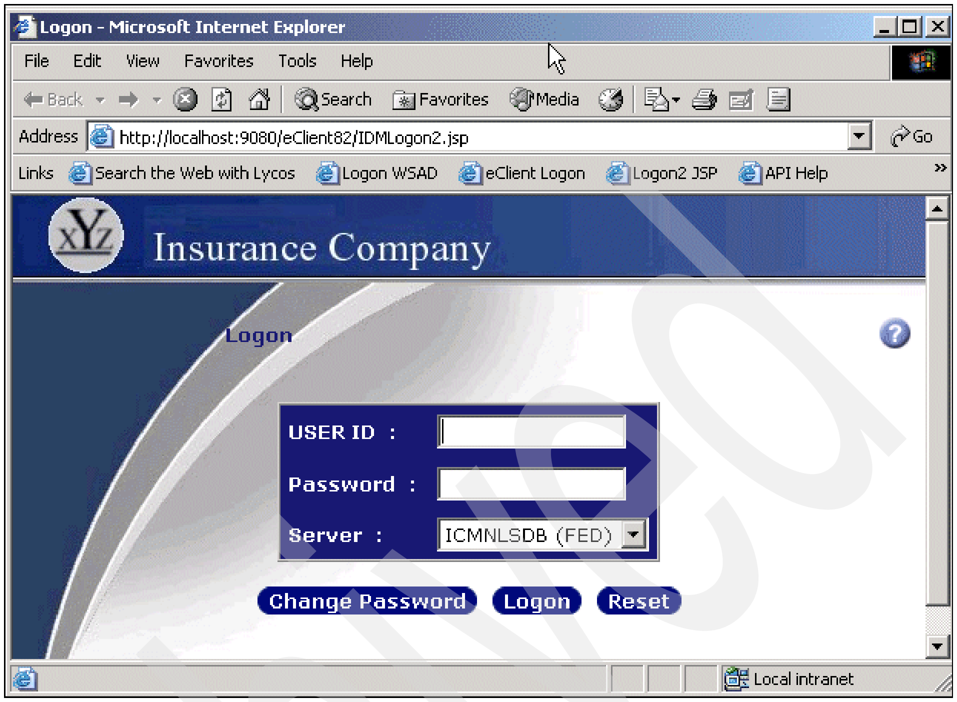

Logon - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:9080/eClient82/IDMLogon2.jsp
Go
Links Search the Web with Lycos Logon WSAD eClient Logon Logon2 JSP API Help
Insurance Company
Logon
USER ID :
Password :
Server : ICMNLSDB (FED)
Change Password Logon Reset
Local intranet

Figure 7-15 Modified IDMLogon2.jsp specifying uppercase label for user ID

If the user ID is not uppercase, check the following:

- Make sure you saved the new IDMLogon2.jsp file.
- Make sure you specified the correct port for your WebSphere Studio Application Developer environment. If you are not sure, stop the WebSphere Studio Application Developer test environment from the servers pane and try to log on using the same URL. It should not be able to find the Web page at all. If it does find the Web page, you are using the URL from another instance of WebSphere Application Server.

Do not forget to put back the original code in IDMLogon2.jsp before continuing.

7.4 Deploying customized eClient to another system

If you are modifying a few JSP files, you do not need to create a new .ear file for your Web application. You can copy the new JSPs to the installedApps directory of the target WebSphere environment.

If you add a custom JSP or servlet to eClient, you can create a new eClient .ear file, which can then be deployed to WebSphere. If you add a servlet, a new .ear file must be created and deployed to WebSphere.

If you are using WebSphere Studio Application Developer for development, WebSphere Studio Application Developer can create a new .ear file that includes the files you modified or added to eClient using WebSphere Studio Application Developer. To do this, select the **eclient82** project in WebSphere Studio Application Developer, right-click the project, select **Export...** and use the GUI to create a new .ear file.

If you are not using WebSphere Studio Application Developer, you can use the WebSphere Application Server Application Assembly Tool. Chapter 13, “Enabling metadata-based content retrieval” on page 281 shows you how to add a servlet or a custom JSP to eClient and how to use the WebSphere Application Assembly Tool to create and deploy a new eClient .ear file.

Part 3

Customizing eClient

In this part, we show you how to customize eClient with sample codes. By demonstrating some of the customization with the sample codes, we intend to give you a better understanding of what and how you can customize eClient for your business needs.

8

Design and implementation considerations

In this chapter, we discuss a variety of issues that should be considered before starting an eClient customization project. These include high-level design and implementation guidelines as well as tips and sample code on how to handle general programming tasks such as reading configuration files and handling errors.

Specifically, this chapter covers:

- Design considerations, including:
  - Considerations for incorporating future upgrades of eClient
  - Considerations for working with different EIP back-end servers
  - EIP back-end access vs direct access to back-end servers
  - Maximum length of a URL
  - Variations in Web browsers
- General programming tips, including:
  - Accessing configuration data using properties files
  - Using cookies to maintain session status
  - Invoking code on the server from JavaScript

8.1 Design considerations

Before you begin designing and implementing your customized eClient, there are several important considerations:

- ► Considerations for incorporating future upgrades of eClient
- ► Considerations for working with different EIP back-end servers
- ► EIP back-end access vs direct access to back-end servers
- ► Maximum length of a URL
- ► Variations in Web browsers

In the following sections, we address each of these in detail.

8.1.1 Considerations for incorporating future upgrades of eClient

One very important factor to consider when customizing the eClient is that IBM will continue to update and enhance eClient. This may include changes to the servlets, tag libraries, and most importantly, to the JSPs.

Since there are many JSPs that you can modify or replace with your own, it is important to give careful consideration to how much you customize the JSPs and how you will merge your customization with future versions of the eClient JSPs.

Keeping your changes manageable and easy to merge into new versions of the eClient will make it easier to take advantage of new eClient functionality provided in future versions of the eClient.

There are several things you can do to make this task easier:

1. Minimize the number of changes you make to the eClient JSPs.

   This is a very simple yet important concept. While it can be tempting to begin making small changes to lots of the JSPs to "tweak" the user interface, it is important to remember that if you want to take advantage of new eClient features in future releases, most likely you will need to re-apply and re-test every customization that you make.

2. Keep as much new code as possible out of the JSPs.

   Put as much custom code into new code modules outside of the JSPs as possible. This can be in brand-new JSPs being the view for newly added functionality (servlets) or new tag libraries. Using tag libraries reduces complexity in the JSPs, allows implementation reuse, and simplifies migration to a new eClient version.

3. Clearly mark your custom code in the JSPs.

   You can make it easier to find your customizations by putting comments before and after each change. This has been done in the examples in the redbook by inserting `XYZ_CHG_BEGIN` and `XYZ_CHG_END` before and after each customization, where `XYZ` is the acronym for the fictional company.

Another alternative is to make each of your customizations configurable via a properties file. This makes it easy to switch back to the default functionality of the eClient, and also ensures that all changes are clearly marked. However, this does require additional effort and results in more source code. Many of the sample customizations in the later chapters of this redbook use this method.

We also recommend using file comparison tools to make it easier to compare source files.

8.1.2 Considerations for working with different EIP back-end servers

Different back-end servers provide different functionality in the eClient. The Content Manager Version 8 back-end server provides the richest set of functionality. Many other back-ends may provide only search-and-view functionality. You need to be aware of this if you are considering using different back-end repositories.

8.1.3 EIP back-end access vs direct access to back-end servers

The eClient can connect directly to a back-end server such as Content Manager Version 7 or Content Manager Version 8. As an alternative, the eClient can also connect to the EIP database and use federated access to the back-end servers.

It is important to note that even if you access your back-end server by connecting to the EIP federated connector, you may see native back-end interfaces for some functions as you drill down on documents in the eClient. For example, when you get into the edit attributes window, the interface no longer uses the attribute display names used in the search templates and the search results. Instead, the system goes directly to the back-end and displays the fields in the order they are defined in Content Manager and with the attributes names defined in Content Manager.

In addition, if you use an EIP search template to perform a search and get a list of folders, when you open the folder, the eClient must revert to a back-end specific view of the folder. Therefore, you see the Content Manager Index Class or Item Type names and the Content Manager attributes names after opening a folder.

8.1.4 Maximum length of a URL

There are two ways to interface with servlets: by putting parameters in the URL or by putting parameters for the servlet in the session object. Each browser has different limits on the length of a URL. In general, you should not use more than 300 characters in a URL.

For example, if you add a new function to the action drop-down box in the eClient search results that can act on more than one item, do not pass the list of PIDs for all selected items in the URL directly. You may want to check the length of the generated list from the selected items to see if you have exceeded the maximum. You need to handle the situation when the maximum length is exceeded.

8.1.5 Variations in Web browsers

If you want to use HTML that is browser specific, for example the Netscape <embed> tag, you should check the browser version in the session object. Be aware that there are differences in the way Web browsers work.

The version of the requesting Web browser is available in the session object and can therefore be checked in a custom servlet or the JSPs.

8.2 General programming tips

In many cases, your customizations may need to have configurable properties. Two common ways to specify configuration information are via properties files and XML files.

An example of using properties files is the IDM.properties file in the eClient installation directory. Properties files can easily be read using class java.util.ResourcBundle or java.util.Properties.

An example of an XML configuration file is the icmrm_logging.xml in the resource manager application. A frequently used XML parser is the Xerces parser, also contained in WebSphere Studio Application Developer.

8.2.1 Accessing configuration data using properties files

A properties file is similar to a Windows .ini file. A properties file contains a pair of keywords and associated values. For example, you may have a properties file, c:\XYZCorp\config.properties, that contains the following text:

```
LOG_FILE_NAME=c:\XYXCorp\eClient.log
ENABLE_PERFORMANCE_LOGGING=true
ENABLE_IDM_EDIT_ATTRIBUTES_CUSTOMIZATIONS=true
```

You can use the following code in your application to retrieve those values:

```
import java.io.FileInputStream;
import java.util.Properties;
Properties eClientProps = new Properties();
FileInputStream in = null;
String strValue = "";
in = new FileInputStream("C:\\XYZCorp\\config.properties");
eClientProps.load(in);
in.close();
strValue = eClientProps.getProperty("LOG_FILE_NAME");
```

The example above (with error handling added) is included in the sample code XYZUtils.java and is called from the modified version of IDMNoteLog.jsp. For more information about using the properties object, see the following URL:

http://java.sun.com/docs/books/tutorial/essential/attributes/properties.htm
l

8.2.2 Using cookies to maintain session status

A cookie is a piece of data passed between a Web server and a Web browser. The Web server sends a cookie that contains data it requires the next time the browser accesses the server. The client returns the cookie on each subsequent request. This is one way to maintain the state between a browser and a server. Cookie data can be stored so that it is only valid for the current browser session, or it can be stored for a longer period of time specified by your application.

If the browser supports cookies, these are used to associate the browser with a session object on the server. For example when using the HttpSession support in the servlet API, the session manager can use cookies to store a session ID on the browser.

But you can also use the Cookies API directly to store and retrieve user-specific data.

Example 8-1 Cookie API sample

```
public void doGet(HttpServletRequest req, HttpServletResponse res) {
    Cookie[] cookies = req.getCookies();
    if(cookies != null) {
        for(int i=0; i<cookies.length; i++) {
            System.out.println(cookies[i].getName()
                                   + "=" cookies[i].getValue());
        }
    }
}
```

Cookies should not be used for things best kept on the server, such as personanlization settings, or for secure information, such as credit card numbers.

The eClient provides several JavaScript utility functions, including cookie functions, in the eclient81.js file. To store data to a cookie from JavaScript, use code similar to the following:

```
setCookie("MY_VAR_NAME”, “My Cookie Value");
```

To retrieve data from a cookie, use code similar to the following:

```
var strValue = ““;
strValue = getCookie("ExtName");
```

If you want to initialize a field on a Web page, place the code in the init() JavaScript function. Many of the eClient JSPs use an init() JavaScript function in them. If not, one can be added. See IDMEditAttributes.jsp for an example.

8.2.3 Invoking code on the server from JavaScript

One way is to write a servlet wrapper that would get called in place of the eClient servlet. It would perform a specific action, then “forward” the request to the eClient servlet.

There are not that many scenarios where this is really required. Usually, there is a better way to do this by changing the application design.

9

Customizing look and feel using style sheets

In this chapter, we describe how you can change the look and feel of eClient using Cascading Style Sheets.

This chapter covers the following sections:

- What is Cascading Style Sheets (CSS)?
- Simple CSS example
- Using CSS for eClient
- Changing background image on eClient

9.1 What is Cascading Style Sheets (CSS)?

Cascading Style Sheets (CSS) are an industry standard method of changing the appearance of elements of HTML pages such as labels, text, entry fields, links, and buttons. CSS allows you to specify the margins and the background images such as a company logo for a document. HTML specifies the content and structure of the document such as fields, captions, links, and tables; the style sheet defines the appearance in terms of fonts, point size, colors, highlighting, and background images.

A style sheet is a text file (eClient uses eclient81.css) containing a list of tags and a set of display properties for each tag. Style sheet definitions can be embedded in HTML, but a more common use of style sheets is to define them in an external style sheet file with a CSS extension. A link statement is used in the HTML to tell the browser the name of the CSS file. When the browser processes the HTML and finds the link tag for a style sheet, it reads the style sheet file and applies the formatting rules to the HTML document.

In the HTML document, the style sheet formatting options are specified for an element of an HTML document (label, field, table, etc.) by using the CLASS parameter. The HTML author can identify the CLASS that is required for different formatting style and place the formatting rule for each CLASS in the CSS file.

In the following section, we show a simple example of how style sheets and HTML work together. In the later sections, we describe how the eClient JSPs use the style sheets and give some examples of how to modify the appearance of the eClient.

The style sheet standard has been promoted by Worldwide Web Consortium, also known as W3. To learn more about style sheets, see the following URL:

http://www.w3.org/Style/

In addition, the following link from the same Web site has useful information that helps you with colors and other style guides in your style sheet:

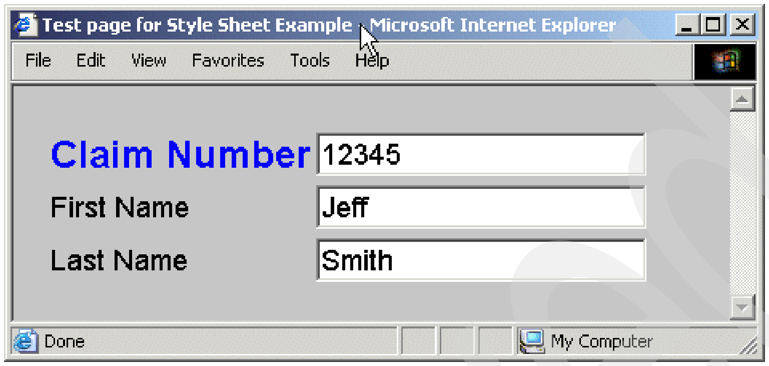
http://www.w3.org/MarkUp/Guide/Style

Most books on HTML usually have a significant section describing style sheets.

9.2 Simple CSS example

CSSSample.html and CSSSample.css are sample files that demonstrate the basics of style sheets.

The CSSSample.html file is what you open in a Web browser. It uses the CSSSample.css style sheet to define the look and feel of the window. When opening the CSSSample.html file in a browser, a window similar to Figure 9-1 appears.


Test page for Style Sheet Example - Microsoft Internet Explorer
File
Edit
View
Favorites
Tools
Help
Done
My Computer

Claim Number 12345
First Name Jeff
Last Name Smith

Figure 9-1 CSSSample.html using CSSSample.css

The CSSSample.html file is shown in Example 9-1. It uses the style sheet CSSSample.css, as specified on line 4 of the source code. The class="<tag>" entries specify the tag from the CSS file to be used to get the format data for the element of the HTML document.

Example 9-1 CSSSample.html

```
<html>
  <head>
    <title>Test page for Style Sheet Example</title>
    <link rel="STYLESHEET" type="text/css" href="c:\CSSSample.css">
  </head>

  <body class="BODY">
    <TABLE>
      <tr>
        <td class="label_important" id="ClmLabel">Claim Number</td>
        <td><input type="text" class="fld" id="ClmNo" value="12345"></td>
      </tr>
      <tr>
        <td class="label_normal" id=FirstNameLabel">First Name</td>
        <td><input type="text" class="fld" id="FName" value="Jeff"></td>
      </tr>
      <tr>
        <td class="label_normal" id=FirstNameLabel">Last Name</td>
        <td><input type="text" class="fld" id="LName" value="Smith"></td>
      </tr>
    </TABLE>
  </body>
</html>
```

The style sheet CSSSample.css defines the styles used in the CSSSample.html file. Its source code is shown in Example 9-2.

Example 9-2 CSSSample.css

```
/*  Sample style sheet for redbook example.  Use with CSSSample.html */

.BODY {
        background : silver;
        padding-top : 10px;
        padding-left : 10px;
}
.BODY_LOGO {
        background : White;
        padding-top : 100px;
        padding-left : 90px;
        background-image : url("c:\AARedbook\background.jpg");
}

.label_normal {
        font-family : Arial, Helvetica, sans-serif;
        color       : Black;
        font-size   : medium;
        font-weight : normal;
}
.label_important {
        font-family : Arial, Helvetica, sans-serif;
        color       : Blue;
        font-size   : large;
        font-weight : bold;
}
.fld {
        font-family : Arial, Helvetica, sans-serif;
        color       : Black;
        font-size   : medium;
        font-weight : normal;
}
```

To test how it works, paste the text from the redbook into c:\CSSSample.html and C:\CSSSample.css. Open the CSSSample.html file in a Web browser. You can see what it looks like initially. Modify the style sheet and the HTML file to see how it affects the output.

Notes on the style sheet:

- ► The .body tag is used by the HTML file for the body of the document.
- ► The tag names (.body, .body_logo, .label_normal, .label_important, and .fld) are references by the class="<tag>" statements in the HTML file. The browser applies the settings in the CSS file to the HTML element. The tag names for the CLASS entries are completely up to the author.
- ► The color field can be a named color, such as Black, Red, and Blue. You may also specify a hex RGB color. See `http://www.w3.org/MarkUp/Guide/Style` for a list of named colors and hex values for other popular colors that can be used.

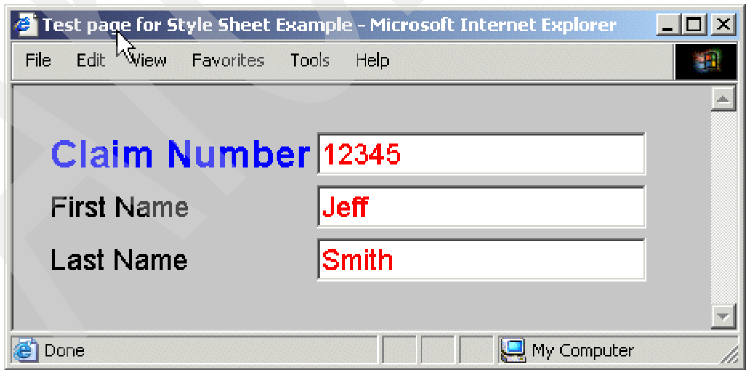
In the following sections, we provide examples of how you can change the sample.

9.2.1 Changing entry field text color

In the CSSSample.css file, change the color value from Black to Red for the tag named .fld as follows:

```
.fld {
        font-family : Arial, Helvetica, sans-serif;
        color       : Red;
        font-size   : medium;
        font-weight : normal;
}
```

Save the CSS file. Open the CSSSample.html again from your browser to view the changes (see Figure 9-2).


Test page for Style Sheet Example - Microsoft Internet Explorer
File
Edit
View
Favorites
Tools
Help
Done
My Computer

Claim Number 12345
First Name Jeff
Last Name Smith

Figure 9-2 CSSSample.html with red text for .fld tag

9.2.2 Using background image

In the CSSSample.html file, change the line:

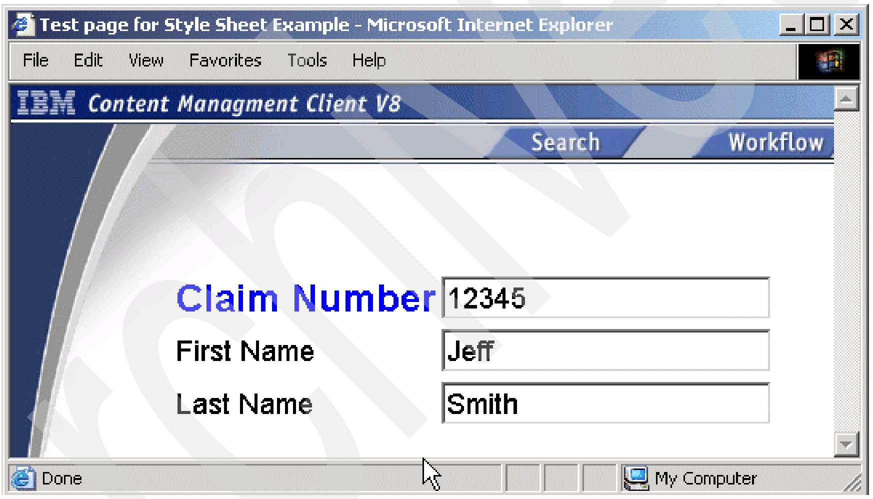
```
<body class=”BODY”>
```

to:

```
<body class="BODY_LOGO">
```

This triggers the HTML file to use the BODY_LOGO tag in the CSS file. The BODY_LOGO tag in the CSS file uses a background JPG file that can be found in the eClient icons directory. You can also use your company logo instead for customization.

Save the HTML file. Open the CSSSample.html again from your browser to view the changes (see Figure 9-3).


Test page for Style Sheet Example - Microsoft Internet Explorer
File Edit View Favorites Tools Help
IBM Content Managment Client V8
Search
Workflow
Done
My Computer

Claim Number 12345
First Name Jeff
Last Name Smith

Figure 9-3 CSSSample.html using <body class=”BODY_LOGO”>

9.3 Using CSS for eClient

In the eClient, HTML is generated when the JSPs execute on the server. Each JSP in the eClient has a section in it that creates HTML to send to the Web browser. In the HTML portion of each eClient JSP, a line similar to line 3 in the following code specifies the name of the style sheet to use:

```
<head>
<title><%= cub.getIdmResourcesString("IDMNoteLogJSP_TitleBar") %></title>
```

```
<link rel="STYLESHEET" type="text/css" href="<%= webAppName
%>/eclient81.css">
```

The above sample code is taken from IDMNoteLog.jsp.

All of the eClient JSPs use the eclient81.css file, which is normally located in the c:\Program Files\IBM\cmEClient\installedApp\ IBM_eClient_82.ear\eclient82.war directory.

Most of the tags in eclient81.css are used for various types of standard HTML elements. There are a few interesting examples that are a little different:

- The BODY tag

  There is no period at the beginning of this tag. The BODY tag is used as the default if there is no specific class specified for the body of an HTML document - that is, if the JSP uses <body> instead of <body class=”MY_BODY_TAG”>. Here you can specify defaults such as background color.

- The .BODY tags (.BODYLOGON, .BODYHOME, .BODYMINI)

  These tags are used in specific JSPs that need different BODY styles. These different tags are used primarily to specify a different background image for different type of windows, such as logon and search windows.

- The .ODD and .EVEN tags

  An interesting example of style sheet usage is how the eClient gets the alternating lines in the search results to have a blue or white background.

  The HTML for the search results uses the following code to specify a CLASS of ODD or EVEN to the odd and even lines in the search results. The variable i is the row number which is used to determine odd or even number of the row. The rowType is EVEN if i is divisible by 2; otherwise, the rowType is ODD. The tag for the class=”<tag>” line is then set to ODD or EVEN depending on the rowType.

  ```
  <% rowType = (i % 2 == 0) ? "EVEN" : "ODD"; i++; %>
  <TR class="<%= rowType %>">
  <TD nowrap class="<%= rowType %>">...
  ```

  The browser sets the colors according to the .EVEN or .ODD tags in the eclient81.css file.The above sample code is taken from the IDMItemTypeList.jsp file.

9.4 Changing background image on eClient

A common change to the look and feel of the eClient is changing the logo in the background of the eClient windows to a background specific to the project or your company.

Since many of the examples in this redbook are created for the XYZ Insurance company that is provided by the Content Manager sample documents, we create a background JPG file for the eClient for the XYZ Insurance company.

Open the eclient82.css file normally located in the c:\Program Files\IBM\cmEClient\installedApp\IBM_eClient_82.ear\eclient82.war directory. Notice that it uses several different background JPG files as shown in Example 9-3.

Example 9-3 eclient82.css code snippet

```
/* body - For Logon screen */
.BODYLOGON {
        background : White;
        padding-top : 0px;
        padding-left : 0px;
        background-image : url(icons/logon_bk.jpg);
}
/* body - Home page or Action Page */
.BODYHOME {
        background : White;
        padding-top : 0px;
        padding-left : 0px;
        background-image : url(icons/home_bk.jpg);
}
/* body - For Frame pages background */
.BODYMINI {
        background : White;
        padding-top : 0px;
        padding-left : 0px;
        background-image : url(icons/mini_bk.jpg);
        background-repeat : no-repeat;
}
```

If you want to customize the background images for your company or customer, make a copy of the original logon_bk.jpg, home_bk.jpg, and mini_bk.jpg files. and modify them to include your customized logo.

Here are some tips for modifying these files:

- ► Keep the graphic dimensions similar.

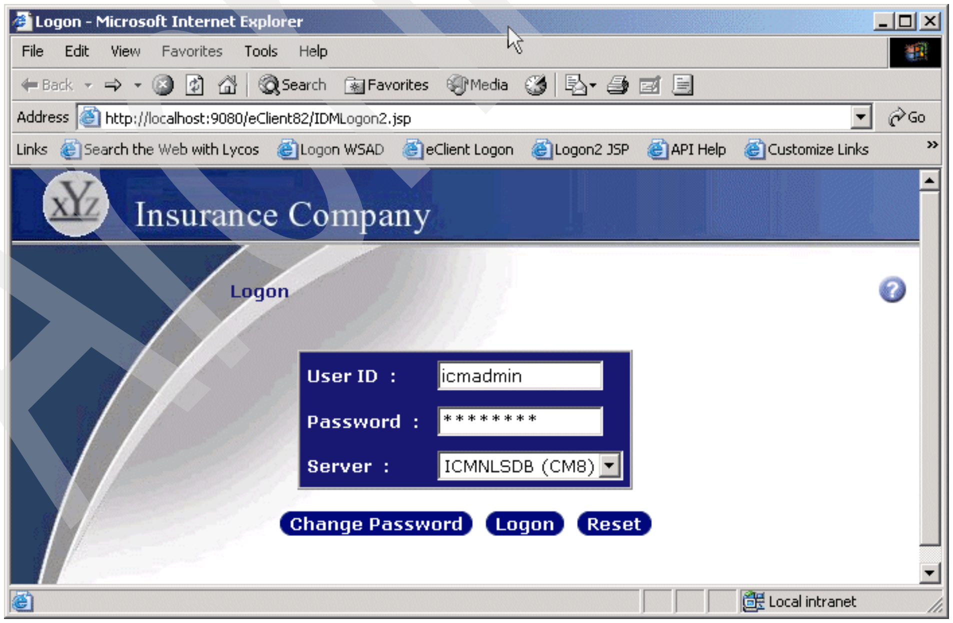
  When modifying these files, be sure not to change the overall dimensions of the graphics in the files. The JSPs are designed to be used with graphics in the approximate shape of the samples. If the shape of the graphics is changed, you may need to change the JSPs so that the text and fields generated in the HTML do not overlay the graphics.

- ► Keep the color scheme the same.

  The eClient uses a color scheme based on blue and white. If you use the same color scheme as the original background, it will look good with the rest of the eClient. If you want to change the overall color scheme of the eClient, you need to modify the colors specified elsewhere in the CSS file.

Once you have created the new JPG files, put them in the eclient82.war\icons directory and modify the eclient81.css file to specify the new JPG files (see the snippet from the CSS file above) instead of the old ones. Restart the eClient WebSphere application and the new graphics should be displayed when you log on to the eClient.

Figure 9-4 shows an example of the new eClient logon window using the customized XYZ Insurance company logos supplied with this redbook.


Logon - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Go
Links Search the Web with Lycos Logon WSAD eClient Logon Logon2 JSP API Help Customize Links
XYZ Insurance Company
Logon
Change Password
Logon
Reset
Local intranet

http://localhost:9080/eClient82/IDMLogon2.jsp

User ID : icmadmin

Password : ********

Server : ICMNLSDB (CM8)

Figure 9-4 XYZ company logon window with customized graphics

Figure 9-5 shows an example of the new eClient main window using the customized XYZ Insurance company logos supplied with the redbook.

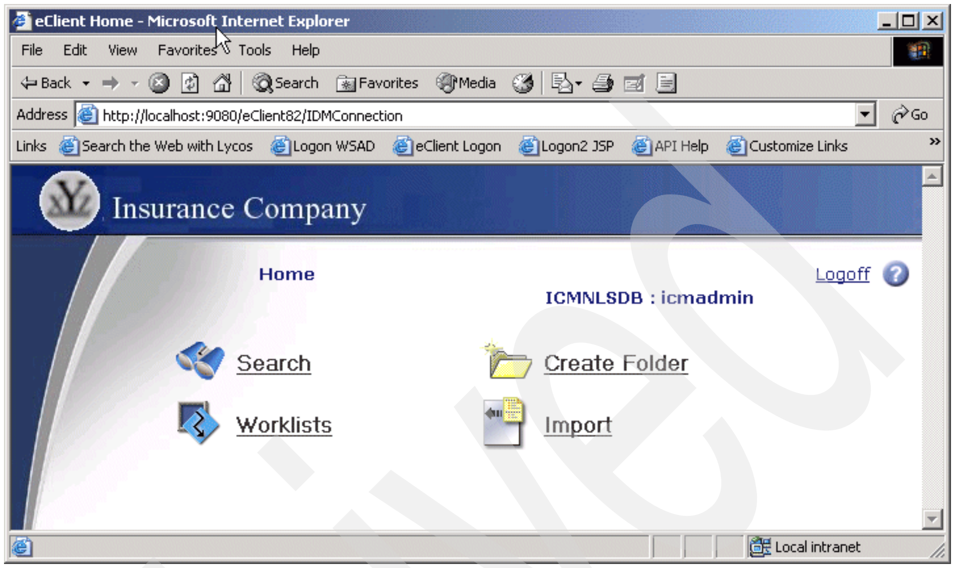
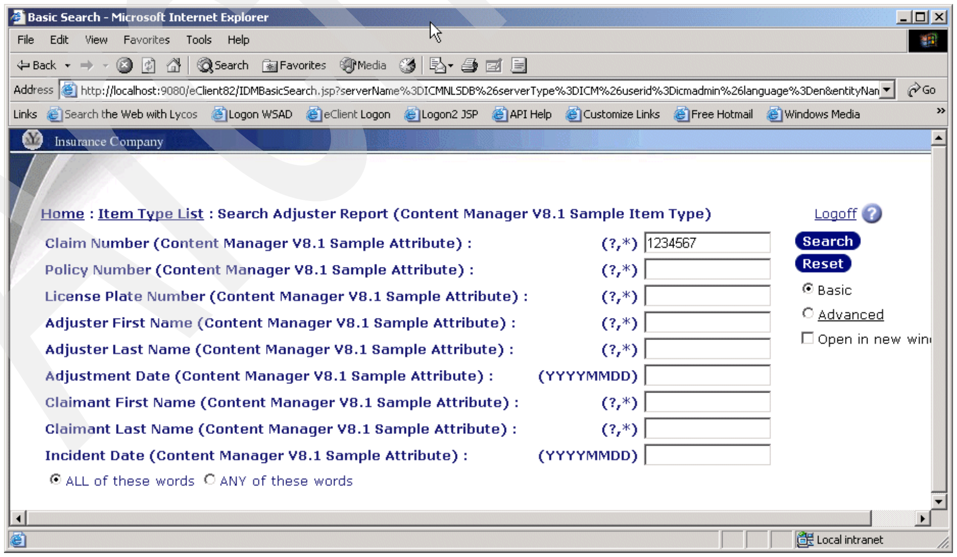

eClient Home - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:9080/eClient82/IDMConnection
Go
Links Search the Web with Lycos Logon WSAD eClient Logon Logon2 JSP API Help Customize Links
Insurance Company
Home
Logoff
ICMNLSDB : icmadmin
Search
Create Folder
Worklists
Import
Local intranet

Figure 9-5 XYZ company home window with customized graphics

Figure 9-6 shows an example of the new eClient search window using the customized XYZ Insurance company logos supplied with the redbook.


Basic Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:9080/eClient82/IDMBasicSearch.jsp?serverName%3DICMNLSDB%26serverType%3DICM%26userid%3Dicmadmin%26language%3Den&entityNan
Go
Links Search the Web with Lycos Logon WSAD eClient Logon Logon2 JSP API Help Customize Links Free Hotmail Windows Media
Insurance Company
Logoff
Search
Reset
Local intranet

Home : Item Type List : Search Adjuster Report (Content Manager V8.1 Sample Item Type)

Claim Number (Content Manager V8.1 Sample Attribute) :	(?,*)	1234567
Policy Number (Content Manager V8.1 Sample Attribute) :	(?,*)	
License Plate Number (Content Manager V8.1 Sample Attribute) :	(?,*)	
Adjuster First Name (Content Manager V8.1 Sample Attribute) :	(?,*)	
Adjuster Last Name (Content Manager V8.1 Sample Attribute) :	(?,*)	
Adjustment Date (Content Manager V8.1 Sample Attribute) :	(YYYYMMDD)	
Claimant First Name (Content Manager V8.1 Sample Attribute) :	(?,*)	
Claimant Last Name (Content Manager V8.1 Sample Attribute) :	(?,*)	
Incident Date (Content Manager V8.1 Sample Attribute) :	(YYYYMMDD)	

[✓ ALL of these words] ANY of these words

[✓ Basic] Advanced

Open in new win

Figure 9-6 XYZ company search window with customized graphics

For instructions on how to download samples provided in this redbook, refer to Appendix B, “Additional material” on page 465 for details.

10

Customizing the edit attributes window

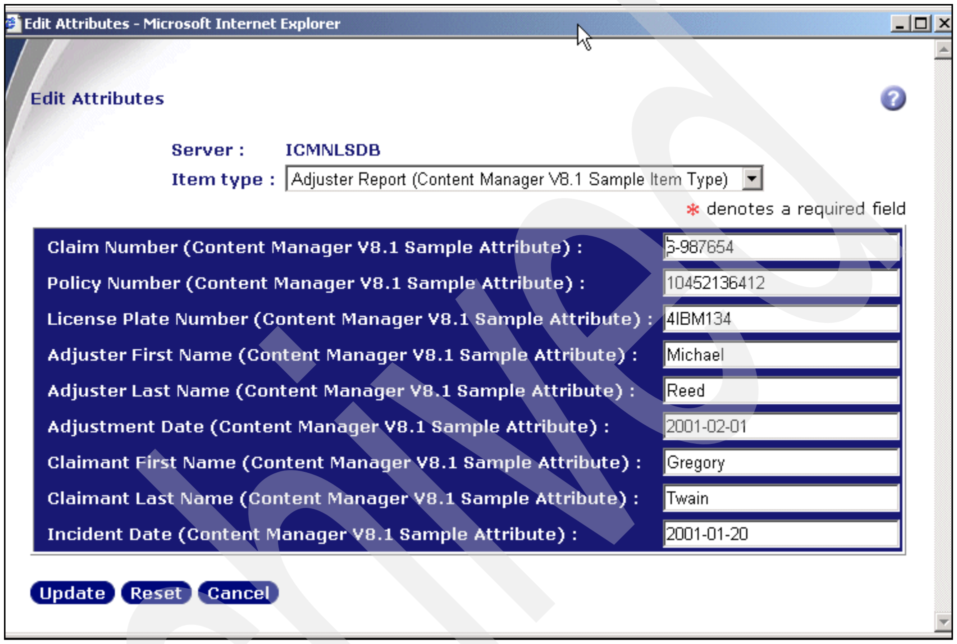
The edit attributes window allows a user to view and modify the metadata associated with a document. In this chapter, we show you how to customize the edit attributes window (IDMEditAttributes.jsp) of eClient through sample codes.

We describe the following customization to the edit attributes window:

- Implementing a combo box for specific attributes
- Making selected attributes read only or hidden
- Modifying the order of the attributes in the window
- Modifying the display name of the item type and attributes
- Re-loading the window when a combo box value is selected

10.1 Overview

The edit attributes window can be invoked from either a search results window or from a document viewer. When invoked, it displays a window such as the one in Figure 10-1 with item type and attributes for a document.


Edit Attributes - Microsoft Internet Explorer

Edit Attributes

Server : ICMNLSDB

Item type : Adjuster Report (Content Manager V8.1 Sample Item Type)

* denotes a required field

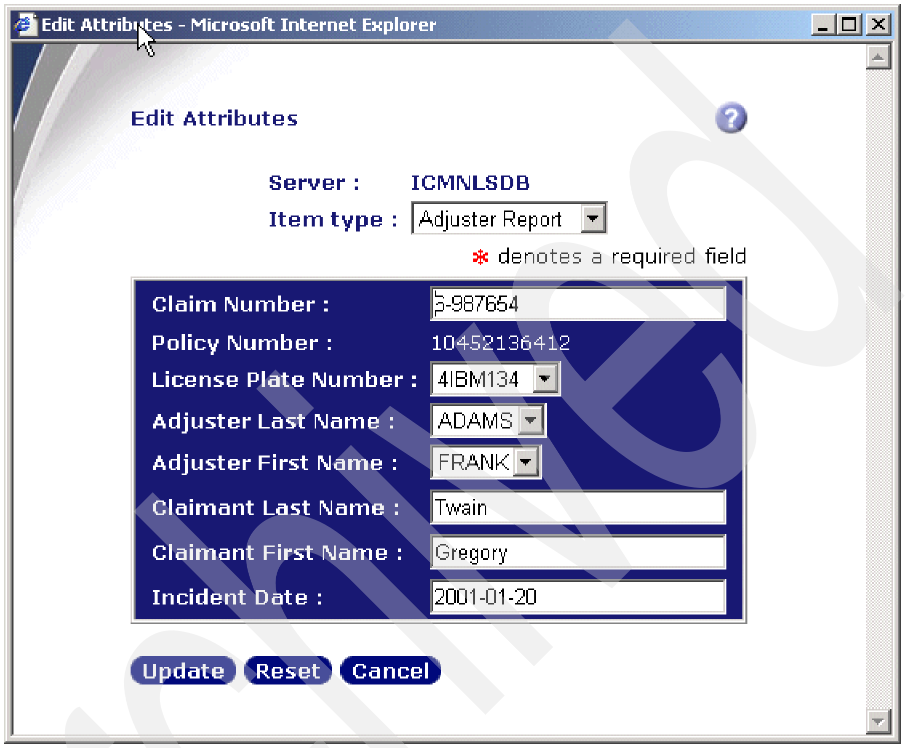
Claim Number (Content Manager V8.1 Sample Attribute) :	5-987654
Policy Number (Content Manager V8.1 Sample Attribute) :	10452136412
License Plate Number (Content Manager V8.1 Sample Attribute) :	4IBM134
Adjuster First Name (Content Manager V8.1 Sample Attribute) :	Michael
Adjuster Last Name (Content Manager V8.1 Sample Attribute) :	Reed
Adjustment Date (Content Manager V8.1 Sample Attribute) :	2001-02-01
Claimant First Name (Content Manager V8.1 Sample Attribute) :	Gregory
Claimant Last Name (Content Manager V8.1 Sample Attribute) :	Twain
Incident Date (Content Manager V8.1 Sample Attribute) :	2001-01-20

Figure 10-1 Standard edit attributes window for Content Manager sample data

The edit attributes window allows a user to change the item type and the attribute values for a document. Note the following:

- ► The native attribute names and values for the back-end datastore are displayed. If you use a federated logon and specify different attribute display names in the EIP entities and search templates, they will not be displayed in this window because eClient gets the attribute names and values directly from the back-end connector.
- ► The edit attributes window does not support index class subsets. Access to specific fields cannot be restricted as they can in the Content Manager thick client.
- ► The attributes are displayed in the order they are defined in the back-end system.

You may want to modify this window to use combo boxes, add field-level security, re-order the fields, and hide some fields. In the following sections, through sample codes, we show you how to make these types of changes to the edit attributes window by customizing IDMEditAttributes.jsp. Figure 10-2 is an example of what the same data looks like when using a customized version of IDMEditAttributes.jsp described in this chapter.


Edit Attributes - Microsoft Internet Explorer

Edit Attributes

Server : ICMNLSDB
Item type : Adjuster Report
* denotes a required field

Claim Number :	5-987654
Policy Number :	10452136412
License Plate Number :	4IBM134
Adjuster Last Name :	ADAMS
Adjuster First Name :	FRANK
Claimant Last Name :	Twain
Claimant First Name :	Gregory
Incident Date :	2001-01-20

Figure 10-2 Sample customized edit attributes window

Figure 10-2 displays the same document as Figure 10-1 on page 226. If you compare the two figures, you see the following differences in the later sample figure:

- The Item type is now replaced with a shorter display name. Everything in parentheses is removed.
- You cannot see it in Figure 10-2, but all of the other Item type values in the combo box are modified to remove the information in parentheses.
- The attribute names are now replaced with shorter display names that do not include the information in the parentheses.
- The Policy Number field is now read-only.
- The License Plate Number field is now converted into a combo box.

- The Adjuster and Claimant’s first and last names are re-ordered so the last name comes first in the list.
- The Adjustment Date field are removed from the display.
- The Adjuster’s first and last names are turned into combo boxes. In addition, when the last name is changed, the list of first names is updated.

The sample code we provide in this chapter includes customization designed to work with the sample data that can be created using the Content Manager First Steps application. The Content Manager sample data includes Item types and sample documents for the fictitious XYZ Insurance company, which maintains policy and claim documents in Content Manager.

Using the Content Manager sample data makes it easier to configure your system to use the sample code, because we do not have to create our own sample data. Some of the sample data item types and attributes for the examples may not always be logical. For example, we demonstrate the use of combo boxes for the insurance adjuster’s last name and first name attributes, which do not represent a real-world scenario; however, it gets across the point of how to make this type of modification while making it easy on you to get up and running with the example.

10.2 Configuring and using customization

There are several components of this customization example. The source files can be downloaded from the Web. See Appendix B, “Additional material” on page 465 for download instructions.

The following files are required for this example:

- IDMEditAttributes.jsp

  The customization uses the user exit feature to keep much of the customer-specific code in Java files instead of in the JSP. By moving as much custom code as possible out of the JSP file, it will be easier to apply your customization to future versions of the eClient.

- XYZEditAttrMethods.java and XYZEditAttrData.java

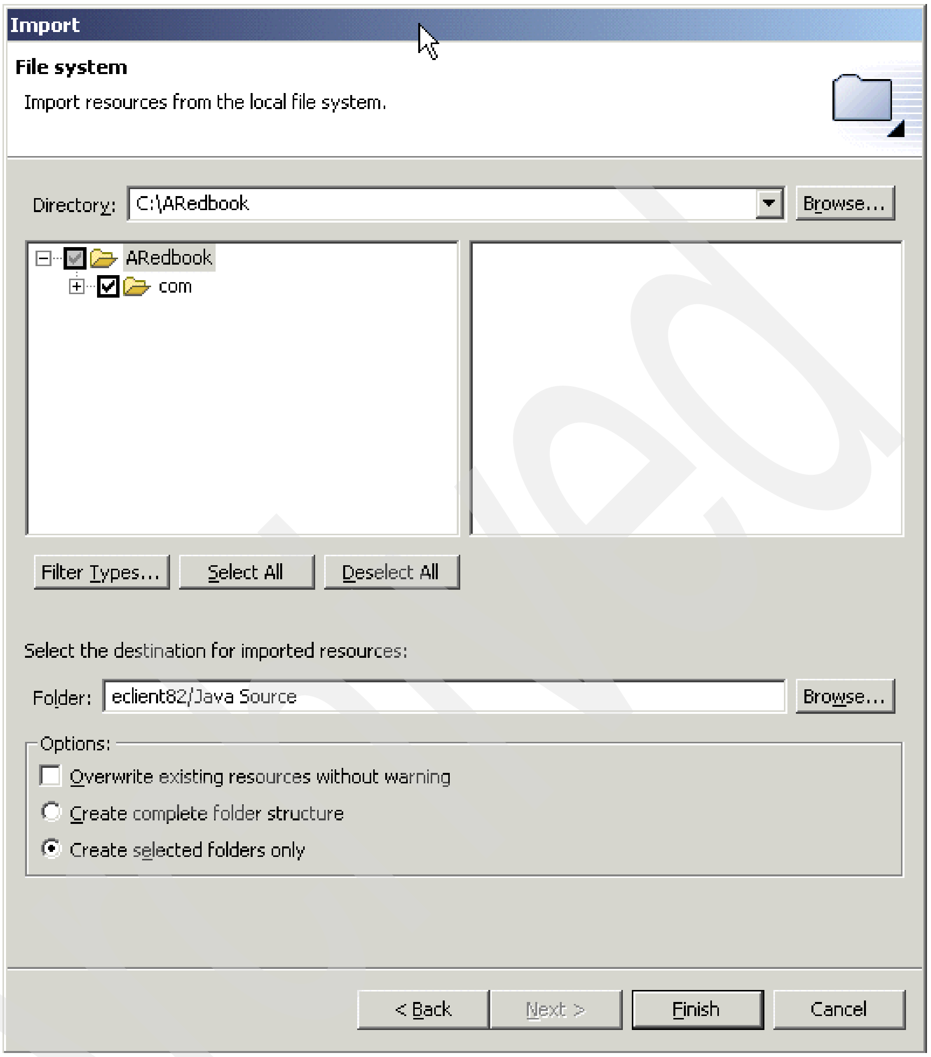
  A majority of the customization is in these Java files. Again, we want to keep the JSP changes as little and as simple as possible. The following methods are implemented in XYZEditAttrMethods.java:

  a. getDisplayName: Displays the name for each item type.

  b. updateAttributeList: For each field being displayed, sets various attributes such as read-only, combo box, or hidden. More details later.

To use the customization on your system, you need to:

1. Meet the prerequisites:
   - Install Content Manager V8.1 or V8.2.
   - Install EIP and eClient V8.2 with the latest Fix Pack.
   - Install the Content Manager sample data using the Content Manager First Steps application.
2. Import the source code and properties files into your environment:
   - Using WebSphere Studio Application Developer:
     1. Open the eClient project in WebSphere Studio Application Developer.
     2. Open the IDMEditAttributes.jsp with the WebSphere Studio Application Developer editor and replace the contents by cutting and pasting the contents of the IDMEditAttributes.jsp file provided with the redbook.
     3. Press Ctrl+S in the WebSphere Studio Application Developer editor to save the file.
     4. Select **File -> Import** from the main menu.
     5. Select **File System** from the list and click **Next**. Figure 10-3 on page 230 appears.
     6. Navigate to the directory containing the com directory with the source files. The com directory should have a subdirectory named XYZUtils, which should contain the Java files.
     7. Browse to or enter `eClient82\Java Source` in the destination field.


Import

File system

Import resources from the local file system.

Directory: C:\ARedbook

- ARedbook
  - [✓] com

Select the destination for imported resources:

Folder: eclient82/Java Source

Options:

- [ ] Overwrite existing resources without warning
- ( ) Create complete folder structure
- (●) Create selected folders only

Figure 10-3 Importing Java files into the eClient project

viii.Click **Finish**. The files are displayed in WebSphere Studio Application Developer as shown in Figure 10-4 on page 231.

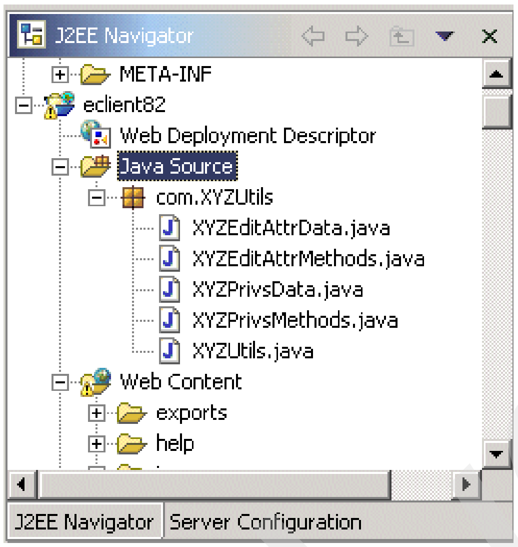


Figure 10-4 Custom Java files in WebSphere Studio Application Developer Navigator view after import

- Using the command line:
  1. Find the eclient82.war directory in the WebSphere directory structure.
  2. Replace the IDMEditAttributes.jsp file to the WebSphere Studio Application Developer Web content directory with the sample file provided with the redbook.
  3. Create directory com\XYZUtils.
  4. Copy XYZEditAttrMethods.java and XYZEditAttrData.java to the new directory.
  5. Open an EIP development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> development window** and compile the files using the **javac** command:

```
javac XYZEditAttrMethods.java
javac XYZEditAttrData.java
```

- Test the changes:

a. Start eClient and log on directly to the Content Manager Version 8 server. Do not use a federated connector.

b. Select **Search**.

c. Select the **Adjuster Report** search template. If you do not see this search template, then the Content Manager Version 8 sample data is not installed correctly.

d. Enter * in the claim number field and click **Search**. You should get three hits in the search results window.

e. Mark the check box for the first document. In our scenario, the claim number is 6-987654.

f. Select **Edit Item Attributes** from the combo box in the search results.

g. It should display the edit attributes window you see in Figure 10-2 on page 227.

10.3 Edit attributes customization overview

The following sections describe how the standard out-of-the-box IDMEditAttributes.jsp file works. Specifically, we cover how it is invoked, how it interacts with the IDMChangeAttributes servlet, and how the Java, HTML, and JavaScript processing in the JSP work.

In addition, we also describe how the changes for the redbook sample are implemented. There were 13 snippets of code that are added to the JSP, along with several methods in a new Java file called XYZEditAttrMethods.java. We cover all of the changes in detail and describe how they can be used together or independently.

10.3.1 Understanding IDMEditAttributes.jsp

This section covers how the standard IDMEditAttributes.jsp works.

Invoking IDMChangeAttributes servlet

The edit attributes window can be invoked in two ways by an eClient user:

1. Invoke the edit attributes window from the search results window.

   The user can mark the check box next to a document in the search results window and select the **Edit attributes** option from the combo box in the search results window.

   This invokes the IDMChangeAttributes servlet from the open_update() JavaScript function in PageComponents\heading.jsp.

2. Invoke the edit attributes window from the standard or the applet viewer.

   The edit attributes window can also be invoked by choosing the edit attributes icon on the toolbar of either viewer.

To invoke the edit attributes window, the eClient invokes the IDMChangeAttributes servlet with the following parameters:

- `srKey=<SRKey`
- `action=display`
- `itemid=<pid string>`

The following code snippet from heading.jsp shows how the eClient invokes the IDMChangeAttributes servlet:

```
var itemId = selectedItems[0];
var url = '<%=webAppName%>/IDMChangeAttributes';
url = url + '?srKey=<%=srKey %>';
url = url + '&action=display';
url = url + '&itemid='+itemId;
wn=generateWindowname();
setWindowname(wn);
window.open( url, wn, 'resizable=1,scrollbars=1,height=300,width=500');
```

The `srKey` is the search results key. It is a string containing the timestamp of the folder that was created for the search results. The JSP and servlet can use it to find out what is in the search results window. `action=display` informs the servlet and JSP that the user is requesting to view the attributes window. The `itemid` is the PID string of the selected document.

What the servlet does

The servlet queries the attributes for the document, the minimum and maximum size of each attribute, whether each attribute is required, the item type for the document, the list of all entities the user has access to, and other information. (Note that *entity* and *item type* are interchangeable. The code often uses entity when referring to the item type of the document.) It then invokes the JSP, which pulls the various data elements from the request. For example, the following lines are used to get the CMBItem object for the document, the entity (item type) for the document, and the ArrayList containing a list of all entities the user has access to:

```
<jsp:useBean id="item" scope="request" class="com.ibm.mm.beans.CMBItem"/>
<jsp:useBean id="entity" scope="request" class="java.lang.String"/>
<jsp:useBean id="entities" scope="request" class="java.util.ArrayList"/>
```

The JSP then goes on to build the HTML for the display of the attributes. This includes the HTML for the display as well as the JavaScript functions that are sent to the browser in the HTML.

Changing item type name

If the user modifies the item type from the edit attributes window's item type list combo box, the JavaScript function reindex is called. This builds a URL to the IDMChangeAttributes servlet with an additional parameter of **'**`?entity=`**"**`<new`

`entity name>`**”**. Control is passed to the servlet, which gets the attribute information for the new entity and calls the JSP again with `action=display`.

Saving changes

If the user makes changes and clicks **Save changes**, the JavaScript validate() function is called to validate user changes. If the validation succeeds, the form is submitted to the IDMChangeAttributes servlet, which then updates the back-end repository.

10.3.2 Modifying IDMEditAttributes.jsp

At a high level, the JSP file is modified in general to support different functional enhancements. The customer-specific logic is all contained in the XYZEditAttrMethods.java class. The JSP file calls two methods in the class to determine how the display should be modified.

The two methods that JSP file calls are:

- getItemTypeDisplayName()

  It passes an item type and returns a display name for each item type. In the example code, this method strips the first left parenthesis and anything after it. For our example, it changes “`Adjuster Report (Content Manager V8.1 Sample Item Type)`” to “`Adjuster Report`.” For your business application, you can modify the method to do other customization.

- updateAttributeList()

  It controls the display of the attributes. When IDMEditAttributes.jsp builds a list of attributes to display, it first builds an ArrayList of XYZEditAttrData structures for each field. See Example 10-1 for the XYZEditAttrData class.

Example 10-1 XYZEditAttrData class

```
package com.XYZUtils;

public class XYZEditAttrData
{
  //
  // This data structure is used by the updateAttributeList() method in
  // XYZEditAttrMethods.java.  That method is called from IDMEditAttributes.jsp
  // while generating the attribute window.
  //
  public String   strAttrNameNative;    // Native attribute name
  public String   strAttrNameDisplay;   // Attribute display name
  public String   strAttrValue;         // Attribute value - Don't change!
  public int      intFieldDataType;     // Field data type

  // Hidden/read only fields
  public int      intHideField;        // Do not display the field
  public int      intReadOnly;         // Set to 1 to make field read-only

  // Combo box related fields
  public int      intComboBox;         // Display combo box for this field?
  public String[] astrComboBoxValues;  // Values for combo box
  // Set to 1 to reload edit attr window when value selected
  public int      intReloadWhenComboValueChanges;
}
```

The JSP builds an ArrayList with one of the structures in Example 10-1 on page 234 for each field to be displayed. It sets the first four fields in the structure before calling the updateAttributeList() method in XYZEditAttrMethods.java. If this method does nothing, then the edit attributes window appears just as it does out of the box. Otherwise, this method can update the ArrayList to make one of many types of changes including the following:

- To hide a field:

  Set `intHideField = 1`

  In our example, the sample code hides the Adjustment Date field.

- To set a field read-only:

  Set `intReadOnly = 1`

  In our example, the sample code sets the Policy Number field to read-only.

- To make a field a combo box:

  Set `intComboBox = 1`

  In our example, the sample code sets `astrComboBoxValues` to a list of values for a combo box.

  The sample code makes VIN, License Number, Adjuster First Name, and Adjuster Last name all combo box fields and provides a list of values for each combo box. The values for the combo boxes are hardcoded in the our example; but this can be replaced with a database lookup in your business application. For example, the Adjuster First Name and Adjuster Last Name can come from an employee database. Also, the VIN number combo box can be loaded from a database that keeps track of VIN numbers of insured under the policy number and the VIN numbers related to claim numbers.

- To re-invoke the JSP file if the combo box value changes:

  Set `intReloadWhenComboValueChanges = 1`

  The sample code uses this for Adjuster Last Name. When the user selects a different Adjuster Last Name from the combo box, it calls the servlet and JSP again so that a corresponding list of the Adjuster First Name values in relation to the selected Adjuster Last Name can be placed in the Adjuster First Name combo box. Again, for our example, these values are hardcoded. You can customize it by looking up values from a corresponding database.

- To re-order the fields:

  Reorder the elements in the ArrayList.

  In our example, the sample code swaps the order of the adjuster and claimant's first and last names.

After returning from updateAttributeList(), the JSP code uses the ArrayList to control how each attribute is displayed. We do not include the entire source code here, since it is too large to embed in this redbook.

To find the changes in the JSP file, search for `XYZ_CHG_BEGIN` in IDMEditAttributes.jsp. Each snippet of the source code is easy to find using this way and the source code is fully documented.

10.3.3 IDMEditAttributes.jsp code change details

There are so many code changes that we cannot include them all in this chapter. This section describes each major piece of code. While reading this section, you should open the following source files in an editor:

- XYZEditAttrData.java
- XYZEditAttrMethods.java
- IDMEditAttributes.jsp

As described in the previous section, XYZEditAttrData.java contains the data structure that is used to control how each field is displayed. IDMEditAttributes.jsp creates an ArrayList of these structures containing the field information retrieved from the back-end system. The ArrayList is then passed to the updateAttributeList() method in XYZEditAttrMethods.java, where it can be modified to specify how each field should be displayed. For example, it can specify whether the field should be hidden, marked as read-only, or should be a combo box.

Each code change in the JSP file is documented with a comment containing the text `XYZ_CHG_BEGIN CHGxx` at the beginning and `XYZ_CHG_END CHGxx` at the end, where `xx` is the change number. You can search for these strings to see the changes, or use a graphical file comparison utility.

There are 13 places where the source code is changed. Some are small changes and others add up to 100 lines of new code. Table 10-1 on page 237 describes the changes.

Table 10-1 List of changes made to IDMEditAttribute.jsp

Change Number	Description
CHG01	Additional import statements for new Java classes.
CHG02	Initialization and debug options.
CHG03	JavaScript validate() function: Gets selected values from combo box field and places them in the right fields before passing control to IDMChangeAttributes servlet.
CHG04	JavaScript reindex() function. This function is called when the user changes the item type. Added `subaction=reindex` to URL so that when the JSP is called again we can tell if it changes the item type.
CHG05	New JavaScript function reload_when_combo_box_changes(). This new function is called when the user modifies a combo box with the intReloadWhencomboValueChanges flag is set in XYZEditAttrData structure. It is similar to reindex and causes the servlet and JSP to be invoked again so that combo box values can be refreshed.
CHG06	New Java code to process new parameters from the URLs created in CHG04 and CHG05.
CHG07	Calls getItemTypeDisplayName and changes the name of the ItemType displayed in the combo box. In the sample, it removes everything after the parenthesis in the item type names.
CHG08	Creates the ArrayList of XYZEditAttrData structures and pass to the updateAttributeList() method, which can then control how each field is displayed.
CHG09	Adds new loop for processing the array of attributes.

Change Number	Description
CHG10	Ensures that inner loop only processes the current field from the new loop coded in CHG09.
CHG11	Adds debug messages to stdout
CHG12	Prevents the code from creating the labels for hidden fields.
CHG13	Creates each field as specified by the call to updateAttributeList in CHG08.
CHG14	Adds closing bracket for new loop level created in CHG09.

The most interesting changes are CHG08 and CHG13.

- CHG08 is where the modification to the user interface is specified via the call to updateAttributeList. This is where the source code specifies that particular fields are read-only, hidden, or combo boxes.
- CHG13 contains the source code that creates the fields of the right type: hidden, read-only, or a combo box.

CHG05 is also important because it allows the form to be reloaded with different combo box values when a specific combo box value is modified.

Refer to the documentation in the source code for additional details.

11

Adding custom functions to the search results window

The eClient search results window allows users to select one or more documents from a set of search results and then select an action to perform on the selected document(s). Actions such as view, delete, edit attributes are available in the standard eClient.

If you need to customize these actions, this chapter provides examples of how to implement custom actions in the eClient search results window and discusses the following topics:

- A list of scenarios where this type of functionality could be used
- A description of how to modify the eClient to implement these functions
- Three examples:
  - Exporting one or more documents to the client workstation by sending them to the browser to be saved
  - Displaying TIFF documents using a TIFF plug-in viewer
  - Performing a custom process against a list of selected documents

11.1 Overview

This chapter describes how to implement custom functions in the search results window. It describes the JSPs that must be modified and/or created, and provides sample code that can be used to implement your own custom functions.

Figure 11-1 is a typical search results window of the eClient.

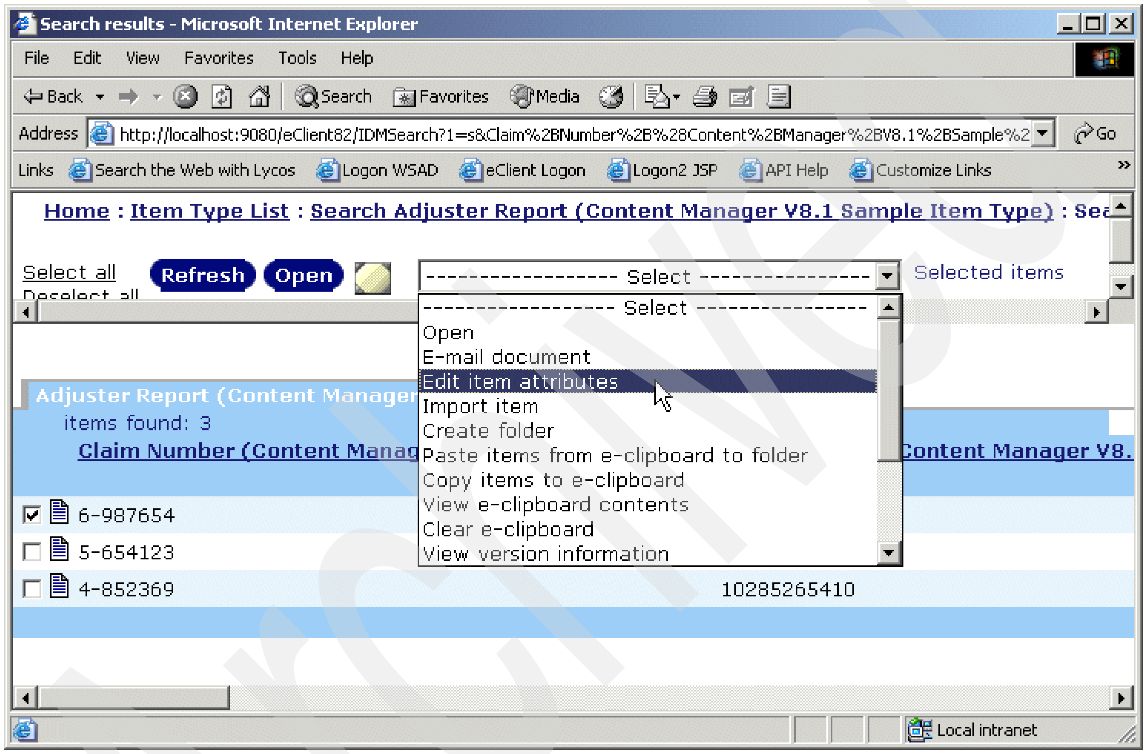

Search results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:9080/eClient82/IDMSearch?1=s&Claim%2BNumber%2B%28Content%2BManager%2BV8.1%2BSample%2
Go
Links Search the Web with Lycos Logon WSAD eClient Logon Logon2 JSP API Help Customize Links
Home : Item Type List : Search Adjuster Report (Content Manager V8.1 Sample Item Type) : Sea
Select all
Deselect all
Refresh
Open
------------------ Select -----------------
Selected items
------------------ Select -----------------
Open
E-mail document
Edit item attributes
Import item
Create folder
Paste items from e-clipboard to folder
Copy items to e-clipboard
View e-clipboard contents
Clear e-clipboard
View version information
Adjuster Report (Content Manager
items found: 3
Claim Number (Content Manag
Content Manager V8.
6-987654
5-654123
4-852369
10285265410
Local intranet

Figure 11-1 eClient search results window

You can select one or more documents (by marking the check box next to the document) and then select an action from the drop-down combo box to be performed on the selected document(s). Some options, such as Edit item attributes, can operate on only one selected document. Other options, such as Delete items, can operate on multiple documents.

The following are some examples of custom actions you can create within the drop-down combo box:

- Perform an automatic update of document attributes

  For example, you can implement the Approve documents option that sets a document attribute status to Approved and set the ApprovedBy attribute to the current user. The updates can occur in the back-end datastore or on another database.

- Look up data from an external system

  You can implement a Get billing summary option that gets an attribute from the selected document (such as customer number and policy number), queries an external database for an account summary, and displays the text output in a new window.

- Perform an operation on the document object

  You can implement a Convert to text option that performs full-text OCR of a TIFF file on the fly and displays the text in a new window.

- Perform automatic foldering or routing

  You can implement a Review document function that automatically places a document in a specific folder, or starts it in a specific workflow.

- Perform automatic export of the document object and attribute data

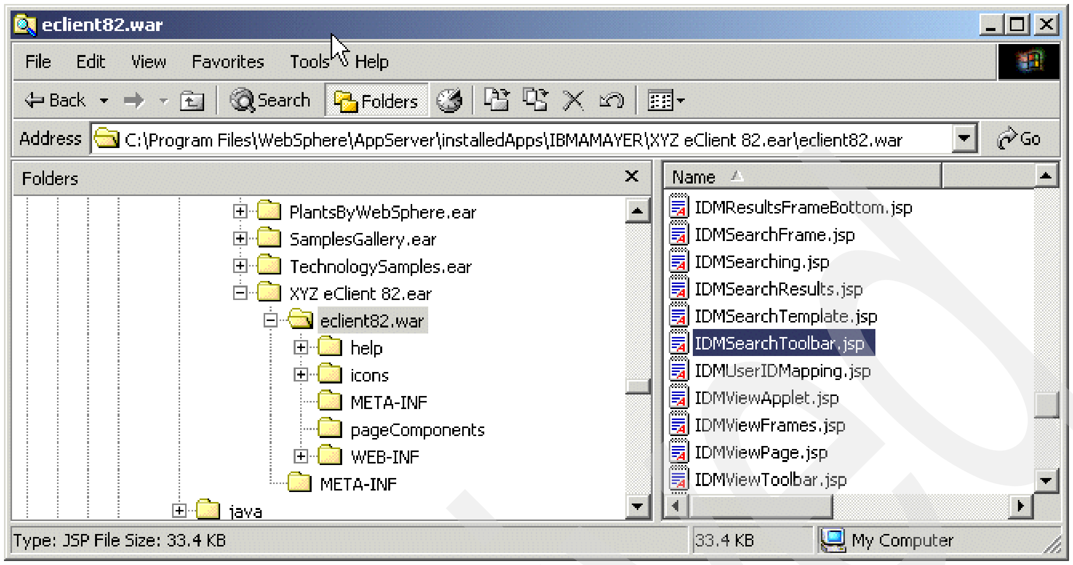
  You can automatically export the document object and attribute data to a server directory. This prepares the selected documents to be written to a CD or imported to another system.

The examples in this chapter provide a framework for implementing a variety of custom functions. Two of the examples (export and view with plug-in) open a new browser window for each selected document. The third example opens a single window that allows a user to specify options for the custom action and then perform custom processing for each selected document.

11.2 Adding custom entries to combo box

There are two JSPs that need to be modified to add a custom function to the search results combo box: IDMSearchToolbar.jsp and heading.jsp. In addition, you probably need to create a new JSP and/or servlet to perform the custom processing on the selected items. The sample code provided does all of the processing in JSPs and does not use custom servlets.

IDMSearchToolbar.jsp is in the eclient82.war directory in WebSphere as shown in Figure 11-2 on page 242. Note that the heading.jsp is in the pageComponents subdirectory below the rest of the eClient JSPs.



Address: C:\Program Files\WebSphere\AppServer\installedApps\IBMAMAYER\XYZ eClient 82.ear\eclient82.war

```
PlantsByWebSphere.ear
SamplesGallery.ear
TechnologySamples.ear
XYZ eClient 82.ear
    eclient82.war
        help
        icons
        META-INF
        pageComponents
        WEB-INF
    META-INF
java
```

```
IDMResultsFrameBottom.jsp
IDMSearchFrame.jsp
IDMSearching.jsp
IDMSearchResults.jsp
IDMSearchTemplate.jsp
IDMSearchToolbar.jsp
IDMUserIDMapping.jsp
IDMViewApplet.jsp
IDMViewFrames.jsp
IDMViewPage.jsp
IDMViewToolbar.jsp
```

Type: JSP File Size: 33.4 KB | 33.4 KB | My Computer

Figure 11-2 eClient source code location

IDMSearchToolbar.jsp contains the code that generates the list of items in the combo box. It also contains the JavaScript function doSelected(), which is executed when a user selects an item in the combo box. The doSelected() function calls the appropriate JavaScript function in heading.jsp (one for each action) to determine which documents are selected, and then performs the selected action. The examples later in this chapter show you how to make these changes.

The sample code shows you how to do two types of processing. In one case, a new browser window is opened for each document. The XYZ export document and XYZ view document with plug-in work this way. The other type of processing opens a single JSP window that can perform an action on all of the selected items. For example, it can add all of the items to a folder, update an attribute in each document, or do whatever custom function you need using Java or the Java beans.

Note that these examples are created to show concepts. They are not extensively tested or intended to be used in production as-is.

11.3 Installing sample code

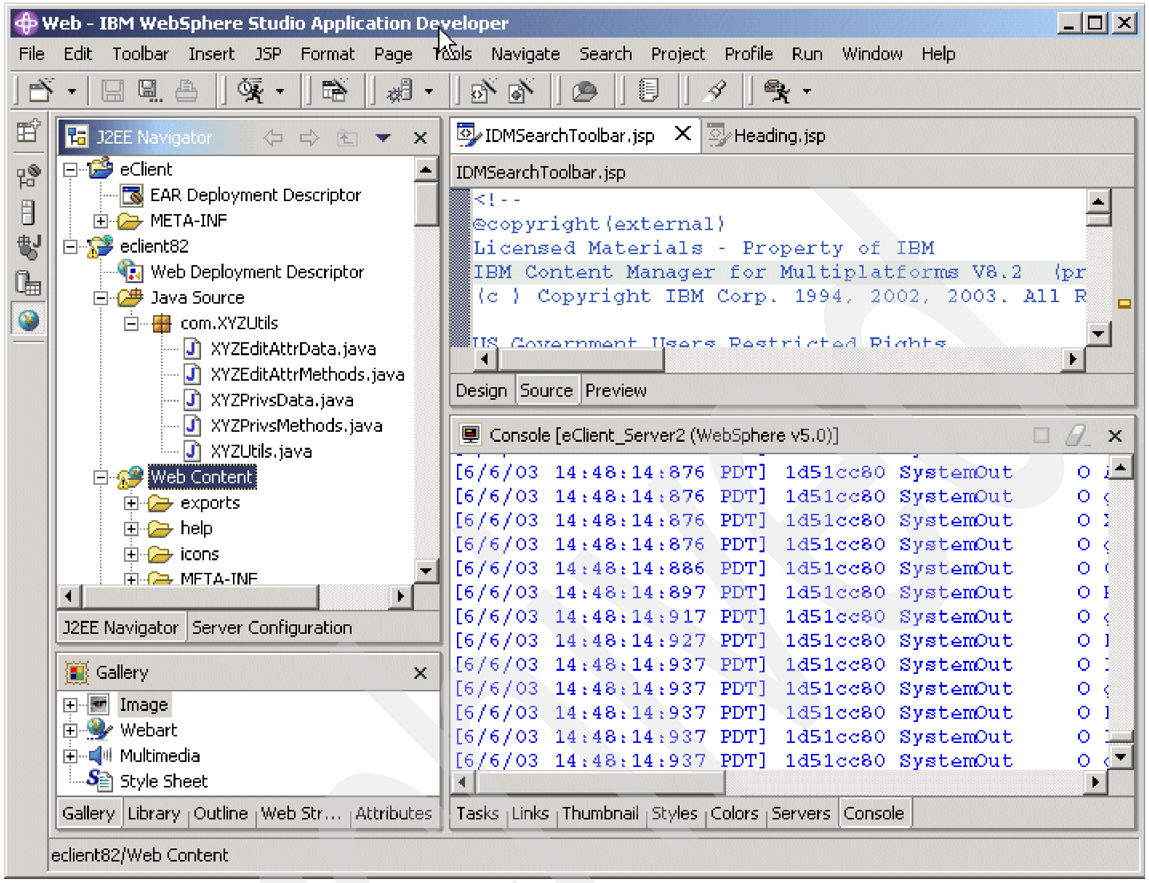
To test the sample code on your workstation you need to install updates to two JSPs and add a new JSP to the eClient.

Follow these steps to install the sample code using the command line:

1. Copy the new IDMSearchToolbar.jsp from the redbook samples over the standard eClient 8.2 JSP in the eclient82.war directory in the WebSphere installedApps directory.
2. Copy the new heading.jsp over the standard eClient 8.2 JSP in the eclient82.war\pageComponents directory in the WebSphere installedApps directory.
3. Copy the new YXZProcessDocuments.jsp and IDMProcessDocumentList.jsp to the eclient82.war directory in the WebSphere installedApps directory.
4. Create directory named exports below the eclient82.war directory. Temporary document files will be stored here.

Follow these steps to install the sample code using WebSphere Studio Application Developer:

1. Open the eClient 8.2 IDMSearchToolbar.jsp and heading.jsp with the WebSphere Studio Application Developer editor.
2. Open the IDMSearchToolbar.jsp and heading.jsp custom files provided with the redbook samples in Notepad or another editor. Copy and paste the new code into the WebSphere Studio Application Developer editor for both IDMSearchToolbar.jsp and heading.jsp. Press Ctrl+S in the WebSphere Studio Application Developer editor for each file to save the changes.
3. Import new JSPs into eClient as follows:
   a. Open the J2EE Navigator view in WebSphere Studio Application Developer and expand **eclient82 -> Web Content** as shown in Figure 11-3 on page 244.

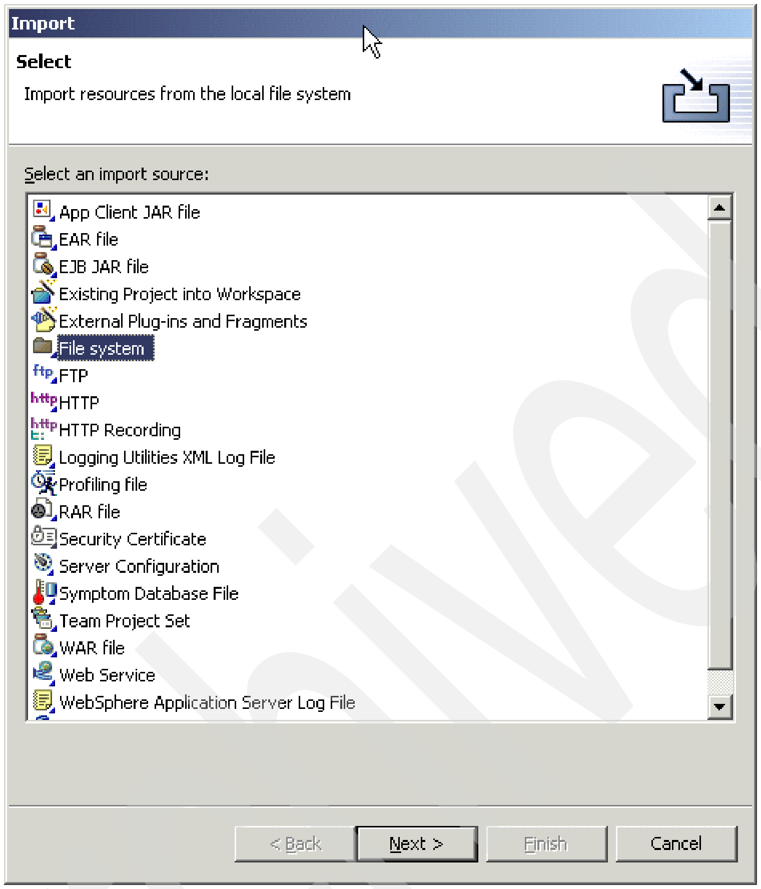


```
<!--
@copyright(external)
Licensed Materials - Property of IBM
IBM Content Manager for Multiplatforms V8.2  (pr
(c ) Copyright IBM Corp. 1994, 2002, 2003. All R

US Government Users Restricted Rights
```

```
[6/6/03 14:48:14:876 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:876 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:876 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:876 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:886 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:897 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:917 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:927 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
```

Figure 11-3 Web content in WebSphere Studio Application Developer J2EE Navigator window

b. From the menu, select **File -> Import...** and a window similar to Figure 11-4 on page 245 appears.


Import

Select

Import resources from the local file system

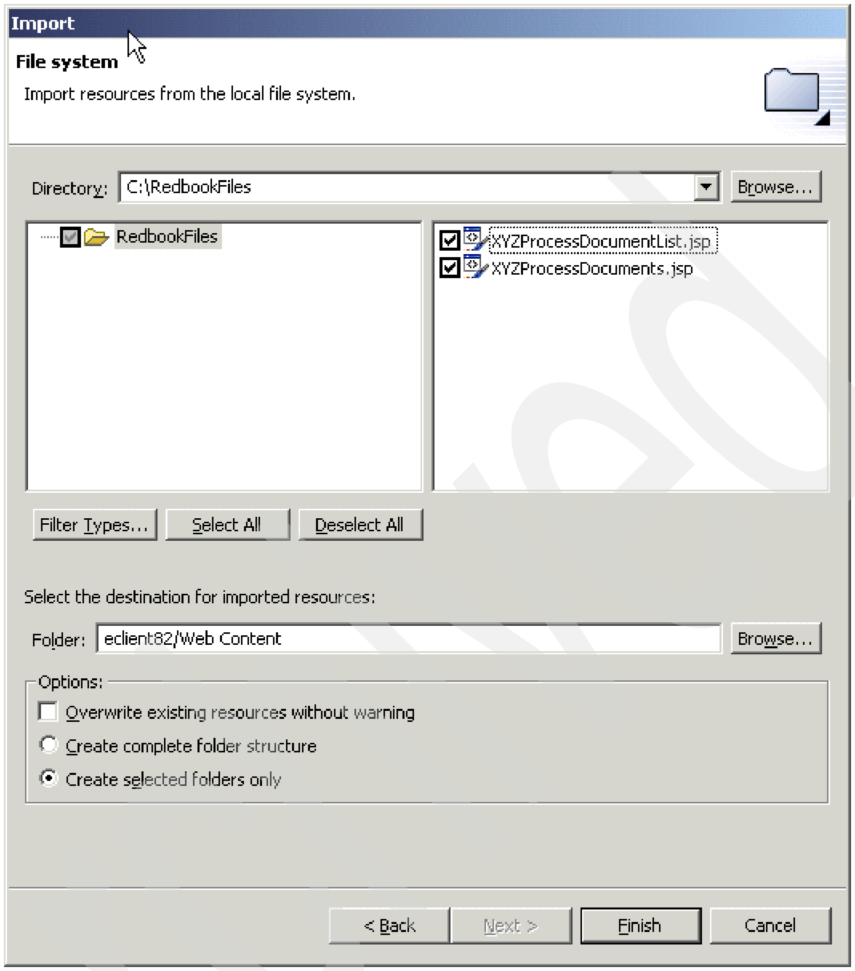
Select an import source:

- App Client JAR file
- EAR file
- EJB JAR file
- Existing Project into Workspace
- External Plug-ins and Fragments
- File system
- FTP
- HTTP
- HTTP Recording
- Logging Utilities XML Log File
- Profiling file
- RAR file
- Security Certificate
- Server Configuration
- Symptom Database File
- Team Project Set
- WAR file
- Web Service
- WebSphere Application Server Log File

Figure 11-4 WebSphere Studio Application Developer file import window

c. Select **File system** and click **Next**.

d. Browse to the directory where the source code is located. Select the files you want to import into your eClient project. See Figure 11-5 on page 246.


Import

File system

Import resources from the local file system.

Directory: C:\RedbookFiles

- [✓] RedbookFiles
- [✓] XYZProcessDocumentList.jsp
- [✓] XYZProcessDocuments.jsp

Select the destination for imported resources:

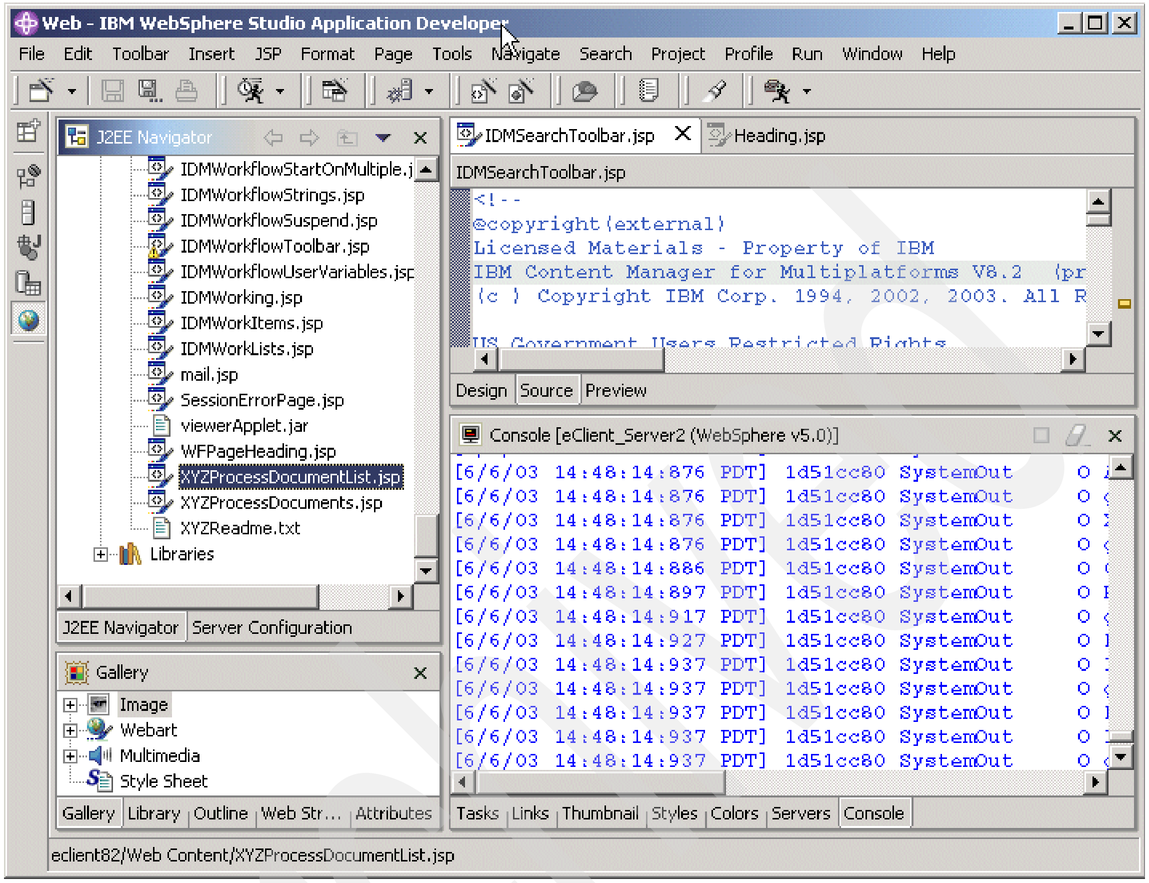
Folder: eclient82/Web Content

Options:
- [ ] Overwrite existing resources without warning
- ( ) Create complete folder structure
- (●) Create selected folders only

Figure 11-5 WebSphere Studio Application Developer import file selection window

e. After you select all the files to import, click **Finish**.

f. Go back to the J2EE Navigator window. Scroll down to the bottom and you should see the two files that are imported. See Figure 11-6 on page 247.


Web - IBM WebSphere Studio Application Developer
File Edit Toolbar Insert JSP Format Page Tools Navigate Search Project Profile Run Window Help
J2EE Navigator
IDMWorkflowStartOnMultiple.j
IDMWorkflowStrings.jsp
IDMWorkflowSuspend.jsp
IDMWorkflowToolbar.jsp
IDMWorkflowUserVariables.jsp
IDMWorking.jsp
IDMWorkItems.jsp
IDMWorkLists.jsp
mail.jsp
SessionErrorPage.jsp
viewerApplet.jar
WFPageHeading.jsp
XYZProcessDocumentList.jsp
XYZProcessDocuments.jsp
XYZReadme.txt
Libraries
J2EE Navigator
Server Configuration
Gallery
Image
Webart
Multimedia
Style Sheet
Gallery Library Outline Web Str... Attributes
IDMSearchToolbar.jsp
Heading.jsp
IDMSearchToolbar.jsp
Design Source Preview
Console [eClient_Server2 (WebSphere v5.0)]
Tasks Links Thumbnail Styles Colors Servers Console
eclient82/Web Content/XYZProcessDocumentList.jsp

```
<!--
@copyright(external)
Licensed Materials - Property of IBM
IBM Content Manager for Multiplatforms V8.2  (pr
(c ) Copyright IBM Corp. 1994, 2002, 2003. All R

US Government Users Restricted Rights
```

```
[6/6/03 14:48:14:876 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:876 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:876 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:876 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:886 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:897 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:917 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:927 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
[6/6/03 14:48:14:937 PDT] 1d51cc80 SystemOut     O
```

Figure 11-6 WebSphere Studio Application Developer J2EE Navigator view with new files

4. After installing the new Java files or whenever any Java files are modified, restart your WebSphere test environment.

11.4 Adding your own custom function

This section describes how to change the JSP file to implement the custom functions mentioned in earlier section. You can use them as guidelines for implementing your own custom modification.

Do the following to implement the custom functions:

1. Add combo entries in IDMSearchToolbar.JSP.

   To add new options to the combo box, add the following source code to the HTML in IDMSearchToolbar.jsp:

```
<!-- XYZ_CHG_BEGIN -->
<option value="XYZExportDocs">XYZ Export Document</option>
<option value="XYZViewWithPlugin">XYZ View Document with Plugin</option>
<!-- XYZ_CHG_END -->
```

2. Update the function doSelected() in IDMSearchToolbar.jsp.

   When the user selects something from the combo box, the JavaScript function doSelected() is called. The docselValue is the text in the value tag of the option statements shown above. Update the source code as follows:

```
function doSelected(docselValue) {
   switch (docselValue) {
      case 'EditAttributes' : // Edit Document Attributes
         parent.ResultsBottom.openUpdate()
         break;
      // XYZ_BEGIN_CHG
      case 'XYZExportDocs':
         //
         // Call the JavaScript function in heading.jsp
         //
         parent.ResultsBottom.XYZProcessDocuments("export");
         break;
      case 'XYZViewWithPlugin':
         //
         // Call the JavaScript function in heading.jsp
         //
         parent.ResultsBottom.XYZProcessDocuments(
                      "display_with_plugin");
         break;
      // XYZ_CHG_END
      case 'Email':   // View Version Info
         parent.ResultsBottom.emailSelected()
         break;
```

3. Add the new JavaScript function in heading.jsp to process the selected documents.

The source file heading.jsp primarily contains a list of JavaScript functions that run in a browser as users interact with the search results window. When a user selects an option in the search results combo box, JavaScript in IDMSearchToolbar.jsp calls a JavaScript function in heading.jsp to perform the action.

For the sample code, the code that exports the document file to disk on the Web server is nearly identical to the code for displaying a document with a TIFF plug-in viewer. The same JavaScript function, XYZProcessDocuments(), is used for both functions and the JavaScript passes a parameter `action` indicating if the document should be viewed (display_with_plugin) or exported (export).

11.5 Using sample code

The provided sample code adds three custom options to the search results combo box (see Figure 11-7 on page 250):

- XYZ Export Document

  This option saves the document object to the Web server, sends a link of the file to the user's browser, prompts the user where to save the file, and exports (saves) the file.

- XYZ View Document with Plugin

  This option displays the document using the AlternaTIFF TIFF plug-in viewer. This TIFF plug-in viewer can be downloaded for free.

- XYZ List Selected Docs

  This option provides a framework that can be used to perform a custom action on one or more selected documents. The other two samples open a new window for each selected document, while this example executes a single JSP that can act on a list of all selected documents.

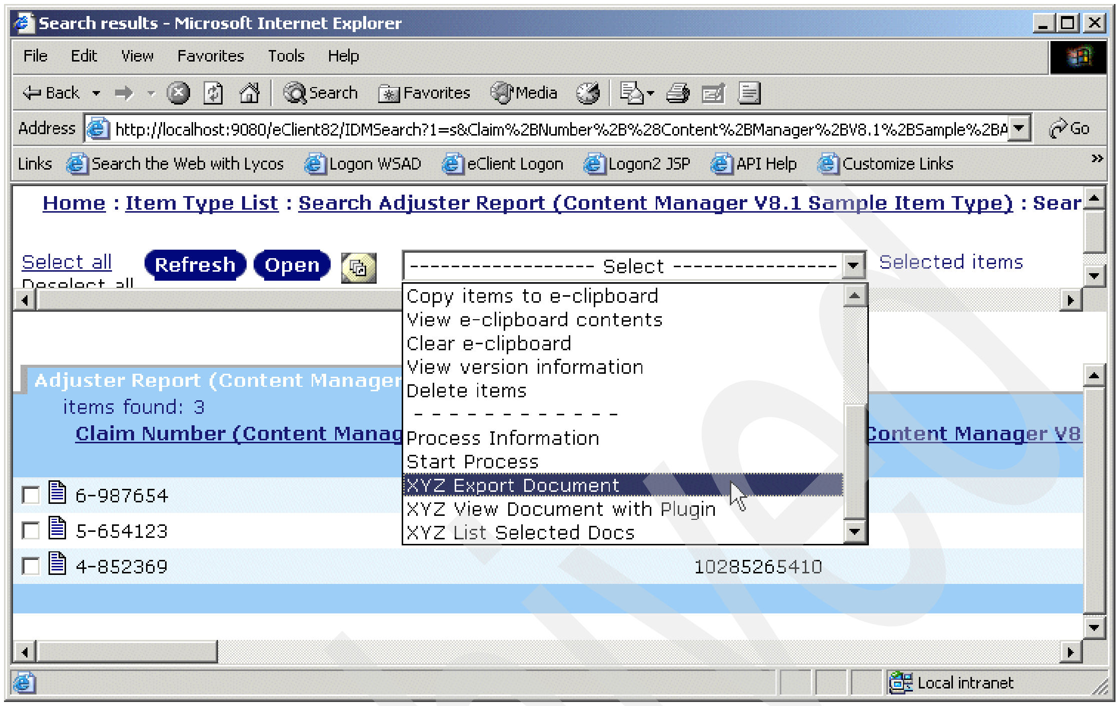
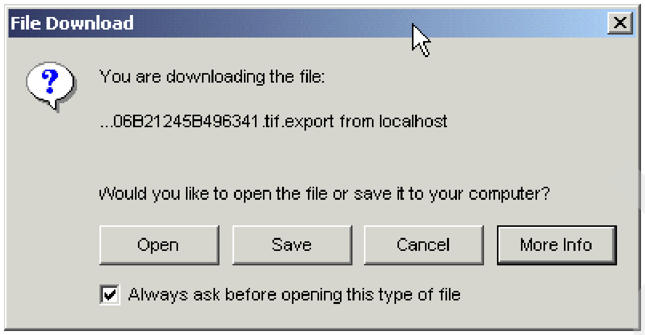

Search results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:9080/eClient82/IDMSearch?1=s&Claim%2BNumber%2B%28Content%2BManager%2BV8.1%2BSample%2BA
Go
Links Search the Web with Lycos Logon WSAD eClient Logon Logon2 JSP API Help Customize Links
Home : Item Type List : Search Adjuster Report (Content Manager V8.1 Sample Item Type) : Sear
Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items
Copy items to e-clipboard
View e-clipboard contents
Clear e-clipboard
View version information
Delete items
- - - - - - - - - - - -
Process Information
Start Process
XYZ Export Document
XYZ View Document with Plugin
XYZ List Selected Docs
Adjuster Report (Content Manager
items found: 3
Claim Number (Content Manag
Content Manager V8
6-987654
5-654123
4-852369
10285265410
Local intranet

Figure 11-7 Custom functions in search results combo box

The rest of this section describes in detail the implementation of the three custom functions.

11.5.1 Export documents option

Users can select one or more documents and choose the XYZ Export Document option. When this option is selected, a window similar to Figure 11-8 on page 251 is displayed for each document, giving the user the option to store the file to their local drive or a network drive.


File Download

You are downloading the file:

...06B21245B496341.tif.export from localhost

Would you like to open the file or save it to your computer?

☑ Always ask before opening this type of file

Figure 11-8   XYZ export documents dialog box

The following is a description of what happens when a user selects the **XYZ Export Documents** option:

1. The doSelected() JavaScript function in IDMSearchToolbar.jsp is called. It is implemented with the following code snippet:

```
case 'XYZExportDocs':
   //
   // Call the JavaScript function in heading.jsp.  This will open
   // a window for each document and allow the user to save the
   // object file to their hard drive.
   //
   parent.ResultsBottom.XYZProcessDocuments("export");
   break;
```

2. The code above calls XYZProcessDocuments() in heading.jsp. XYZProcessDocuments() determine which documents are selected and invokes the XYZProcessDocuments.jsp file for each document with an action keyword in the URL set to export. See Example 11-1 for the XYZProcessDocuments() code snippet.

Example 11-1   XYZProcessDocuments() code snippet

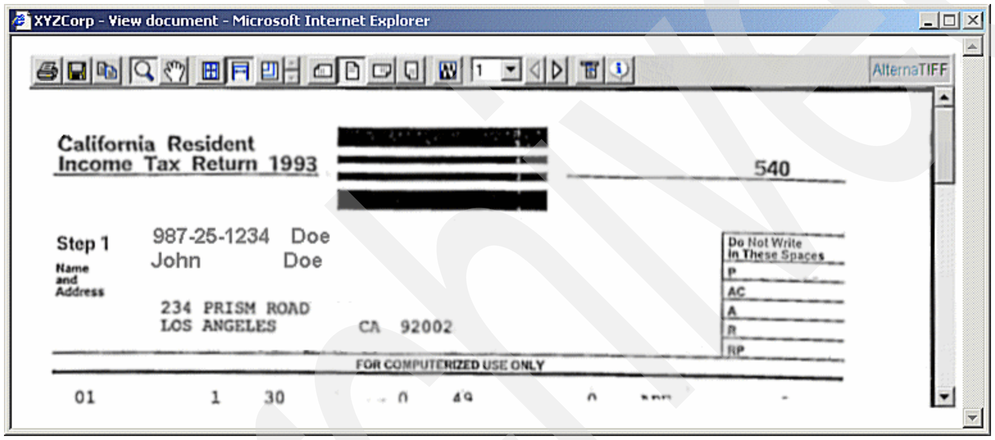
```
function XYZProcessDocuments(strAction)
{
   //alert("In XYZProcessDocuments()");

   // If no items in search results window
   if (count == 0)
      return;

   //
   // If more than one item in the search results window
   //
   if (count > 1)
   {
      var selected = false
      //
      // Loop through each item in the search results window
      //
      for (i = 0; i < document.results.check1.length; i++)
      {
         // If the document is selected
         if (document.results.check1[i].checked == true)
         {
            //
            // Open the document or folder
            //
            var docURL = document.results.check1[i].value
            selected = true;
            if (docURL.indexOf("folder=") == -1)
            {
               //
               // Invoke the XYZProcessDocuments.jsp with the following 
parameters:
               //    srKey  - Handle to the search results
               //    action - display_with_plugin or export
               //    itemId - Itemid (PID string) of document
               wn=generateWindowname()+i;
               setWindowname(wn);
               var itemId = getItemId(docURL);

               var docURL = '<%= webAppName 
%>/XYZProcessDocuments.jsp?srKey=<%= srKey %>&action=' + strAction + 
'&itemid='+itemId;

               alert("url is " + docURL);
               window.open( docURL, wn, 'resizable=1,scrollbars=1');
            }
            else
            {
               openFolder(docURL);
               break;
            }
         }
      }
      //
      // If no items selected, display message
      //
      if (selected == false)
      {
         alert("You must select at least one document");
      }
   }
   else if (count == 1)
   {
      if (document.results.check1.checked == true)
      {
         var docURL = document.results.check1.value;
         if (docURL.indexOf("folder=") == -1)
         {
            //
            // Invoke the XYZProcessDocuments.jsp with the following 
parameters:
            //    srKey  - Handle to the search results
            //    action - display or export
            //    itemId - Itemid of document to export or display
            wn=generateWindowname()+i;
            setWindowname(wn);
            var itemId = getItemId(docURL);
            var docURL = '<%= webAppName %>/XYZProcessDocuments.jsp?srKey=<%= 
srKey %>&action=' + strAction + '&itemid='+itemId;

            window.open( docURL, wn, 'resizable=1,scrollbars=1');
         }
         else
         {
            openFolder(docURL);
         }
      }
   }
}
// XYZ_CHG_END
```

3. At this point, the XYZProcessDocuments.jsp gets the control. The source code is too large to include in the redbook, but it is documented and provided as part of the redbook sample. Take note of the following:
   - The strAction field receives an action: either export or display_with_plugin. The same JSP is invoked for the first two examples in this chapter, but they use a different action parameter to call the XYZProcessDocuments.jsp.
   - The strItemID field receives the PID string of the document to process.
   - Class XYZ_Export_Utils has a function putPartInFile() that stores the object for a PID string to a file on the Web server. You need to modify the location of the exported file in this function to a location on the CLASSPATH in your system environment.

Tip: If you are not making too many changes, it may be faster to develop your Java code in the JSP file and then move it to a Java file when it is done. With WebSphere Studio Application Developer, you have to restart your WebSphere test environment every time that the Java files are modified, but with the code in your JSP, you can save the JSP and re-test.

11.5.2 View document with plug-in option

The second option provided allows you to view a document using a TIFF plug-in viewer instead of the viewers that shipped with the eClient. When you select the **XYZ View Document with Plugin** option, a window similar to Figure 11-9 appears.



California Resident
Income Tax Return 1993

540

Step 1 987-25-1234 Doe
Name and Address John Doe

234 PRISM ROAD
LOS ANGELES CA 92002

Do Not Write In These Spaces
P
AC
A
R
RP

FOR COMPUTERIZED USE ONLY

01 1 30

Figure 11-9 Sample document display with the AlternaTIFF plug-in viewer

If you are using Microsoft Internet Explorer, the AlternaTIFF plug-in will be automatically downloaded the first time you view a document with this viewer.

If you are using Netscape, you must download and install the AlternaTIFF plug-in first prior to viewing any document with the viewer. Refer to instructions on downloading the plug-in at `http://www.alternatif.com`. You need to download alternatif-1_5_5.exe and run it to configure the plug-in for NetScape.

Note that the AlternaTIFF plug-in does not support documents compressed with the LZW algorithm, which is used by the Content Manager First Steps sample data. In order to use the plug-in, you need to import some TIFF files that do not use LZW compression.

11.6 Process selected documents option

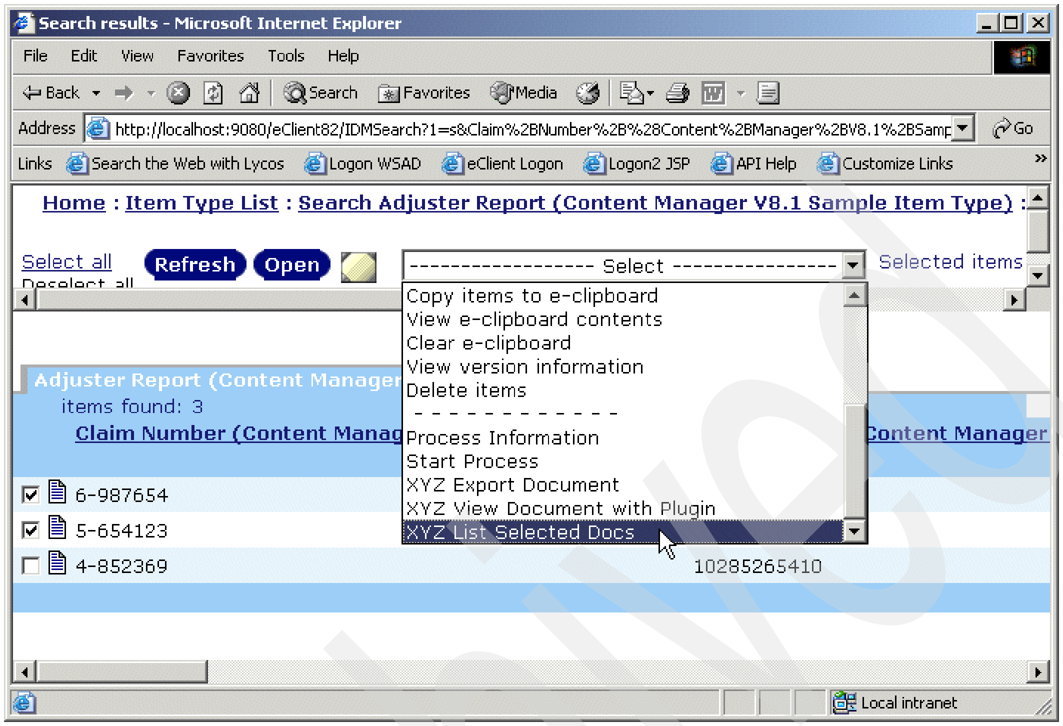
This example provides a framework to show you how to implement a custom function in the eClient search results window that performs an action on one or more selected documents.

The custom action, for example, can include updating a specific attribute of each document to a specific value, adding each document to a workbasket, and querying an external database via Java code to look up external data for each document.

The customization consists of three parts:

1. Modify the eClient IDMSearchToolbar.jsp to add the custom function XYZ List Selected Docs to the search results action combo box.
2. Modify the eClient heading.jsp to invoke a custom JSP when a user selects the new action from the search results window.
3. Add a new custom JSP, XYZProcessDocumentList.jsp, which enables you to gather additional parameters for your custom action, performs the custom function, and displays the results to the user.

When a user selects multiple documents in the eClient search results window and then selects the **XYZ List selected documents** option (see Figure 11-10 on page 256), the sample code displays a list of PID strings that are selected from the search results window. You can write custom Java code to perform more meaningful action for each document.


Search results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:9080/eClient82/IDMSearch?1=s&Claim%2BNumber%2B%28Content%2BManager%2BV8.1%2BSamp
Go
Links Search the Web with Lycos Logon WSAD eClient Logon Logon2 JSP API Help Customize Links
Local intranet

Home : Item Type List : Search Adjuster Report (Content Manager V8.1 Sample Item Type) :

Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items

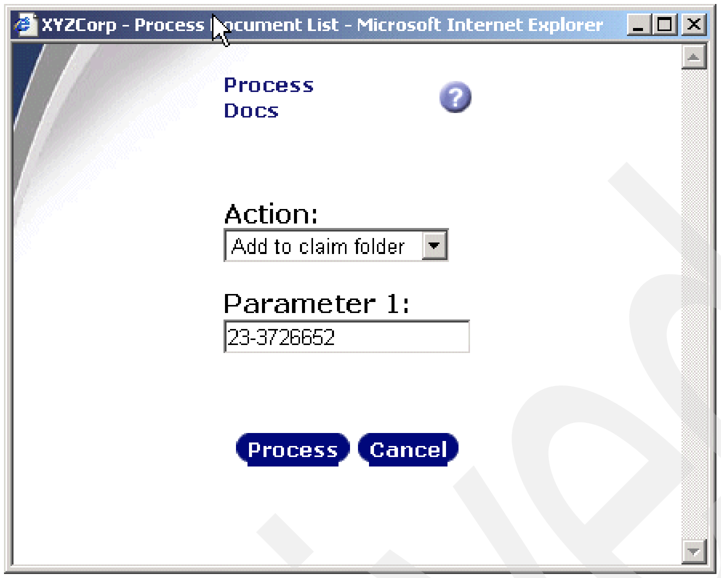
Copy items to e-clipboard
View e-clipboard contents
Clear e-clipboard
View version information
Delete items
- - - - - - - - - - - -
Process Information
Start Process
XYZ Export Document
XYZ View Document with Plugin
XYZ List Selected Docs

Adjuster Report (Content Manager
items found: 3
Claim Number (Content Manag
Content Manager

☑	6-987654	
☑	5-654123	
☐	4-852369	10285265410

Figure 11-10 Selecting custom function from search results

The XYZ List selected documents option invokes the XYZProcessDocumentList.jsp with an action of display. A window similar to Figure 11-11 on page 257 appears.


XYZCorp - Process Document List - Microsoft Internet Explorer
Process Docs
Process
Cancel

Action: Add to claim folder

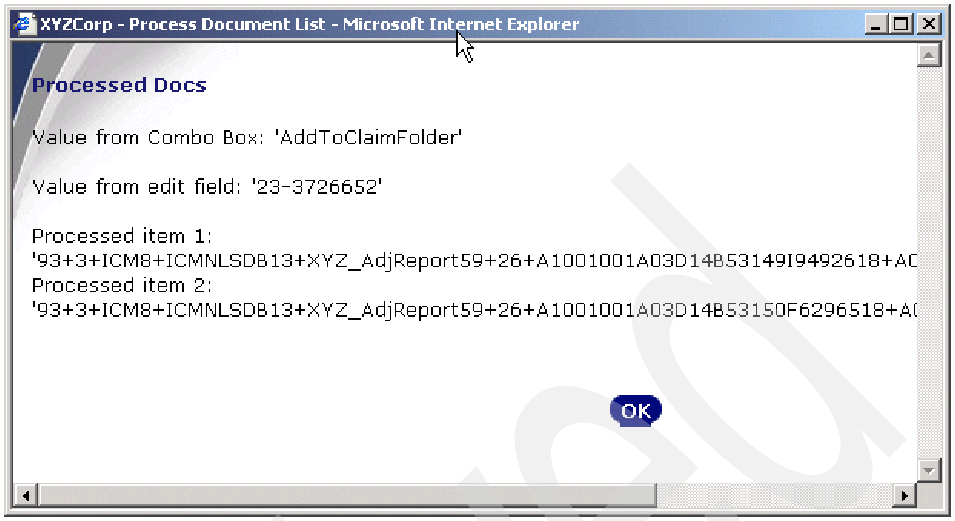
Parameter 1: 23-3726652

Figure 11-11 XYZProcessDocumentList.jsp - Display mode

The window shown in Figure 11-11 can be modified to collect additional information you need for your custom action. If you do not need to collect additional information, invoke XYZProcessDocumentList.jsp from heading.jsp and pass an action of execute instead of display.

In Figure 11-11, the Action is Add to claim folder, and Parameter 1 is set to a claim number. Note that the sample code does not have any specific code to implement the Add to claim folder action; it is provided as a possibility of what you can add to customize your eClient application. The code in the JSP for the display mode has access to all of the PID strings, so you could query the claim number attribute of all selected documents to ensure that the claim number is the same, and then pre-fill the Parameter 1 field with the claim number. Alternatively, you can allow a user to enter a claim number and when the user clicks **Process**, have the custom code store the claim number from the Parameter 1 field as an attribute for each document and also add the document to the claim folder.

When the user clicks **Process** on the display window, JavaScript code in XYZProcessDocumentList.jsp is invoked again, but this time with an action of execute. The code in the JSP takes a different path for the execute action. It lists the values from the display window in Figure 11-11 (the Action value and the Parameter 1 value), loops through each selected document, and displays the PID string in the browser window as shown in Figure 11-12 on page 258. You need to place your custom code to perform the real action in the execute portion of the JSP code.


XYZCorp - Process Document List - Microsoft Internet Explorer

Processed Docs

Value from Combo Box: 'AddToClaimFolder'

Value from edit field: '23-3726652'

Processed item 1:
'93+3+ICM8+ICMNLSDB13+XYZ_AdjReport59+26+A1001001A03D14B53149I49492618+AC

Processed item 2:
'93+3+ICM8+ICMNLSDB13+XYZ_AdjReport59+26+A1001001A03D14B53150F6296518+A(

Figure 11-12 XYZProcessDocumentList.jsp - Execute mode

The rest of this section describes some of the custom code used to implement this example:

1. In Heading.jsp, JavaScript function XYZProcessDocumentList()

   This JavaScript function is called when the user selects **XYZ List Selected Docs** from the search results combo box. It creates a list of item IDs and invokes the XYZProcessDocumentList.JSP file with the action set to display.

   Note that the PID strings for each document are fairly long. You can pass up to about 15 PID strings on the URL for the JSP. If you pass more than about 15 PID strings, you exceed the maximum size of a URL. To get around this, the PID strings are placed in a hidden field on an HTML form. See Example 11-2 for a code snippet.

Example 11-2 XYZProcessDocumentList() code snippet

```
function XYZProcessDocumentList()
{
   pn = parent.ResultsBottom.pageName;
   var numSelected = 0;
   var pidStrings = "";

   //alert("In XYZProcessDocList");

   // If no items in search results window
   if (count == 0)
      return;
```

```
    //
    // Create a list of selected itemids
    //

    if (count > 1)// More than one item in search results window
    {
       for (i = 0; i < document.results.check1.length; i++)
       {
          if (document.results.check1[i].checked == true)
          {
             var itemId = getItemId(document.results.check1[i].value);
             selected = true;
             if (numSelected > 0)
             {
                pidStrings += ",";
             }
             pidStrings += itemId;
             numSelected++;
          }
       }
    }
    else if (count == 1)
    {
       //
       // There is only one item in the search results window.  If it
       // is checked, then add it to pidStrings
       //
       if (document.results.check1.checked == true)
       {
          var itemId = getItemId(document.results.check1.value);
          numSelected = 1;
          pidStrings = itemId;
       }
    }
    if (numSelected > 0)
    {
       //
       // Open XYZProcessDocumentList.jsp with action = 'display'.  Note that 
we also
       // pass in the commad delimited list of PID strings.
       //

       var items = pidStrings;
       var url = '<%= webAppName %>/XYZProcessDocumentList.jsp?srKey=<%= srKey 
%>&action=display';

       var htmlcode ="<html><head><title>XYZ List Selected 
Documents</title></head>" +
             "<body>" +
             "<form name=\"XYZProcessDocumentListForm\" method=\"post\"
action=" + url + " >" +
             "<input type=\"hidden\" name=\"itemids\" value="+items+">" +
              "</form></body></html>";

       // Open a new window with no URL.  After the window is created we'll add
the list of
       // itemids to the form, set the URL, and submit the form.
       wn=generateWindowname();
       setWindowname(wn);
       submitWindow = window.open("", wn,
'resizable=1,scrollbars=1,height=300,width=700');
       submitWindow.document.write(htmlcode);
       submitWindow.document.close();

       // Submit the form.
       submitWindow.document.XYZProcessDocumentListForm.submit();
       return submitWindow;
    }
    else
    {
       alert("You must select at least one document");
    }
}
// XYZ_CHG_END
```

2. XYZProcessDocumentList.JSP

   This file is too large to include in this redbook. Refer to the redbook sample for the complete source code. Three areas in the sample code that we want to bring attention to are as follows:

   – The strAction variable receives the action of display or execute and is used to branch the code appropriately.

   – The JavaScript validate() function is used to validate the data entered on the display form. Example 11-3 is a code snippet that shows how to ensure that the Parameter 1 field value is entered.

Example 11-3 validate() code snippet

```
   //
   // This function is called when the user presses 'OK' on
   // the 'display' window.
   //
   // If you put entry fields on the form, you can validate
   // what the user entered in this JavaScript function.
   // Display an alert message and return false if data
   // not valid, otherwise this calls the progress function
   // to display the 'processing' window.
   //
   function validate()
   {
      //alert("In validate function");

      var txtParam1 =
document.forms["processForm"].elements["txtParam1"].value;

      if (txtParam1.length < 1)
      {
         alert("Parameter 1 field required");
         return false;
      }

      // Display progress window
      //progress("SearchingWait");
      progress("Processing, please wait");

      // Return true.  If this function returns true, then the
      // JavaScript function specified in href for the OK button
      // is called.  That function is processDocs() below which
      // will re-invoke this JSP with an action of 'execute'.
      return true;
   }
```

– The JavaScript function ProcessDocs() is used to get the value from the combo box and entry field from the HTML form and re-invoke the JSP with an action of execute.

Hopefully, with the given sample code, you get an idea of where and how you can customize some of the eClient search results window. As you are planning and making your modifications, always remember to put in the proper comments and isolate your changes so that it will be easier for you to upgrade to later versions of the eClient.

12

Using EIP custom privileges

This chapter describes how you can implement custom privilege definitions in the EIP System Administration client and how these privileges can be accessed by your custom eClient application.

Specifically, this chapter covers the following topics:

- How to define custom privileges
- How to assign custom privileges to users
- How to check for user privileges

12.1 Overview

There are many predefined privileges, privilege groups, and privilege sets. Privileges determine whether a particular user can perform a specific function, such as deleting a document and creating a folder.

When you implement a custom function in the eClient, you may want to control the users that can access the function. Instead of having to create a custom method to define who can access this new function, you can create a custom privilege using the EIP System Administration client, and use custom code to check privileges for a particular user.

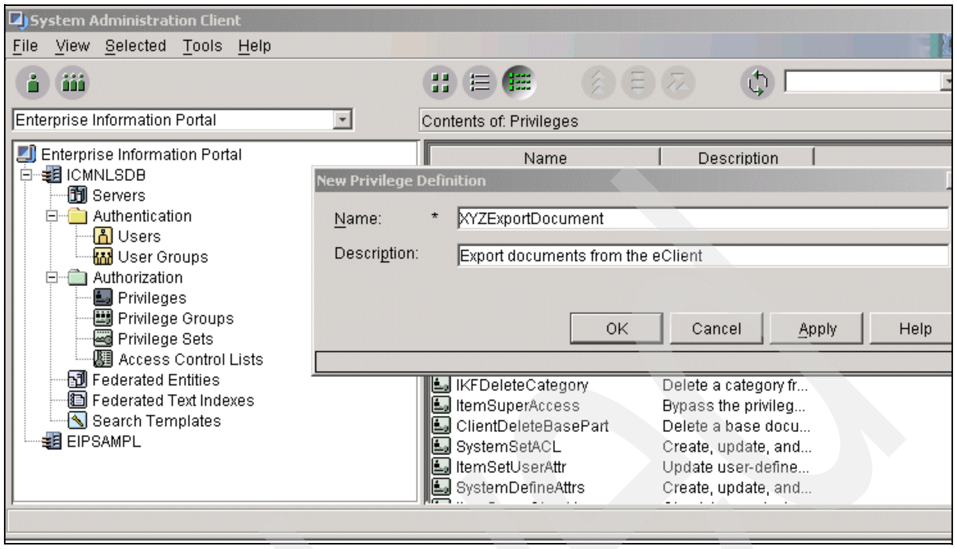
The custom privileges can be accessed via APIs if you use the Content Manager Version 8 connector or the federated connector. This chapter describes how this can be done and provides sample code to determine if the user has a particular privilege.

In this chapter, we guide you through how to use the EIP System Administration client to create a custom privilege and assign it to a user. We also show you the Java code necessary to check for a particular privilege. You can use the privilege to control access to your custom functions in eClient JSPs and servlets.

12.2 Defining custom privileges

To define a custom privilege, open the EIP System Administration client and perform the following steps:

1. Create a custom privilege.
   a. In the tree on the left, select **Authorization -> Privileges**. Right-click **Privileges** and select **New**.
   b. Enter a privilege name. For our scenario, it is XYZExportDocument. Enter a description for the privilege. For our scenario, we entered `Export documents from the eClient`. See Figure 12-1 on page 265.
   c. Click **Apply**.
   d. For our example, also add two other custom privileges:
      - XYZApproveDocument
      - XYZRejectDocument



Name: * XYZExportDocument

Description: Export documents from the eClient

Name	Description
IKFDeleteCategory	Delete a category fr...
ItemSuperAccess	Bypass the privileg...
ClientDeleteBasePart	Delete a base docu...
SystemSetACL	Create, update, and...
ItemSetUserAttr	Update user-define...
SystemDefineAttrs	Create, update, and...

Figure 12-1 Adding XYZExportDocument custom privilege

2. Create a custom privilege group and add the custom privilege to it.

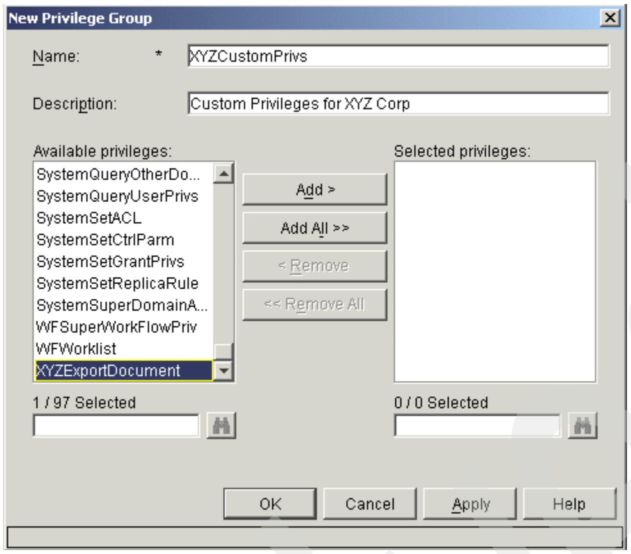
   a. Right-click **Privilege Groups** and select **New**.

   b. Enter a privilege group name. For our scenario, it is XYZCustomPrivs. Enter a description for the privilege group. For our scenario, we entered `Custom Privileges for XYZ Corp`. See Figure 12-2 on page 266.

   c. Select the custom privilege from the Available Privileges and click **Add**. For our scenario, the custom privilege is the one we just created, XYZExportDocument.

   d. Click **OK**.

   Note that the sample code accesses the privilege in the privilege group with one API, and provides another API to check for a particular privilege in the privilege group. For this reason, you may want to include all of your custom privileges in one custom privilege group.



Name: * XYZCustomPrivs

Description: Custom Privileges for XYZ Corp

Available privileges:
SystemQueryOtherDo...
SystemQueryUserPrivs
SystemSetACL
SystemSetCtrlParm
SystemSetGrantPrivs
SystemSetReplicaRule
SystemSuperDomainA...
WFSuperWorkFlowPriv
WFWorklist
XYZExportDocument

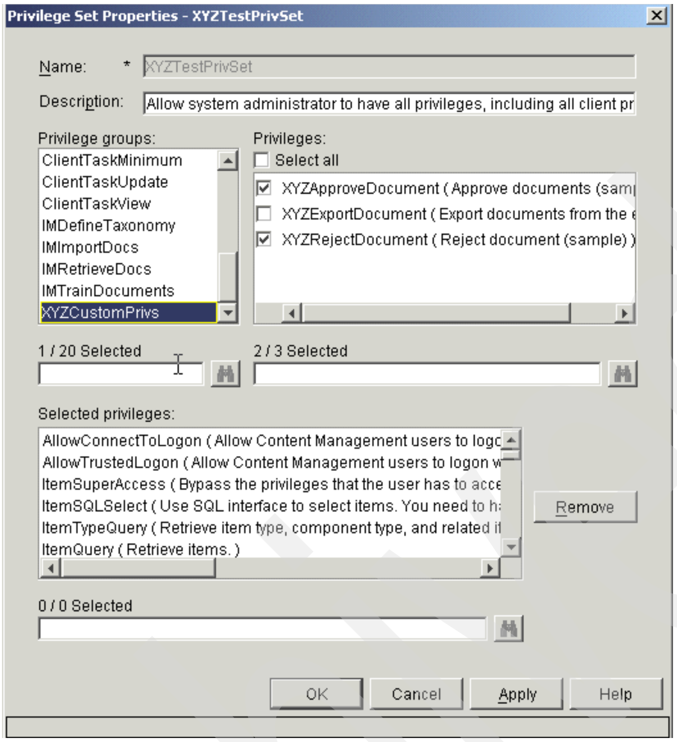
1 / 97 Selected

Selected privileges:

0 / 0 Selected

Figure 12-2 Adding XYZCustomPrivs privilege group

3. Create a privilege set that does not allow export.
   a. From the System Administration client, expand **Authorization -> Privilege Sets**.
   b. Right-click the AllPrivs privilege set in the window on the right and select **Copy**.
   c. Enter a privilege set name. For our scenario, we entered `XYZTestPrivSet`. Enter a description for the privilege set.
   d. From the left Privilege groups pane, select **XYZCustomPrivs**. From the right Privilege pane, uncheck **XYZExportDocument**. See Figure 12-3 on page 267.
   e. Click **OK**.



Name: * XYZTestPrivSet

Description: Allow system administrator to have all privileges, including all client pr

Privilege groups:
- ClientTaskMinimum
- ClientTaskUpdate
- ClientTaskView
- IMDefineTaxonomy
- IMImportDocs
- IMRetrieveDocs
- IMTrainDocuments
- XYZCustomPrivs

Privileges:
- [ ] Select all
- [✓] XYZApproveDocument (Approve documents (sam
- [ ] XYZExportDocument (Export documents from the e
- [✓] XYZRejectDocument (Reject document (sample))

1 / 20 Selected

2 / 3 Selected

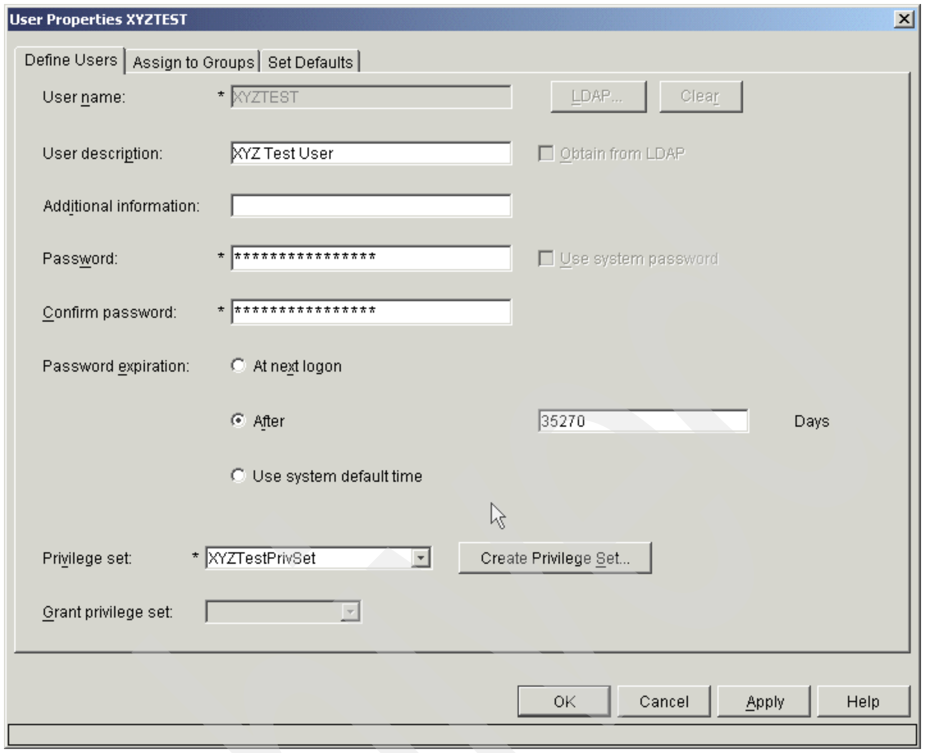
Selected privileges:
- AllowConnectToLogon (Allow Content Management users to logc
- AllowTrustedLogon (Allow Content Management users to logon w
- ItemSuperAccess (Bypass the privileges that the user has to acce
- ItemSQLSelect (Use SQL interface to select items. You need to ha
- ItemTypeQuery (Retrieve item type, component type, and related i
- ItemQuery (Retrieve items.)

0 / 0 Selected

Figure 12-3 Creating new privilege set with export disabled

4. Create a user using the new privilege set:
   a. From the System Administration client, expand **Authentication -> Users**.
   b. Right-click the user **ICMADMIN**, and select **Copy**.
   c. Enter a user ID, description, and password.
   d. Select **XYZTestPrivSet** from both the Privilege set and Grant privilege set drop-down boxes. See Figure 12-4 on page 268.
   e. Click **OK**.

A more typical production implementation of security uses access control lists. This allows you to assign specific privileges for groups of users. Refer to *IBM Content Manager for Multiplatforms: System Administration Guide*, SC27-1335 for more information on using access control lists.


User Properties XYZTEST

Define Users | Assign to Groups | Set Defaults

User name:	* XYZTEST	
User description:	XYZ Test User	☐ Obtain from LDAP
Additional information:		
Password:	* ****************	☐ Use system password
Confirm password:	* ****************	
Password expiration:	○ At next logon	
	● After	35270 Days
	○ Use system default time	
Privilege set:	* XYZTestPrivSet	
Grant privilege set:		

Figure 12-4 Creating test user ID using the new privilege set

12.3 Checking for privileges in a JSP

This section describes how to check for privileges with the sample code provided in the redbook. There are three pieces of code included and described here:

- The sample code you can place in your JSP or servlet to query a custom privilege.
- XYZPrivsData.java, a simple Java class that defines a data structure used by XYZPrivsMethods.java.
- XYZPrivsMethods.java, a Java class that contains the code that can be called by your custom eClient application to check to see if a custom privilege is set or not.

12.3.1 Installing sample code

There are two ways to add XYZPrivsData.java and XYZPrivsMethods.java to your eClient project: using WebSphere Studio Application Developer or using the command line.

Using WebSphere Studio Application Developer

You can install the sample code using WebSphere Studio Application Developer:

1. Add the files XYZPrivsData.java and XYZPrivsMethods.java to the project.
   a. Open the eClient project in WebSphere Studio Application Developer.
   b. Select **File -> Import**.
   c. Select **File System**, click **Next**.
   d. Navigate to the directory containing the com directory with the source files. The com directory should have subdirectory named XYZUtils which should contain the Java files.
   e. Browse to or enter `eclient82\Java Source` in the destination folder field.
   f. Click **Finish**.
2. Copy IDMSearchToolbar-Privs.jsp to IDMSearchToolbar.jsp. You can do this by copying it using the file system, or by cutting and pasting the code using the WebSphere Studio Application Developer editor.
3. Restart the WebSphere test environment. Anytime the Java files are changed, you must restart the WebSphere Studio Application Developer test environment so the changes take effect.

Using command line

You can install the sample code using the command line:

1. Build the class files for each Java file.
2. Locate the eclient82.ear directory in the WebSphere installedApps directory.
3. Locate the eclient82.war\WEB-INF\classes directory
4. Create a com\XYZUtils directory under the classes directory
5. Copy the class files to that directory.
6. Copy IDMSearchToolbar-Privs.jsp to IDMSearchToolbar.jsp in the eclient82.war directory.
7. Restart the WebSphere test environment. Anytime the Java files are changed, you must restart the WebSphere Studio Application Developer test environment so the changes take effect.

12.3.2 Checking custom privileges

In this section, we present the code snippet that checks custom privileges.

You can use the results of the query to control access to your custom functionality as follows:

1. Add the following to the import statement:

   ```
   com.XYZUtils.XYZPrivsData,
   com.XYZUtils.XYZPrivsMethods
   ```

2. Add the source code shown in Example 12-1 to your JSP or servlet.

   The sample code first checks to make sure you are using the federated connector. If not, custom privilege is not supported. After that, it uses getPrivDefs() to load all the privileges, and then uses isAuthorized() to check whether the custom export privilege is set. See the comments within the code for a detailed explanation.

Example 12-1 Code snippet to check custom export privilege

```
// XYZ_CHG_BEGIN
//
// Check to s ee if custom privileges are set.  This code works for federated
// layer only.
boolean blnExport = true;

if (connection.getDsType().equalsIgnoreCase("FED"))
{
   XYZPrivsMethods    XYZPrivs = new XYZPrivsMethods();
   // Load privilege IDs for XYZ_PRIV_GRP_MAIN
   XYZPrivs.getPrivDefs(connection,XYZPrivs.XYZ_PRIV_GRP_MAIN);
   // Check to see if current user has a specified privilege.
   blnExport = XYZPrivs.isAuthorized(connection, XYZPrivs.XYZ_PRIV_EXPORT_DOCS
);
}
else
{
   System.out.println("Custom privileges only supported with federated
connector");
}
System.out.println("Value of blnExport is " + blnExport );
```

12.3.3 Source file XYZPrivsData.java

The source code for this file is available with the redbook. This file contains only a simple data structure that is used by XYZPrivsMethods.java. Example 12-2 on page 271 shows the source code for XYZPrivsData.java.

Example 12-2 XYZPrivsData.java

```
package com.XYZUtils;

public class XYZPrivsData
{
    public String   strPrivName;  // Name of the privilege
    public long     lPrivID;      // Internal ID of the privilege

   // Constructor
   public XYZPrivsData()
   {
   }
}
```

12.3.4 Source file XYZPrivsMethods.java

This is a Java class that contains the code that can be called by your custom eClient application to check if a custom privilege is set or not. The source code for this file is available with the redbook. This is the code that does the bulk of the lookup. Note that you must set the name of your privilege group and your custom privileges in this code. Example 12-3 shows the source code for XYZPrivsMethods.java. See the comments within the source code for a detailed explanation.

Example 12-3 XYZPrivsMethods.java

```
package com.XYZUtils;

import com.ibm.idm.beans.IDMConfigBean;
import com.ibm.mm.beans.*;
import com.ibm.mm.sdk.common.*;
import com.ibm.mm.sdk.server.*;
import java.util.ArrayList;

//
// XYZPrivsMethods
//
public class XYZPrivsMethods {

   // The name(s) of custom privilege groups go here.
   // These values are be used on calls to the getPrivDefs() method.
   public static String XYZ_PRIV_GRP_MAIN       = "XYZCustomPrivs";

   //
   // These are the names of the custom privileges as defined in EIP
   // System Administration client.  Each of them should be in the custom
   // group(s) defined above.  These static strings should be used in the
   // call to isAuthorized().
   //
   public static String XYZ_PRIV_EXPORT_DOCS  = "XYZExportDocument";
   public static String XYZ_PRIV_ACCEPT_DOCS  = "XYZApproveDocument";
   public static String XYZ_PRIV_REJECT_DOCS  = "XYZRejectDocument";
   public static String XYZ_PRIV_PRIV4        = "Add another custom priv
here...";
   public static String XYZ_PRIV_PRIV5        = "Add another custom priv
here...";

   private ArrayList alPrivs = null;
   private String strLoadedPrivGroup = "";
   private IDMConfigBean cvb = new IDMConfigBean();

   // Constructor
   public XYZPrivsMethods()
   {
      System.out.println("XYZPrivsMethods constructor");
   }

   //
   // Get the list of privileges in the XYZPrivs privilege group.  This must
be
   // called before calling isAuthorized().
   //
   public void getPrivDefs( CMBConnection connection, String strPrivGroup )
      throws Exception
   {
      alPrivs = null;

      System.out.println("getPrivDefs: Getting privs for group " +
strPrivGroup );

    if (connection.getDsType().equals(cvb.CMB_DSTYPE_FED))
    {
      //System.out.println("get federated privs");
    }
    else if (connection.getDsType().equals(cvb.CMB_DSTYPE_ICM))
    {
      //System.out.println("get CM8 privs.");
    }
    else
    {
      System.out.println("getPrivDefs works only with FED or CM8 connection");
      return;
    }

      //Get the list of XYZ Privileges and their ids
      CMBSchemaManagement scheme = connection.getSchemaManagement();
```

```
        dkDatastore dStore = connection.getDatastore();
        dkDatastoreDef dStoreDef = dStore.datastoreDef();
        dkDatastoreAdmin dStoreAdmin = dStoreDef.datastoreAdmin();
        dkAuthorizationMgmt dAuthMgmt = dStoreAdmin.authorizationMgmt();

        if (dAuthMgmt == null)
        {
           System.out.println("dAuthMgmt = null");
        }

        // Get the Privilege group specified
        dkPrivilegeGroup XYZPrivGroup =
dAuthMgmt.retrievePrivilegeGroup(strPrivGroup);

        if (XYZPrivGroup == null)
        {
           //
           // Use EIP System Administration application to create
           // custom privileges and a privilege group.
           //
           System.out.println("Cannot find privilege group '" +
                           strPrivGroup);
           strLoadedPrivGroup = "";
        }
        else
        {
           dkCollection collPrivs = XYZPrivGroup.listPrivileges();

           alPrivs = new ArrayList( collPrivs.cardinality() );

           dkIterator collIter = collPrivs.createIterator();
           while (collIter.more())
           {
            if (connection.getDsType().equals(cvb.CMB_DSTYPE_FED))
            {
              // Federated connection
              System.out.println("get federated priv");
                 DKPrivilegeFed XYZPrivFed = (DKPrivilegeFed) collIter.next();
                 System.out.println("Privilege '" + XYZPrivFed.getName() +
                                 "' from PrivGroup '" + strPrivGroup +
                                 "' has id " + XYZPrivFed.getID());
                 XYZPrivsData XYZPrivData = new XYZPrivsData();
                 XYZPrivData.strPrivName = XYZPrivFed.getName();
                 XYZPrivData.lPrivID = XYZPrivFed.getID();
                 alPrivs.add( XYZPrivData );
            }
            else if (connection.getDsType().equals(cvb.CMB_DSTYPE_ICM))
            {
              // Connection is to CM 8 System
```

```
                System.out.println("get CM8 priv");
                   DKPrivilegeICM XYZPriv = (DKPrivilegeICM) collIter.next();
                   System.out.println("Privilege '" + XYZPriv.getName() +
                                   "' from PrivGroup '" + strPrivGroup +
                                   "' has id " + XYZPriv.getID());

                   XYZPrivsData XYZPrivData = new XYZPrivsData();
                   XYZPrivData.strPrivName = XYZPriv.getName();
                   XYZPrivData.lPrivID = XYZPriv.getID();
                   alPrivs.add( XYZPrivData );
             }
            }
            strLoadedPrivGroup = strPrivGroup;
         }

         return;
      }

      //
      // This function finds out if the privilege name is set for the logged on 
   user.
      //
      public boolean isAuthorized( CMBConnection connection, String strMyPrivName 
   )
         throws Exception
      {
         int myPrivID = (int) getPrivID( strMyPrivName );
         System.out.println("In isAuthorized(name) " + strMyPrivName + " is " + 
   myPrivID);
         if (myPrivID <= 0)
         {
            if (strLoadedPrivGroup.length() == 0)
            {
               System.out.println("getPrivDefs() must be called before 
   isAuthorized()");
            }
            else
            {
               System.out.println("Privilege '" + 
                               strMyPrivName + 
                               "' not found in group '" + 
                               strLoadedPrivGroup + "'.");
            }
            return false;
         }

         return isAuthorized( connection, myPrivID );
      }
      //
      // This function finds out if the privilege ID is set for the logged on
   user.
      //
      private boolean isAuthorized( CMBConnection connection, int myPrivID )
         throws Exception
      {
         String privSetName = null;
         System.out.println("IsAuthorized(id) Checking for privid=" + myPrivID );

       if (connection.getDsType().equals(cvb.CMB_DSTYPE_FED))
       {
            DKDatastoreFed dsFed    = (DKDatastoreFed) connection.getDatastore();
            DKDatastoreDefFed dkDef = (DKDatastoreDefFed)dsFed.datastoreDef();
            DKDatastoreAdminFed dsAdminFed = (DKDatastoreAdminFed)
   dkDef.datastoreAdmin();
            DKUserMgmtFed dkUMgmtFed= (DKUserMgmtFed)dsAdminFed.userManagement();
            DKUserDataFed udfUser   = (DKUserDataFed)dkUMgmtFed.retrieveUserDef(
   connection.getUserid() );

            DKACLMgmtFed aclMgmt    = dsAdminFed.aclManagement();
            // Get the users privilege set
            long lPrivId = udfUser.getPrivSetCode();
            DKPrivilegeSetFed privSet =
   (DKPrivilegeSetFed)aclMgmt.retrievePrivilegeSet( lPrivId );

            privSetName = privSet.getName();
            System.out.println("Priv set is " + privSetName );

            // Get list of privileges in the priv set
            dkCollection collPrivs = privSet.listPrivileges();
            dkIterator collIter = collPrivs.createIterator();

            // See if the priv we are looking for is set
            while (collIter.more())
            {
               DKPrivilegeFed XYZPrivFed = (DKPrivilegeFed) collIter.next();
               //System.out.println("  " + XYZPrivFed.getID() );
               if (XYZPrivFed.getID() == myPrivID)
               {
                  //System.out.println("Got a match -- authorized!");
                  return true;
               }
            }
            return false;
       }
         else if (connection.getDsType().equals(cvb.CMB_DSTYPE_ICM))
       {
            System.out.println("Checking priv for DS Type ICM");
```

```
          DKDatastoreICM dsICM    =
             (DKDatastoreICM) connection.getDatastore();
          //System.out.println("dsICM set");
          DKDatastoreDefICM dkDef =
             (DKDatastoreDefICM)dsICM.datastoreDef();
          //System.out.println("dkDef set");
          DKDatastoreAdminICM dsAdminICM =
             (DKDatastoreAdminICM) dkDef.datastoreAdmin();
          //System.out.println("dsAdminICM set");
          DKUserMgmtICM dkUMgmtICM =
             (DKUserMgmtICM)dsAdminICM.userManagement();
          //System.out.println("dkUMgmtICM set");
          DKUserDefICM udfUser=
             (DKUserDefICM)dkUMgmtICM.retrieveUserDef( connection.getUserid()
);
          //System.out.println("udfUser set");

          DKAuthorizationMgmtICM aclMgmt =
             (DKAuthorizationMgmtICM)dsAdminICM.authorizationMgmt();
          // Get the users privilege set
          long lPrivId = udfUser.getPrivSetCode();
          DKPrivilegeSetICM privSet =
(DKPrivilegeSetICM)aclMgmt.retrievePrivilegeSet( lPrivId );

          privSetName = privSet.getName();
          System.out.println("Priv set is " + privSetName );

          // Get list of privileges in the priv set
          dkCollection collPrivs = privSet.listPrivileges();
          dkIterator collIter = collPrivs.createIterator();

          // See if the priv we are looking for is set
          while (collIter.more())
          {
             DKPrivilegeICM XYZPrivICM = (DKPrivilegeICM) collIter.next();
             //System.out.println("  " + XYZPrivICM.getID() );
             if (XYZPrivICM.getID() == myPrivID)
             {
                System.out.println("Got a match -- authorized!");
                return true;
             }
          }
          return false;
       }
    else
    {
       System.out.println("Only supported for FED or CM8 connection.");
       return false;
    }
```

```
   }

   //
   // This function gets the privilege ID for a specified
   // privilege name.  The privilege name must be in the
   // Privilege Group specified in the call to getPrivDefs.
   //
   private long getPrivID( String strMyPrivName )
      throws Exception
   {
      if (alPrivs == null)
      {
         return -1;
      }

      for (int i = 0; i<alPrivs.size(); i++)
      {
         XYZPrivsData XYZPriv = (XYZPrivsData) alPrivs.get(i);
         if (strMyPrivName.equalsIgnoreCase( XYZPriv.strPrivName ) )
         {
            return XYZPriv.lPrivID;
         }
      }
      return 0;
   }
}
```

Part 4

Integrating eClient

In this part, we cover e-mail integration and special topics on Information Mining Service, Siebel integration, and single sign-on integration. The installation, setup, configuration, and integration are presented with detailed step-by-step instructions. In the Information Mining Service chapter (Chapter 13, “Enabling metadata-based content retrieval” on page 281), we provide detailed sample codes for enabling metadata-based data retrieval.

Even if you are not using Information Mining Service, we highly recommend that you read the chapter for more in-depth knowledge about eClient customization.

13

Enabling metadata-based content retrieval

In this chapter, we introduce extensions to eClient and the underlying data model that helps to streamline processes such as content acquisition and content retrieval.

This chapter covers the following Information Mining Service related topics:

- Using categories and summaries in eClient searches
- Creating categories and summaries during document import
- Searching for related items
- Organizing existing items in Content Manager

13.1 Using categories and summaries in eClient searches

Many companies have thousands of text documents in the company databases that are not organized according to any well-defined structure. Usually, there is no automatic process to classify the huge amount of information that comes in daily. An example of this information includes customer requests and news.

To search for documents in these databases, you can specify the terms that you think appear in the documents. This may lead to large result lists and the search query needs to be refined several times to get a meaningful set of results. It is hard to determine the meaningful results because most of the time you just get document names or titles, but you do not know what is actually in the documents. In addition, it is not possible to list all documents that discuss a certain topic, such as microelectronics, but do not contain the term microelectronics itself. This makes the information search time-consuming and cost-intensive for both customers and employees.

In the following sections, using our XYZ company as an example, we show you a simple way to automatically organize your data and solve the problems described. eClient provides a generic search interface to directly search in Content Manager or to search in other content servers using the federated connector. It allows one to build a query with Content Manager item attributes or with criteria of an EIP search template.

We enhance these search capabilities so that at the end of this chapter, you can restrict your eClient search to certain topics and the eClient search results provide a content summary for each document.

13.1.1 Introducing the scenario

We use the press articles that are installed with the EIP sample data as described in 6.2.1, “Setting up sample data” on page 128. To access the data, we use the search template SearchLongBySource which is defined in the sample database EIPSAMPL.

The articles are loaded into Content Manager. We will not simply store the articles. We first use the Information Mining Service of EIP to determine the topic (category) of each document, create a summary, and filter the textual content for indexing. This information is then stored along with the original document in a appropriate Content Manager item. The required steps to create the item type are discussed in 13.1.2, “Creating the data model” on page 283. The document analysis and upload is described in 13.1.3, “Loading the data” on page 286.

Because the documents are already available as text, the filter step is not necessarily required. You could also have other document formats, such as PDF; we do not want to limit the scenario to a certain text format.

In 13.1.4, “Searching with eClient” on page 290, we use eClient to search in the new structure.

The customization steps in 13.1.5, “Customizing eClient” on page 294 are required to make sure the document summaries are displayed properly on the eClient search results window.

13.1.2 Creating the data model

Figure 13-1 shows the structure of the existing DB2 table containing the press articles in the sample database IBMPRESS and the new Content Manager item type to be used to store the documents along with the additional information in database ICMNLSDB.

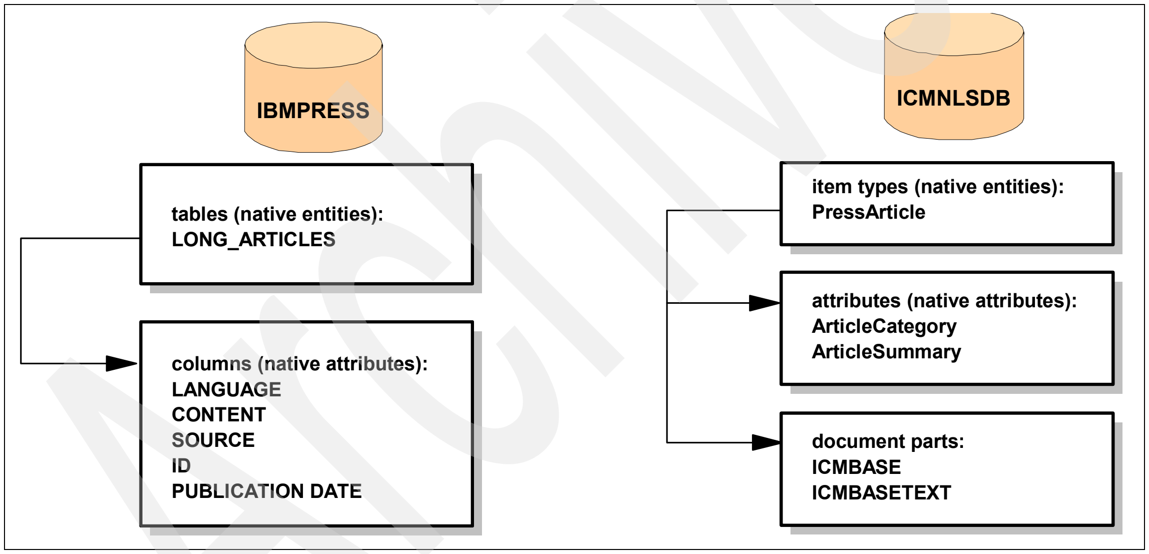
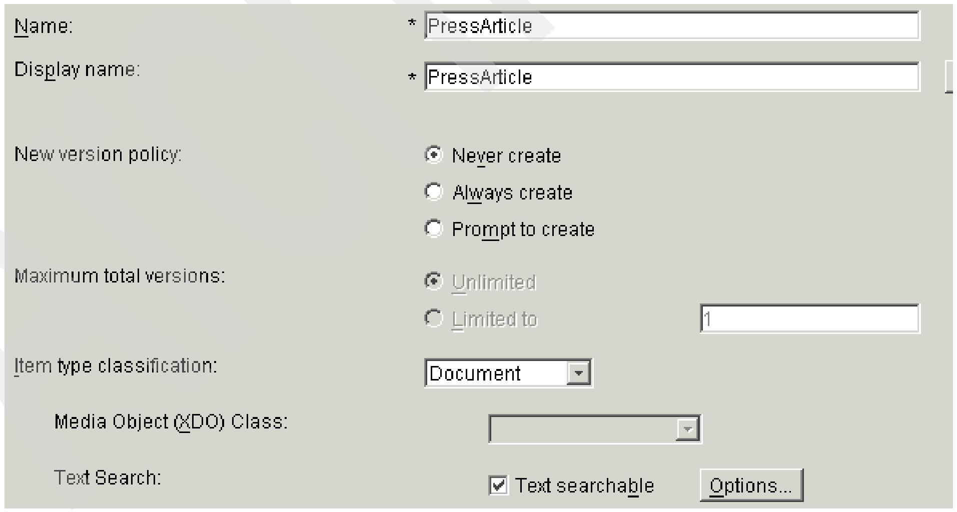


Figure 13-1 Sample database and category search item type

The item type has two attributes to store the category (ArticleCategory) and the summary (ArticleSummary). Furthermore, it provides two parts to store the textual content for indexing (ICMBASETEXT) and the original document (ICMBASE).

To create the item type with the Content Manager System Administration Client, follow these steps:

1. Log on to the System Administration Client, using server type Content Manager and server ICMNLSDB.
2. In the tree view for database ICMNLSDB, expand the category **Library Server Parameters** and select **Configurations**. Right-click **Library Server Configuration** in the Configurations window and select **Properties**.
3. In the Library Server Configuration window, select the **Features** tab, select the **Enable Text Information Extender** check box, and specify the password. Click **OK**.
4. Back in the tree view for database ICMNLSDB, expand the category **Data Modeling**.
5. Right-click **Item Types** and select **New**.
6. In the New Item Type Definition window, select the **Definition** tab and:
   a. Enter the name: `PressArticle`.
   b. Select item type classification: **Document**.
   c. Select the **Text searchable** check box.
   d. Click **Options** next to the Text searchable check box. In the Text Search Options window, specify an index update every 1 minute. Click **OK**.
7. The item type definition should now look similar to Figure 13-2.



Name:	* PressArticle
Display name:	* PressArticle
New version policy:	(●) Never create ( ) Always create ( ) Prompt to create
Maximum total versions:	(●) Unlimited ( ) Limited to 1
Item type classification:	Document
Media Object (XDO) Class:	
Text Search:	✓ Text searchable Options...

Figure 13-2 PressArticle item type definition

8. Select the **Attributes** tab and click the icon on the left to open the New Attribute window (or press Alt+t).

9. In the New Attribute window, enter the name `ArticleCategory` and attribute type Variable character, a minimum character length of 0, and a maximum character length of 1024. Click **Apply**.

10. Now enter the name `ArticleSummary`, attribute type Variable character, a minimum character length of 0, and a maximum character length of 8192. Click **OK**.

11. Back in the New Item Type Definition window, select the new attributes **ArticleCategory** and **ArticleSummary** from the list of available attributes (hold down the Ctrl key) and click **Add**. The Attributes tab should now look similar to Figure 13-3.

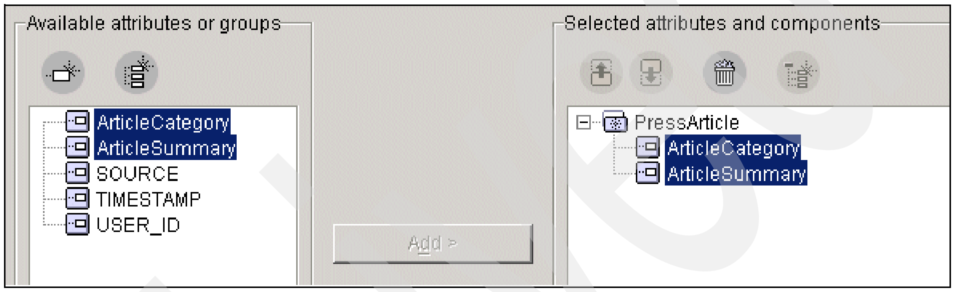
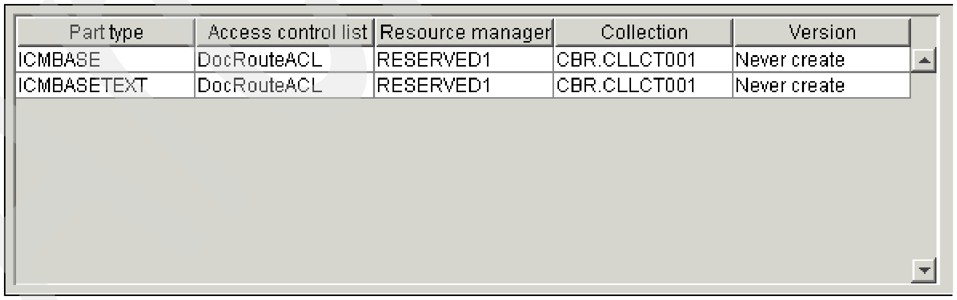


Figure 13-3 PressArticle item type attributes

12. Select the **Document Management** tab and click **Add**.

13. In the Define Document Management Relations window, select part type **ICMBASE**, click **Apply**, select part type **ICMBASETEXT** and click **OK**. The tab should look similar to Figure 13-4.



Part type	Access control list	Resource manager	Collection	Version
ICMBASE	DocRouteACL	RESERVED1	CBR.CLLCT001	Never create
ICMBASETEXT	DocRouteACL	RESERVED1	CBR.CLLCT001	Never create

Figure 13-4 PressArticle item type parts

14. Click **OK**. The PressArticle item type appears in the Item Types window.

Now the data model is defined and you can start to load data.

13.1.3 Loading the data

Using the Information Mining Service of EIP, you can automatically extract the textual content and create the category and summary information for a document as shown in Figure 13-5.

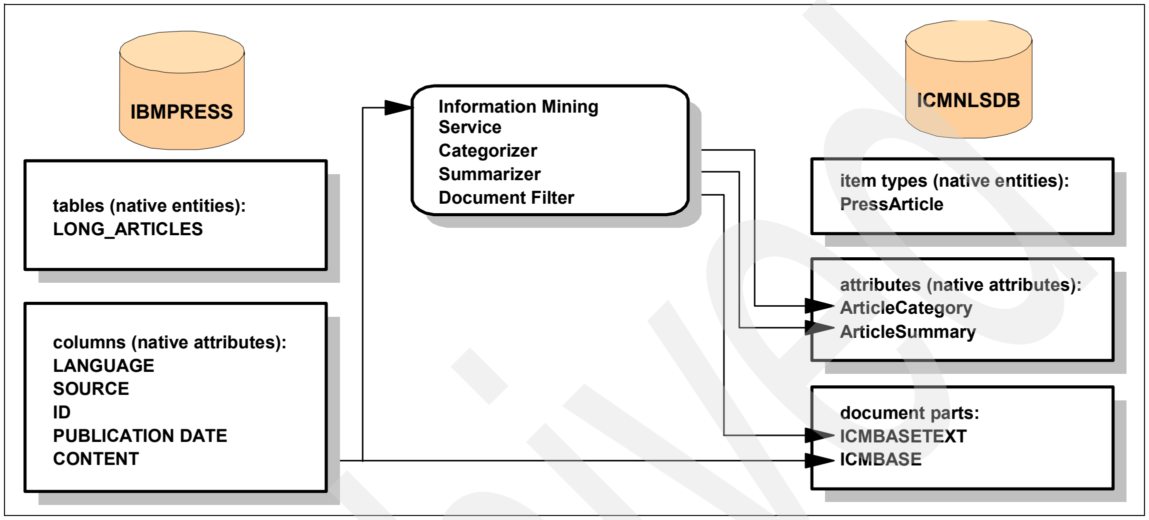


Figure 13-5 Creating structure using Information Mining Service

Unlike the summarizer and the document filter, the categorizer needs to be trained before you can use it. To train the categorizer means to define the set of available categories, assign typical text documents to each category and create a categorization schema that allows categorization of arbitrary new text documents.

To train the categorizer, you use the Information Structuring Tool, which is part of the Information Mining Service. Please follow the instructions in “Running the categorization sample” on page 155 to train the categorizer before you continue with this scenario.

After the training, the categorizer is able to distinguish between four different categories:

- Desktop_and_mobile_computers
- Global Financing
- Microelectronics
- Research

The application ImportPressArticles.java, which loads the data according to Figure 13-5, can be found in the samples directory of the redbook. It is implemented using the JavaBeans API. Figure 13-6 shows the beans and how they are connected to define the required event flow.

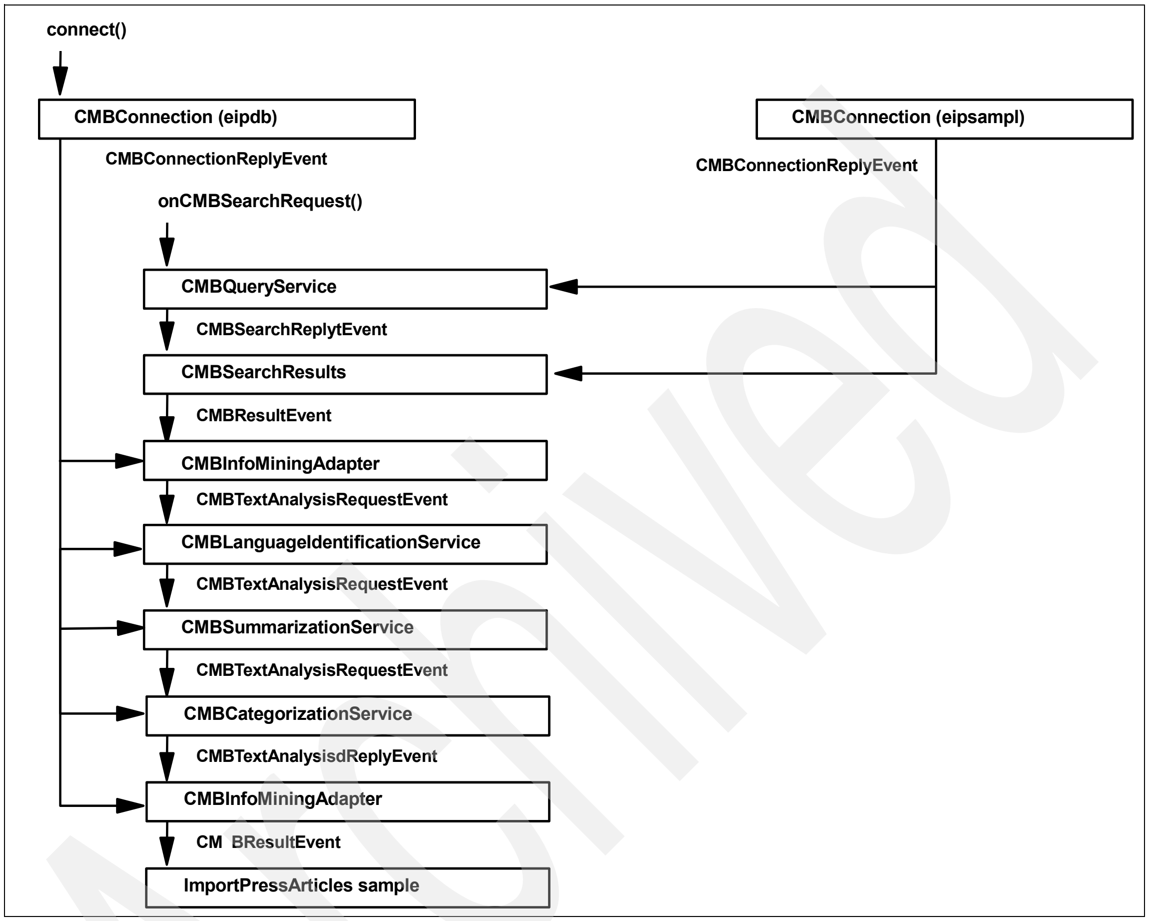


Figure 13-6 Loader application JavaBeans event flow

The query bean and search results bean need a connection to the EIPSAMPL database, because this is where the federated entity and the search template SearchLongBySource is defined, which is required to access the data in DB2 through the federated connector. The beans of the Information Mining Service need to be connected to the EIP administration database EIPDB. The categorization bean, for example, needs to look up the categorization model from this database.

The search results bean populates the result list and fires a result event, which contains the result items, to the adapter bean, which retrieves the items and extracts the text from the item parts using the document filter. The text gets stored in the item in a transient object. The adapter then initiates a text analysis request against the language identification bean. Language identification is a required analysis before you can run summarization and categorization. The categorization bean reads the available category structure (taxonomy) and categorization model from the catalog we just configured using the Information Structuring Tool, and uses this information to determine the category for each item. The summary and category get stored in the item but transient only.

The categorization bean then invokes the adapter bean, which converts the reply event back to a search result event, where each contained item is now enriched with the textual content, the summary, and the category. It also contains the language, but we ignore the language in this scenario.

Finally, the application creates items of item type PressArticle in the ICMNLSDB database that contain all the created information; therefore, a third connection is required.

The event flow for the text analysis is defined with the following code snippet:

```
queryService.addCMBSearchReplyListener(searchResults);
searchResults.addCMBResultListener(adapter1);
adapter1.addCMBTextAnalysisRequestListener(languageIdentificationService);
languageIdentificationService.addCMBTextAnalysisRequestListener(summarizati
onService);
summarizationService.addCMBTextAnalysisRequestListener(categorizationServic
e);
categorizationService.addCMBTextAnalysisReplyListener(adapter2);
adapter2.addCMBResultListener(this);
```

The search template can be retrieved by name (SearchLongBySource):

```
CMBSearchTemplate searchTemplate =
schema.getSearchTemplate(SEARCH_TEMPLATE_NAME);
```

A value needs to be specified for the search criteria (source):

```
String[] searchValues = {SEARCH_VALUE};
searchTemplate.setSearchCriterion(SEARCH_CRITERION_NAME,
CMBBaseConstant.CMB_OP_EQUAL, searchValues);
```

And to start the search, a request event is created with the template and fired against the query bean:

```
CMBSearchRequestEvent searchRequest = new CMBSearchRequestEvent(this,
CMBSearchRequestEvent.CMB_REQUEST_SEARCH_SYNCH, searchTemplate);
queryService.onCMBSearchRequest(searchRequest);
```

The textual content, the category and the summary can then be obtained from the record object contained in a result item:

```
CMBRecord currentRecord = currentItem.getInfoMiningRecord();
String category = (String) currentRecord.getValue("IKF_CATEGORIES");
String summary = (String) currentRecord.getValue("IKF_SUMMARY");
String content = (String) currentRecord.getValue("IKF_CONTENT");
```

The original document needs to be retrieved:

```
eipsamplDataManagement.setDataObject(currentItem);
eipsamplDataManagement.retrieveItem();
CMBObject eipsamplObject = eipsamplDataManagement.getContent(0);
```

Now we start to create a new item:

```
CMBItem item = new CMBItem();
```

Specify the item type name PressArticle as the entity name:

```
item.setEntityName(ICM_ENTITY_NAME);
```

And set the category and summary on the appropriate attributes ArticleCategory and ArticleSummary:

```
item.addAttr(ICM_CATEGORY_ATTR_NAME, category,
CMBItem.CMB_DATATYPE_VSTRING);
item.addAttr(ICM_SUMMARY_ATTR_NAME, summary, CMBItem.CMB_DATATYPE_VSTRING);
```

The original document is set on an object that will become one part of the new item:

```
CMBObject icmBaseObject = new CMBObject();
icmBaseObject.setData(eipsamplObject.getData());
icmBaseObject.setPartType("ICMBASE");
```

Another object is created for the textual content:

```
CMBObject icmBaseTextObject = new CMBObject();
icmBaseTextObject.setData(content.getBytes());
icmBaseTextObject.setPartType("ICMBASETEXT");
```

Then the data management is used to add both parts to the item and finally create it in Content Manager:

```
icmDataManagement.setDataObject(item);
icmDataManagement.addContent(icmBaseObject);
icmDataManagement.addContent(icmBaseTextObject);
icmDataManagement.createItem(ICM_ENTITY_NAME);
```

To compile and run the sample:

1. Open an EIP development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**.
2. Change to the directory where you stored the redbook samples.
3. To compile the sample, run **`javac ImportPressArticles.java`**.
4. To run the sample, enter **`java ImportPressArticles`**.

The persistent identifier, the category and the summary is printed for each document to be stored. After the application has finished, all documents are loaded and you can run eClient to search.

13.1.4 Searching with eClient

Using eClient, you can now search in categories and get document summaries in the search result list.

To search in the new data model, open eClient and on the Logon window, enter `ICMNLSDB(CM8)` for the server as shown in Figure 13-7.

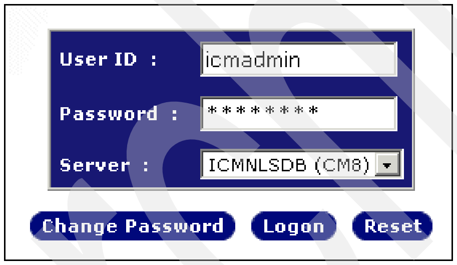
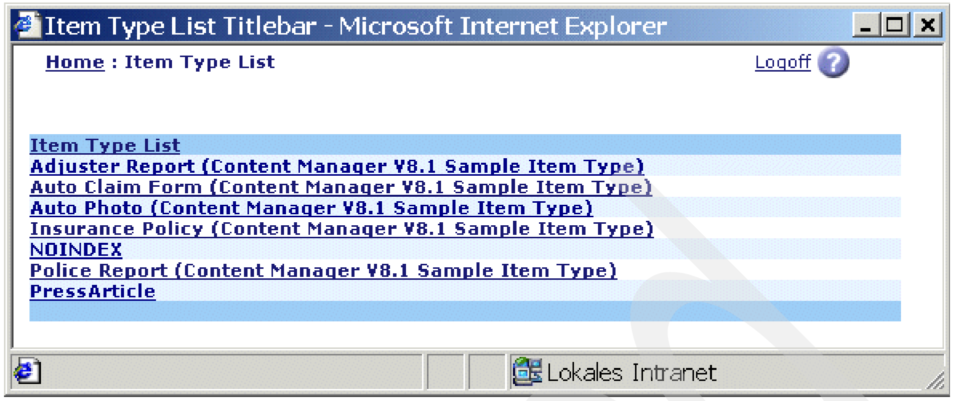

User ID : icmadmin
Password : ********
Server : ICMNLSDB (CM8)

Figure 13-7 Log on to Content Manager database

The available item types are listed on the Item Type List window as shown in Figure 13-8 on page 291. Select the new **PressArticle** item type.


Item Type List Titlebar - Microsoft Internet Explorer

Home : Item Type List

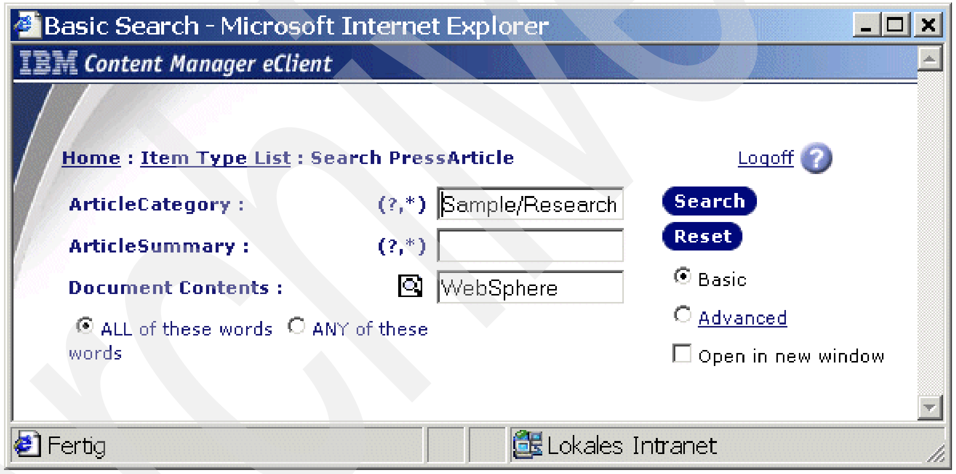
Logoff

Item Type List
Adjuster Report (Content Manager V8.1 Sample Item Type)
Auto Claim Form (Content Manager V8.1 Sample Item Type)
Auto Photo (Content Manager V8.1 Sample Item Type)
Insurance Policy (Content Manager V8.1 Sample Item Type)
NOINDEX
Police Report (Content Manager V8.1 Sample Item Type)
PressArticle

Lokales Intranet

Figure 13-8 List of available item types

Figure 13-9 shows the search form that comes up.


Basic Search - Microsoft Internet Explorer

IBM Content Manager eClient

Home : Item Type List : Search PressArticle

Logoff

ArticleCategory : (?,*) Sample/Research

ArticleSummary : (?,*)

Document Contents : WebSphere

[✓ ALL of these words] ANY of these words

Search

Reset

[✓ Basic]

Advanced

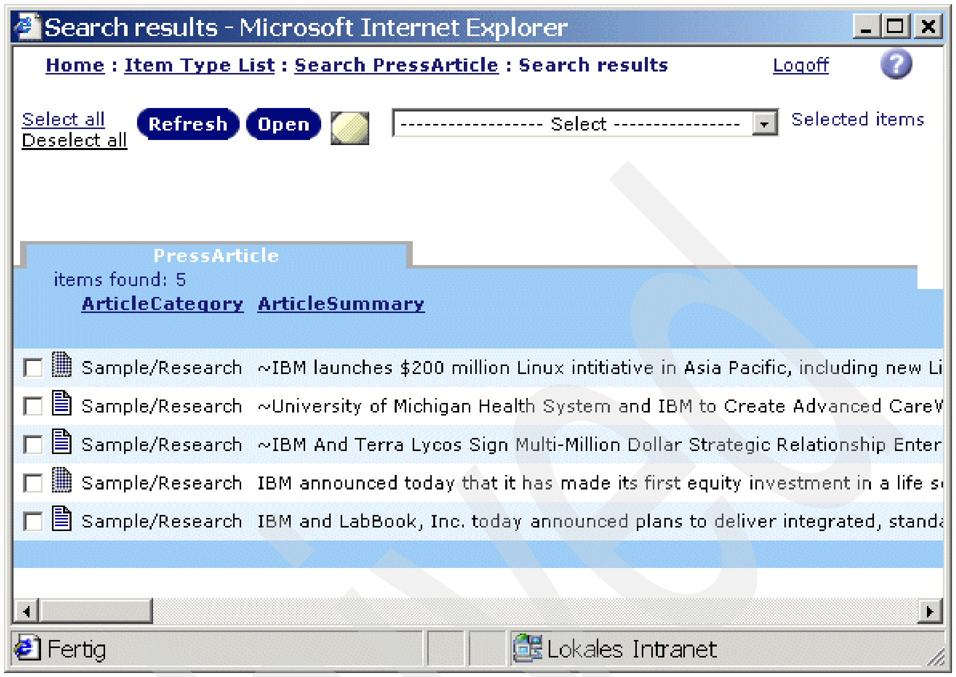
Open in new window

Fertig

Lokales Intranet

Figure 13-9 eClient search in category

The search can now be restricted to a certain category. To search for all research documents that are related to WebSphere, enter that category name (`Sample/Research`) in the ArticleCategory field and `WebSphere` in the Document Contents field. Click **Search** and the result list should looks similar to Figure 13-10 on page 292.



Search results - Microsoft Internet Explorer

Home : Item Type List : Search PressArticle : Search results Logoff

Select all
Deselect all

PressArticle

items found: 5

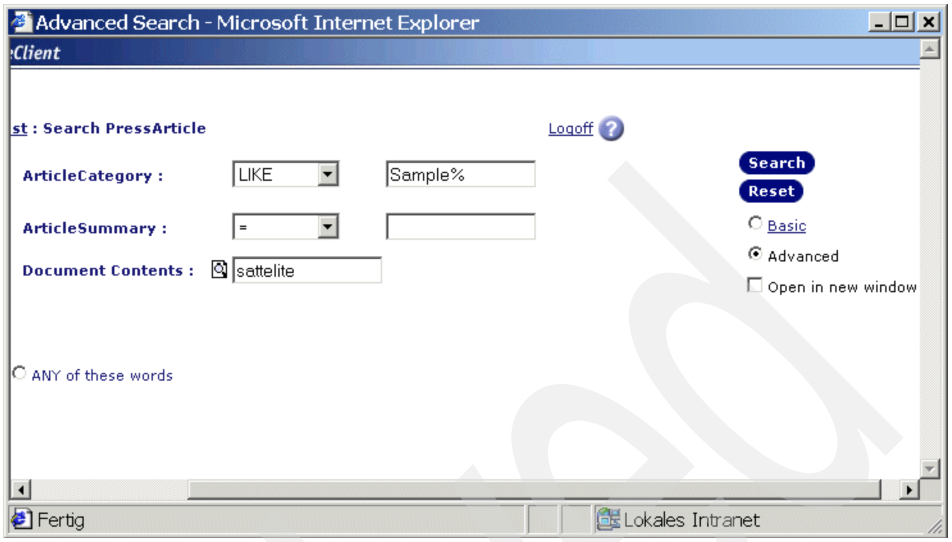
	ArticleCategory	ArticleSummary
☐	Sample/Research	~IBM launches $200 million Linux intitiative in Asia Pacific, including new Li
☐	Sample/Research	~University of Michigan Health System and IBM to Create Advanced CareW
☐	Sample/Research	~IBM And Terra Lycos Sign Multi-Million Dollar Strategic Relationship Enter
☐	Sample/Research	IBM announced today that it has made its first equity investment in a life s
☐	Sample/Research	IBM and LabBook, Inc. today announced plans to deliver integrated, stand

Fertig Lokales Intranet

Figure 13-10 eClient category search result list

All search results belong to the topic (or business segment) you are interested in and contain the specified term. The short result list and the displayed summary help to find the interested documents quickly.

The categories are organized hierarchically and you might want to search for all documents assigned to the categories of a subtree. You can do that on the Advanced Search window as shown in Figure 13-11 on page 293.


Advanced Search - Microsoft Internet Explorer

st : Search PressArticle

ArticleCategory : LIKE Sample%

ArticleSummary : =

Document Contents : sattelite

ANY of these words

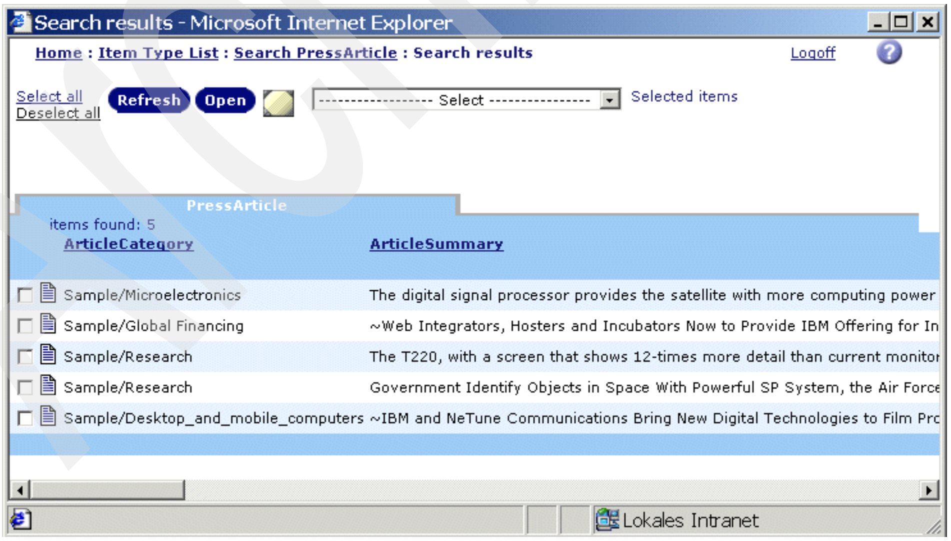
[Basic ✓ Advanced] Open in new window

Fertig — Lokales Intranet

Figure 13-11 eClient advanced search in sub-categories

Choose the **LIKE** operator and specify the root category of the subtree with a trailing %.

The resulting list contains all documents that belong to the subtree, starting with the specified category as shown in Figure 13-12.


Search results - Microsoft Internet Explorer

Home : Item Type List : Search PressArticle : Search results

PressArticle

items found: 5

ArticleCategory	ArticleSummary
Sample/Microelectronics	The digital signal processor provides the satellite with more computing power
Sample/Global Financing	~Web Integrators, Hosters and Incubators Now to Provide IBM Offering for In
Sample/Research	The T220, with a screen that shows 12-times more detail than current monitor
Sample/Research	Government Identify Objects in Space With Powerful SP System, the Air Force
Sample/Desktop_and_mobile_computers	~IBM and NeTune Communications Bring New Digital Technologies to Film Prc

Figure 13-12 eClient sub-category search result list

Because the summary appears in just one line, we need to customize eClient to enhance the readability. This is discussed in 13.1.5, “Customizing eClient” on page 294.

13.1.5 Customizing eClient

With a word wrap at the right border of the browser window, the summary shown in the search result list would be more readable. The eClient JSP file ItemTable.jsp is responsible for the display of item collections. You need to adapt this file.

Follow these steps to update the file:

1. Open file ItemTable.jsp in your preferred development environment.
2. About line 1285, replace the line::

   ```
   rowsColumns[row][col + 1] = "<TD class='" + rowType + "' align='left' " +
   colWidth + " nowrap>" + item.getAttrValue(colName) + "</TD>";
   ```

   with:

   ```
   String attrValue = item.getAttrValue(colName);
   if(attrValue.length() > 100)
   rowsColumns[row][col + 1] = "<TD class='" + rowType + "' align='left'
   width='100%' >" + attrValue + "</TD>";
   else
   rowsColumns[row][col + 1] = "<TD class='" + rowType + "' align='left' " +
   colWidth + " nowrap>" + attrValue + "</TD>";
   ```

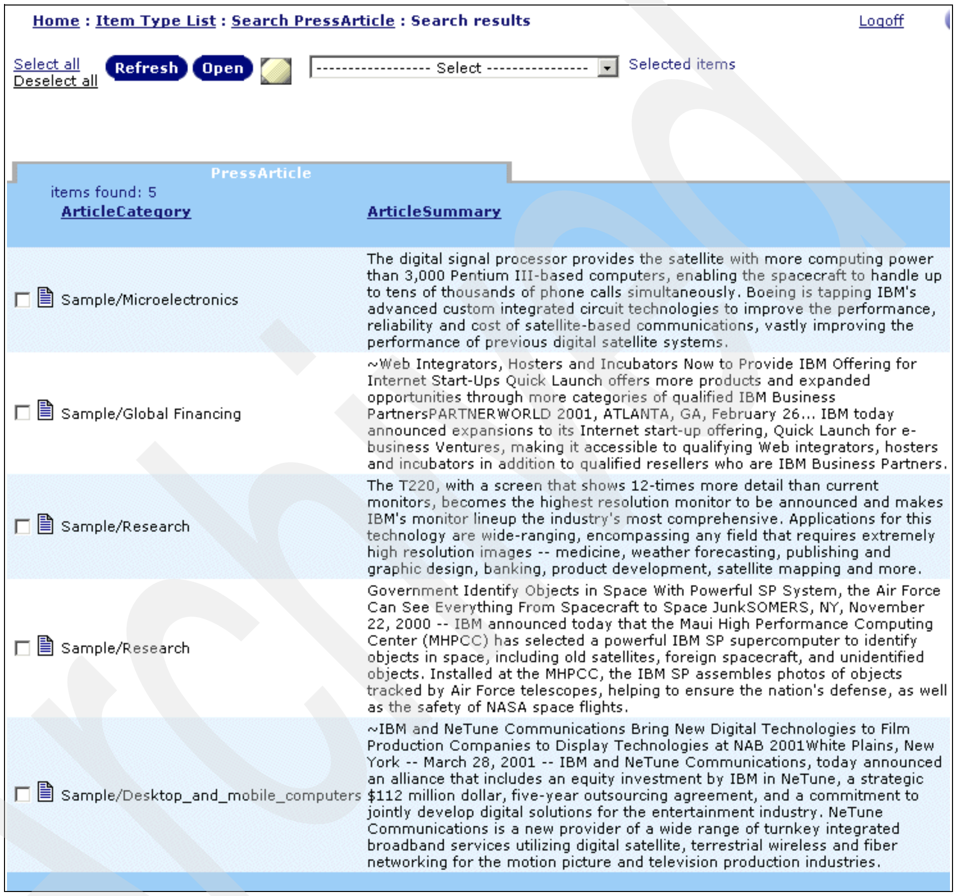
Now eClient needs to be updated with the JSP file. If you use the WebSphere Application Server Application Assembly Tool, do the following steps:

1. Make sure the ItemTable.jsp file resides in a directory named pageComponents.
2. Open the eClient .ear file using the WebSphere Application Server Application Assembly Tool and expand the category **Web Modules -> eclient82 -> Files**. You see class files, JAR files and resource files.
3. Right-click **Resource Files** and select **Add Files**. The Add Files window appears.
4. Click **Browse**. Select the directory (it must appear in the File name field) that contains the pageComponents directory and click **Select**.
5. From the upper-right pane in the Add Files window, select **pageComponents** and click **Add**. The JSP file appears in the Selected Files list as pageComponents/ItemTable.jsp.
6. Click **OK** and select **Yes** on the confirmation window to overwrite the existing JSP file.

7. Select **File -> Save** to save the .ear file and then select **File -> Close**.

Use the WebSphere Application Server Administrative Console to uninstall eClient and install the new version.

The new search result looks like Figure 13-13.


Home : Item Type List : Search PressArticle : Search results
Logoff
Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items
PressArticle
items found: 5

ArticleCategory	ArticleSummary
Sample/Microelectronics	The digital signal processor provides the satellite with more computing power than 3,000 Pentium III-based computers, enabling the spacecraft to handle up to tens of thousands of phone calls simultaneously. Boeing is tapping IBM's advanced custom integrated circuit technologies to improve the performance, reliability and cost of satellite-based communications, vastly improving the performance of previous digital satellite systems.
Sample/Global Financing	~Web Integrators, Hosters and Incubators Now to Provide IBM Offering for Internet Start-Ups Quick Launch offers more products and expanded opportunities through more categories of qualified IBM Business PartnersPARTNERWORLD 2001, ATLANTA, GA, February 26... IBM today announced expansions to its Internet start-up offering, Quick Launch for e-business Ventures, making it accessible to qualifying Web integrators, hosters and incubators in addition to qualified resellers who are IBM Business Partners.
Sample/Research	The T220, with a screen that shows 12-times more detail than current monitors, becomes the highest resolution monitor to be announced and makes IBM's monitor lineup the industry's most comprehensive. Applications for this technology are wide-ranging, encompassing any field that requires extremely high resolution images -- medicine, weather forecasting, publishing and graphic design, banking, product development, satellite mapping and more.
Sample/Research	Government Identify Objects in Space With Powerful SP System, the Air Force Can See Everything From Spacecraft to Space JunkSOMERS, NY, November 22, 2000 -- IBM announced today that the Maui High Performance Computing Center (MHPCC) has selected a powerful IBM SP supercomputer to identify objects in space, including old satellites, foreign spacecraft, and unidentified objects. Installed at the MHPCC, the IBM SP assembles photos of objects tracked by Air Force telescopes, helping to ensure the nation's defense, as well as the safety of NASA space flights.
Sample/Desktop_and_mobile_computers	~IBM and NeTune Communications Bring New Digital Technologies to Film Production Companies to Display Technologies at NAB 2001White Plains, New York -- March 28, 2001 -- IBM and NeTune Communications, today announced an alliance that includes an equity investment by IBM in NeTune, a strategic $112 million dollar, five-year outsourcing agreement, and a commitment to jointly develop digital solutions for the entertainment industry. NeTune Communications is a new provider of a wide range of turnkey integrated broadband services utilizing digital satellite, terrestrial wireless and fiber networking for the motion picture and television production industries.

Figure 13-13 Customized eClient search result list

Now the summary is displayed properly and provides a good idea whether a document is of interest or not.

13.2 Creating categories and summaries during document import

In 13.1, “Using categories and summaries in eClient searches” on page 282, we discuss how to import new documents using a separate application that automatically creates additional category and summary information.

eClient itself also supports document import. If you specify `importSupported=true` in the IDM.properties file of eClient, the welcome window provides an Import link. You can import a file from the file system and manually specify attribute values.

In the following, we show how to change the import behavior of the eClient, so that additional information such as categories and summaries are automatically created and stored in the new item.

We assume the item type PressArticle has been created as described in 13.1.2, “Creating the data model” on page 283. The provided solution itself does not depend on this item type. It is customizable and can therefore be used with any other item type.

It is also necessary to train the categorizer. Please follow the instructions in “Running the categorization sample” on page 155 before you continue with this scenario.

13.2.1 Changing eClient import behavior

Figure 13-14 on page 297 illustrates the control flow that is initiated when you click the **Import** link on the eClient welcome window.

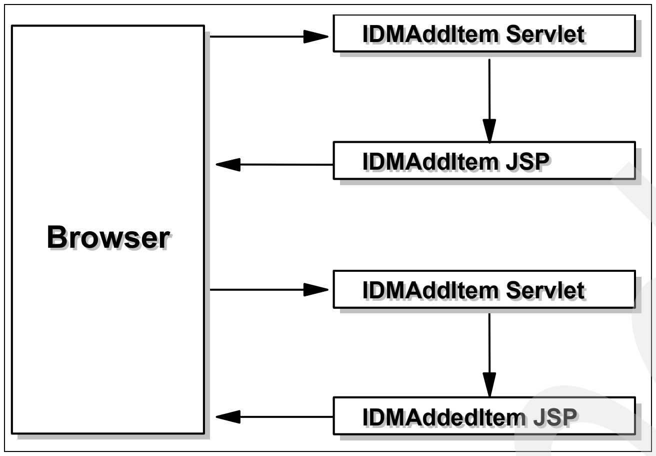
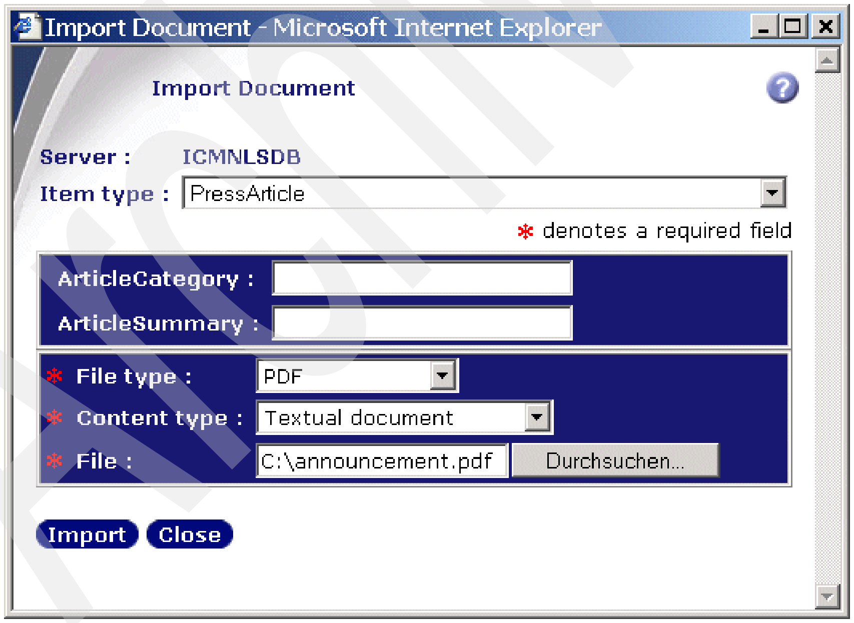


Figure 13-14 eClient import control flow

The browser sends a request to the IDMAddItem servlet. The servlet calls the IDMAddItem JSP, which creates an HTML form where you can specify attribute values and select the file to upload. The form is shown in Figure 13-15.



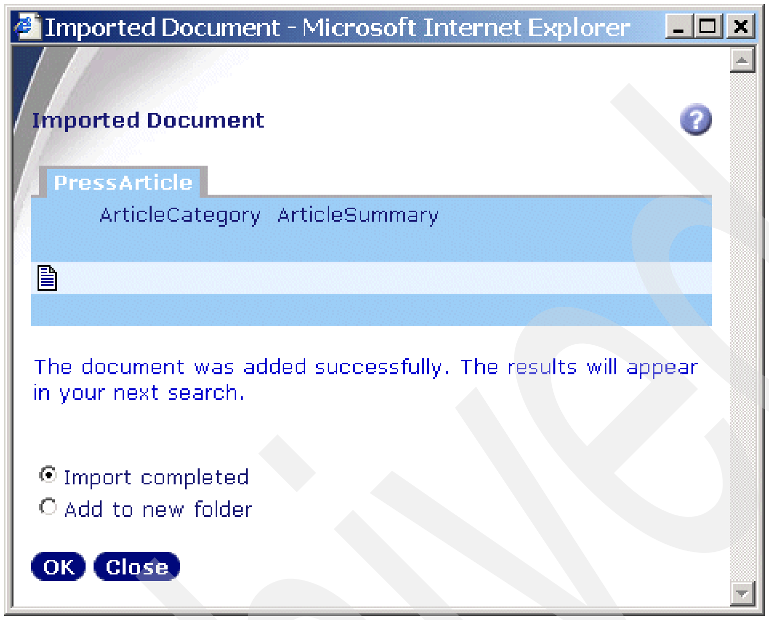
Server : ICMNLSDB
Item type : PressArticle

✱ denotes a required field

ArticleCategory :	
ArticleSummary :	
✱ File type :	PDF
✱ Content type :	Textual document
✱ File :	C:\announcement.pdf

Figure 13-15 Output of IDMAdditem.jsp

If you submit the form, the browser sends the information back to the IDMAddItem servlet, which then creates a new item in the server. Now the servlet calls the IDMAddedItem JSP to create the confirmation window as shown in Figure 13-16.


Imported Document - Microsoft Internet Explorer

Imported Document

PressArticle

ArticleCategory	ArticleSummary

The document was added successfully. The results will appear in your next search.

[✓ Import completed]
Add to new folder

Figure 13-16 Output of IDMAddedItem.jsp

Because the values for the category attribute and the summary attribute can be created automatically, we should add this to the control flow so the users do not need to enter these values manually.

How can that be achieved?

According to the Model-View-Controller design of eClient (see Chapter 5, “eClient architecture” on page 101), you cannot do that in the JSPs.

Since we also cannot change the IDMAddItem servlet itself, it would be a good idea to configure a servlet filter, which enriches the servlet request by wrapping the request object and returning the created values for the attributes when the getParameter method is called on the wrapper.

But you cannot do that because of two problems. The servlet filter concept is introduced with Servlet 2.3 in J2EE 1.3. But the eClient is currently a J2EE 1.2 application, which supports Servlet 2.2 only. The other problem is that the getParameter method does not work because the content type of the post form data is multipart/form-data, and the Servlet specification demands content type application/x-www-form-urlencoded for the parameter set. The data can only be retrieved as a data stream by calling getInputStream. You cannot provide a wrapper for this stream.

You could solve the filter problem by changing the control flow so that a custom servlet is called first and in this servlet the information is created and forwarded to the IDMAddItem servlet. But you would still need to wrap the request object, which is not possible.

You cannot create the attribute values before the item is created, but you can do it afterwards as illustrated in Figure 13-17.

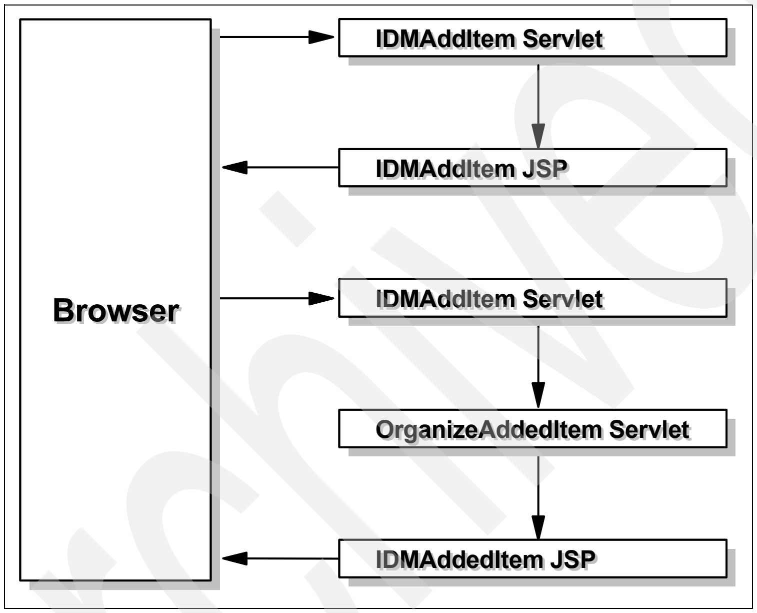


Figure 13-17 Calling custom servlet after new item has been added

After the IDMAdditem servlet created the item, it calls the custom OrganizeAddedItem servlet. This behavior can be configured in the IDM.properties file of the eClient. The custom servlet reads the required information directly from the request or the session, creates the attribute values and updates the item in the server. Then the custom servlet calls the IDMAddedItem JSP to create the response window.

13.2.2 Implementing custom servlet

The OrganizeAddedItem servlet is provided in the redbook samples as OrganizeAddedItem.java.

In the init method, the implementation first reads the servlet initialization parameters:

- eipDatabaseName, used to connect to the Information Mining Service
- entityName, to identify the items to be updated
- categoryAttributeName, the name of the attribute to store the category
- summaryAttributeName, the name of the attribute to store the summary
- contentIndex, the index of the content part in the item
- catalogName, the name of the Information Mining Service catalog containing the categorization schema
- responseJspUrlPattern, the URL pattern to be used to call the response JSP after the servlet has finished

These parameters make the servlet adaptable to other databases.

Both service methods, doGet and doPost, call processRequest. Hence, the servlet behavior is the same for both request types.

In the processRequest method, the item to be updated is retrieved from the request object:

```
CMBItem item = (CMBItem)req.getAttribute("item");
```

And the connection object is retrieved from the session object:

```
CMBConnection connection =
(CMBConnection)req.getSession().getAttribute("connection");
```

If the item is not of the expected item type, we do not process it and forward to the response JSP using the URL pattern previously read from an initialization parameter:

```
if(!item.getEntityName().equals(entityName)) {
getServletContext().getRequestDispatcher(responseJspUrlPattern).forward(req
, resp);
}
```

In order to use the Information Mining Service, it needs to be connected to the EIP database. One way to do that is to connect a CMBConnection that has the connectToIKF property set to true. We use the same user ID as the connection read from the session:

```
eipConnection = new CMBConnection();
eipConnection.setServerName(eipDatabaseName);
eipConnection.setUserid(connection.getUserid());
eipConnection.setPassword(connection.getPassword());
eipConnection.setConnectToIKF(true);
eipConnection.connect();
```

The service object can then be retrieved from the connection. For performance reasons, we also retrieve and keep the catalog and taxonomy objects required for categorization:

```
ikfService = eipConnection.getIKFService();
catalog = ikfService.getLibrary().getCatalog(catalogName);
taxonomy = catalog.getTaxonomy();
```

Then the current summary and category values and the content is read from the item:

```
String category = item.getAttrValue(categoryAttributeName);
String summary  = item.getAttrValue(summaryAttributeName);
CMBObject content = dataManagement.getContent(contentIndex);
```

We call the document filter to create the text document object required for the analysis steps:

```
DKIKFTextDocument ikfDocument =
ikfFilter.getTextDocument(content.getDataStream());
```

If the retrieved summary is empty because the user did not specify one, we now run the language identifier, because summarization requires the language of the document:

```
DKIKFLanguageIdentificationResult langResult[] =
langIdentifier.analyze(ikfDocument);
```

The language is set on the document object:

```
ikfDocument.setLanguage(language);
```

And the summarizer is called to create the summary:

```
summary = summarizer.analyze(ikfDocument).getSummary();
```

If the retrieved category is empty, we determine it the same way. After the language has been detected and set on the document, the categorizer is called:

```
DKIKFCategorizationResult categorizationResult[] =
categorizer.analyze(ikfDocument);
```

Because the resulting category object does not contain the category path, we need to retrieve the corresponding category object from the taxonomy and use that to get the path:

```
DKIKFCategory resultCategory = categorizationResult[0].getCategory();
category = taxonomy.getCategory(resultCategory).getPathAsString();
```

Finally, the new attribute values are set on the item using the attribute names retrieved from the intitalization parameters, the item is checked out, updated, and checked in again:

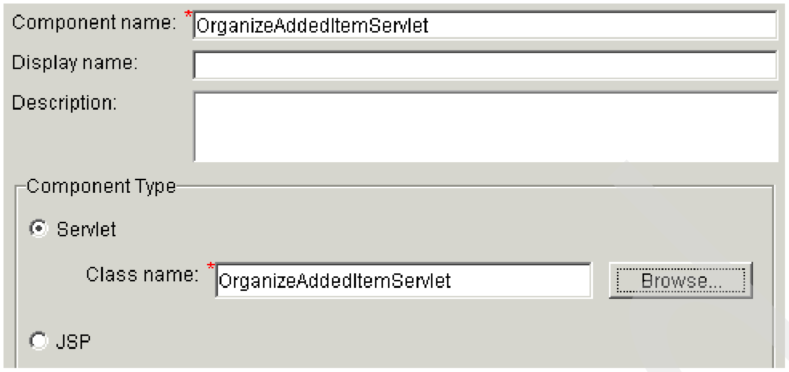
```
item.setAttrValue(categoryAttributeName, category);
item.setAttrValue(summaryAttributeName, summary);
dataManagement.setDataObject(item);
dataManagement.checkOut();
dataManagement.updateItem();
dataManagement.checkIn();
```

Now the work is done and we can call the response JSP using the URL pattern also retrieved from an initialization parameter:

```
getServletContext().getRequestDispatcher(responseJspUrlPattern).forward(req
, resp);
```

After the servlet is implemented, it needs to be added to eClient. If you use the WebSphere Application Server Application Assembly Tool, do the following steps:

1. Open the eClient .ear file using the WebSphere Application Server Application Assembly Tool and expand the category **Web Modules -> eclient82 -> Files**. You see class files, JAR files and resource files.
2. Right-click **Class Files** and select **Add Files**. The Add Files window appears.
3. Click **Browse**. Select the directory (it must appear in the File name field) that contains the OrganizeAddedItemServlet.class file and click **Select**.
4. From the upper-right pane in the Add Files window, select **OrganizeAddedItemServlet.class** and click **Add**. The file appears in the Selected Files list.
5. Click **OK**.
6. Right-click **Web Components** and select **New**.
7. In the New Web Component Window, enter `OrganizeAddedItemServlet` as the component name, select the component type **Servlet** and click **Browse**.
8. In the Select file for Class name window, expand the category **eclient82.war -> WEB-INF**. Select the **OrganizeAddedItemServlet.class** file in the list on the right. Click **OK**.
9. The New Web Component window now looks like Figure 13-18 on page 303.


Component name: * OrganizeAddedItemServlet

Display name:

Description:

Class name: * OrganizeAddedItemServlet

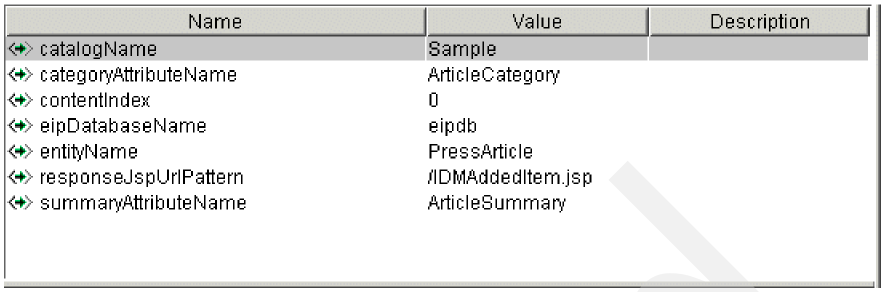
Figure 13-18 Adding new Web component to eClient

10. Click **OK**. The new servlet appears in the components list.
11. Expand the new **OrganizeAddedItemServlet** category on the left, right-click **Initialization Parameters** and select **New**.
12. In the New Initialization Parameter window, enter `eipDatabaseName` as the parameter name and `EIPDB` as the parameter value. Click **Apply**. In the same way, add the remaining parameters of Table 13-1.

Table 13-1 Initialization parameters for new eClient servlet

Parameter name	**Parameter value**
eipDatabaseName	EIPDB
entityName	PressArticle
categoryAttributeName	ArticleCategory
summaryAttributeName	ArticleSummary
contentIndex	0
catalogName	Sample
responseJspUrlPattern	/IDMAddedItem.jsp

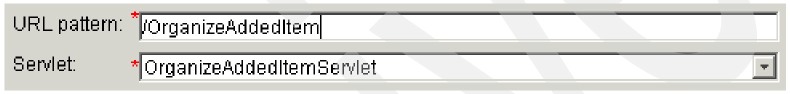
13. Close the New Initialization Parameter window. Now the parameter list looks like Figure 13-19 on page 304.



Name	Value	Description
catalogName	Sample	
categoryAttributeName	ArticleCategory	
contentIndex	0	
eipDatabaseName	eipdb	
entityName	PressArticle	
responseJspUrlPattern	/IDMAddedItem.jsp	
summaryAttributeName	ArticleSummary	

Figure 13-19 Initialization parameters list box

14. Right-click **Servlet Mapping** and select **New**.
15. In the New Servlet Mapping window, enter `/OrganizeAddedItem` as the URL pattern and select the servlet **OrganizeAddedItemServlet** as shown in Figure 13-20.



URL pattern: */OrganizeAddedItem

Servlet: *OrganizeAddedItemServlet

Figure 13-20 Adding URL pattern for new eClient servlet

16. Click **OK**. The new pattern appears in the URL patterns list.
17. Select **File -> Save** to save the .ear file and then click **File -> Close**.

Use the WebSphere Application Server Administrative Console to uninstall eClient and install the new version.

Finally, you need to edit the IDM.properties file in the eClient directory and replace the line:

```
Output.IDMAddItem_out=/IDMAddedItem.jsp
```

with this line:

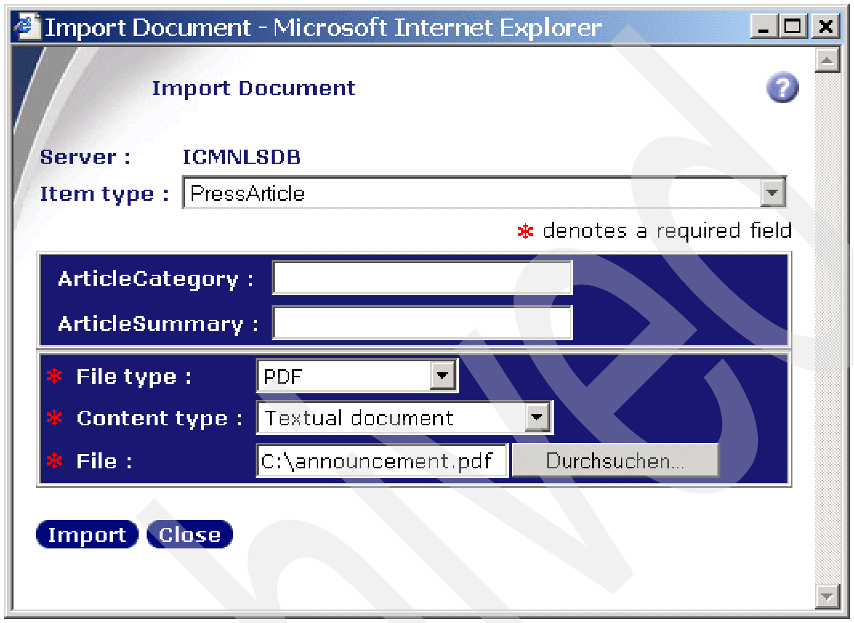
```
Output.IDMAddItem_out=/OrganizeAddedItem
```

Now the eClient control flow is configured according to Figure 13-17 on page 299 and you can start to import documents. If you want to use an item type other than PressArticle, just adapt the servlet initialization parameters accordingly.

13.2.3 Running new eClient import

From the users' perspective, the look and feel of the eClient import has not changed because no servlet or JSP has changed. But now, if you do not specify the value for the category or summary attribute, the appropriate value is created automatically.

The import form for the PressArticle item type looks like Figure 13-21.


Import Document - Microsoft Internet Explorer

Import Document

Server : ICMNLSDB

Item type : PressArticle

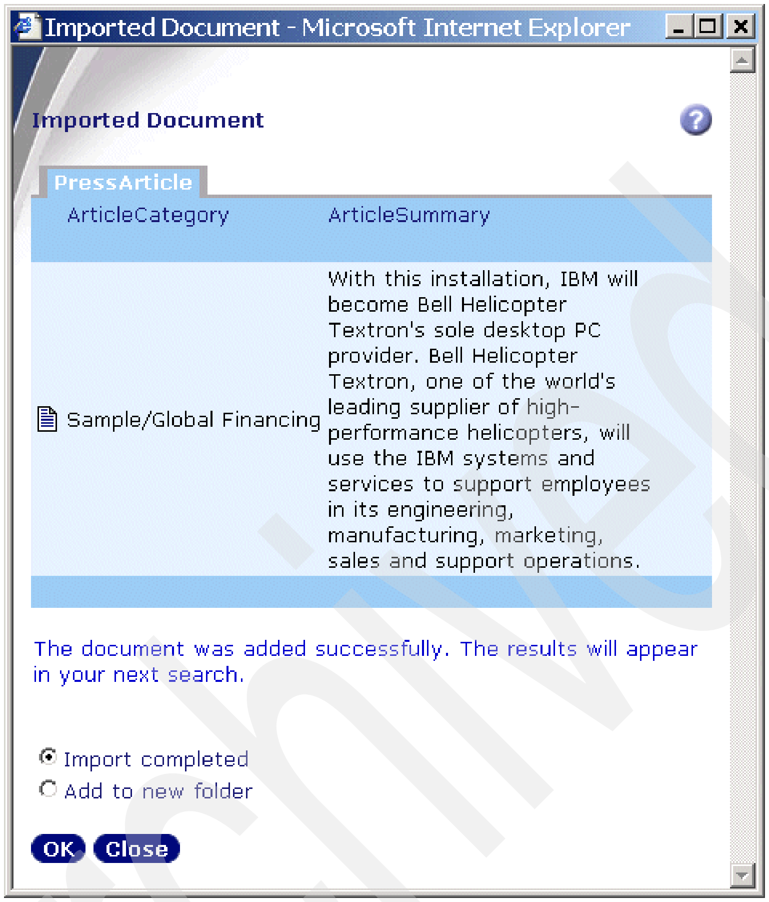
* denotes a required field

ArticleCategory :	
ArticleSummary :	
* File type :	PDF
* Content type :	Textual document
* File :	C:\announcement.pdf

Figure 13-21 Importing document

After you specify the file type, content type and the file to be imported, you can now decide whether you want to specify the category and summary manually or let eClient do it automatically. Just leave the appropriate field blank.

If you submit the form and leave both fields blank, the category and summary information is created, stored in the item and displayed in the confirmation window, as shown in Figure 13-22 on page 306.



ArticleCategory	ArticleSummary
Sample/Global Financing	With this installation, IBM will become Bell Helicopter Textron's sole desktop PC provider. Bell Helicopter Textron, one of the world's leading supplier of high-performance helicopters, will use the IBM systems and services to support employees in its engineering, manufacturing, marketing, sales and support operations.

The document was added successfully. The results will appear in your next search.

[✓ Import completed] Add to new folder

Figure 13-22 Import confirmation with category and summary

If your confirmation window looks different because the summary appears in only one line, you might want to change that in the ItemTable.jsp file as described in 13.1.5, “Customizing eClient” on page 294.

If you want to change the created summary, you can do that with the Edit item attributes function of eClient.

13.3 Searching for related items

When your parametric search is successful and returns a relevant document, the next question typically is “Are there more documents like that?”. When the content is organized using categories/taxonomies, looking for other documents in the same category can often answer this question.

However, related documents can also be found by looking at the terms and expressions in the document. This is useful if there are no predefined categories or the categories do not express the “relatedness” you are looking for.

In this scenario, we add a new action to the eClient search results window that analyzes a selected item of the result list and searches for items related to it using an automatically created text search query.

We use the Information Mining Service of EIP to automatically extract the words to be used in the query from the textual content of an item.

As a result, if the parametric search returns a document for a certain customer name that discusses new information technology in the life sciences business, a query containing the words information, technology, life and sciences (and the customer name) will be created. This query may be restricted to certain categories, yielding the “best of both worlds” when looking for related information.

In the following, we assume the item type PressArticle has been created as described in 13.1.2, “Creating the data model” on page 283 and some items have been created as described in 13.1.3, “Loading the data” on page 286.

The provided solution itself does not depend on this item type. It is customizable and can be used with any other text searchable item type.

13.3.1 Adding new search results action

To add the new action, you need to know how to extend the combo box on the search results window and how to integrate the new search into eClient.

In 5.2.2, “Inspecting eClient control flow” on page 110, we discuss the control flow of a search in eClient. The IDMSearchFrame.jsp is responsible for the overall results window. It calls the IDMSearchToolbar.jsp to create the toolbar containing the combo box with the available actions:

```
<frame noresize title="ResultsToolbar" name="ResultsToolbar" src="<%=
webAppName %>/IDMSearchToolbar.jsp ...
```

You need to change the IDMSearchToolbar.jsp to add the new action to the list.

We show the required changes for this scenario below. A general description of how to add a new action to the window can be found in Chapter 11, “Adding custom functions to the search results window” on page 239.

According to the Model-View-Controller design of eClient (see Chapter 5., “eClient architecture” on page 101), the new action needs to be implemented in a new servlet that is called when you select the action on the search results window.

If you want to work with the related documents in the same way as with arbitrary eClient search results, you need to use the appropriate eClient JSPs to create the results window. That means you have to provide the search results in the session object, where the JSPs can pick them up (see Figure 5-7 on page 111). You do not need to deal with the result processing at all if you directly reuse the IDMSearch servlet.

Figure 13-23 shows how the new servlet fits into the search control flow.

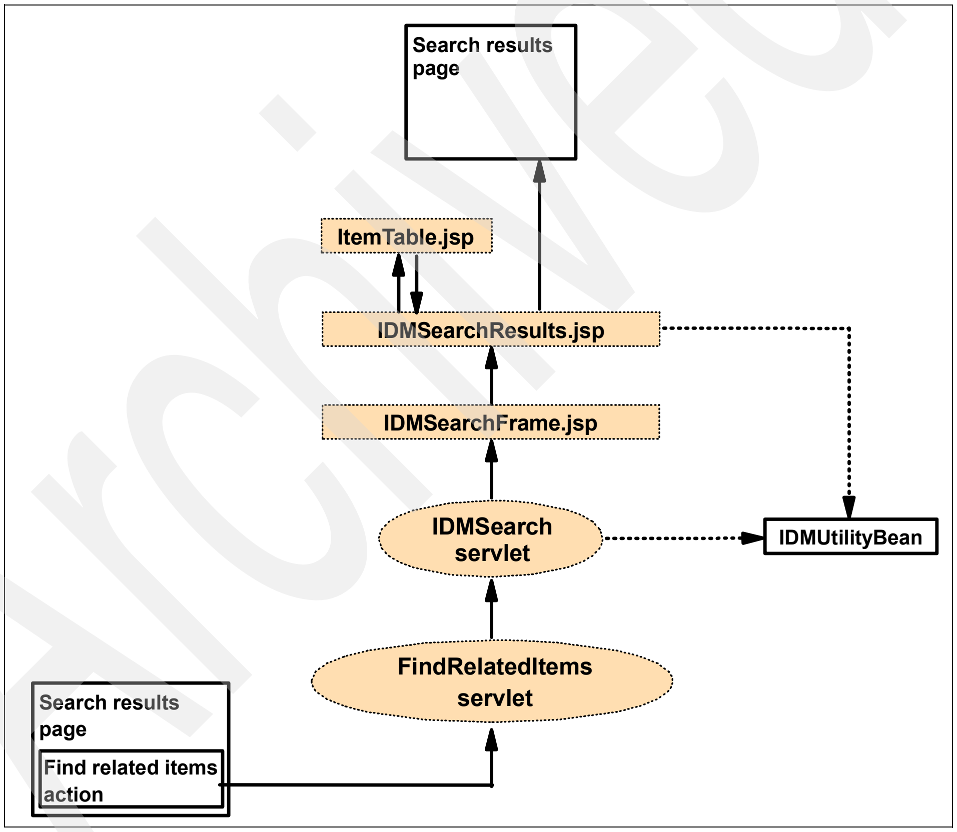


Figure 13-23 Reusing search IDMSearch servlet

The new FindRelatedItems servlet gets called if you select the action on the search results window, analyzes the selected document using the Information Mining Service, creates a new query and calls the IDMSearch servlet to process the query. The IDMSearch servlet then triggers the creation of the results window as usual.

In a manual search, the IDMSearch servlet gets called by the IDMBasicSearch.jsp. The FindRelatedItems servlet has to use the same format in its request to the IDMSearch servlet. However, you do not need to inspect the JSP code to find out how the request looks. You can change the form in the JSP, so that it calls the WebSphere Application Server snoop servlet:

```
<FORM NAME="searchCriteria" action="http://localhost/snoop" Method="Get">
```

This way, you find out that if you search for the words ibm, learning and services using the PressArticle item type, the required URL looks like:

```
/IDMSearch?Document%2BContents=ibm+learning+services&attrCount=0&radioAnyAl
l=ANY&NewWindow=true&Entity=PressArticle
```

13.3.2 Changing JSPs

You need to change the IDMSearchToolbar.jsp to add the new action to the combo box of the search results window and pageComponents/Heading.jsp to implement the JavaScript function that gets called if the new action has been selected.

Both JSPs are provided in the redbook samples.

In IDMSearchToolbar.jsp, we add the new option to the combo box (only if the datastore is Content Manager):

```
<option value="FindRelatedItems">Find Related Items</option>
```

As is done for the other actions, we also add an entry to the switch statement in the doSelected method that calls the JavaScript method findRelatedItems to be defined in Heading.jsp:

```
case 'FindRelatedItems':
parent.ResultsBottom.findRelatedItems()
break;
```

In pageComponents/Heading.jsp, we implement a new findRelatedItems function. In the function, we first determine the item that has been selected in the search result list:

```
if (document.results.check1[i].checked == true) {
pidString = document.results.check1[i].value;
```

Then we create a URL that calls the new FindRelatedItems servlet. The servlet needs to know the PID of the document to be analyzed and the entity (item type) name to create a proper URL to call the IDMSearch servlet:

```
var url = '<%= webAppName %>/FindRelatedItems?entityName=<%= entity
%>&pidString='+pidString;
```

Finally a new window is opened and the control is forwarded to the FindRelateditems servlet to perform the analysis and search:

```
window.open( url, wn, 'resizable=1,scrollbars=1,height=600,width=900');
```

The update of eClient is described in the following section.

13.3.3 Implementing custom servlet

The FindRelatedItems servlet is provided in the redbook samples as FindRelatedItems.java.

In the init method, the implementation first reads the servlet initialization parameters:

- eipDatabaseName, used to connect to the Information Mining Service.
- contentIndex, the index of the content part in the item.
- maxQueryLengthFeatures, the maximum number of extracted features to be used in the query. A feature can consist of multiple words.

These parameters make the servlet adaptable to other databases.

Both service methods, doGet and doPost, call processRequest. Hence, the servlet behavior is the same for both request types.

In the processRequest method, the entity name and the PID of the item to be analyzed is retrieved from the request object:

```
String entityName = req.getParameter("entityName");
String pidString = req.getParameter("pidString");
```

The connection object is retrieved from the session object:

```
CMBConnection connection =
(CMBConnection)req.getSession().getAttribute("connection");
```

In order to use the Information Mining Service, it needs to be connected to the EIP database. One way to do that is to connect a CMBConnection that has the connectToIKF property set to true. We use the same user ID as the connection read from the session:

```
eipConnection = new CMBConnection();
eipConnection.setServerName(eipDatabaseName);
eipConnection.setUserid(connection.getUserid());
eipConnection.setPassword(connection.getPassword());
eipConnection.setConnectToIKF(true);
eipConnection.connect();
```

The service object can then be retrieved from the connection:

```
ikfService = eipConnection.getIKFService();
```

Then the item is retrieved from the Content Manager. To retrieve the original document, the content part with the index specified in the servlet initialization parameters is used:

```
CMBItem item = new CMBItem(pidString);
item.setConnection(currentConnection);
dataManagement.setDataObject(item);
dataManagement.retrieveItem();
CMBObject content = dataManagement.getContent(contentIndex);
```

We call the document filter to create the text document object required for the analysis steps:

```
DKIKFTextDocument ikfDocument =
ikfFilter.getTextDocument(content.getDataStream());
```

Now the language identifier gets called, because information extraction requires the language of the document:

```
DKIKFLanguageIdentificationResult langResult[] =
langIdentifier.analyze(ikfDocument);
```

The language is set on the document object:

```
ikfDocument.setLanguage(language);
```

And the information extractor is called to extract terms from the document:

```
infoExtractor.setFeatureTypes(DKIKFFeature.TYPE_TERM);
DKIKFFeature[] features = infoExtractor.analyze(ikfDocument).getFeatures();
```

Up to the specified maximum number of extracted features are stored in a temporary query string:

```
int n = Math.min(maxQueryLengthFeatures, features.length);
for(int i = 0; i < n; i++) {
queryString.append(features[i].getString() + " ");
}
```

Then the final query string is created by adding a plus sign before each extracted word. This results in a basic text search query as described in the EIP programming guide (select **Working with Content Manager Version 8.2 -> Understanding text search -> Understanding text search syntax**) where each of the specified words must occur in the document:

```
StringTokenizer tokenizer = new
StringTokenizer(queryString.toString().trim());
queryString = new StringBuffer();
int tokenCount = tokenizer.countTokens();
for(int i = 0; i < tokenCount; i++) {
String token = tokenizer.nextToken();
queryString.append(" +" + token);
}
```

Finally the URL for the IDMSearch servlet is created:

```
String url = "/IDMSearch?Document%2BContents=" +
URLEncoder.encode(queryString.toString().trim()) +
"&attrCount=0&radioAnyAll=ANY&NewWindow=true&Entity=" + entityName;
```

And the request is forwarded to the IDMSearch servlet:

```
getServletContext().getRequestDispatcher(url).forward(req, resp);
```

To compile the servlet:

1. Open an EIP development window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**.
2. Change to the directory with the redbook samples for this chapter.
3. Run **`javac FindRelatedItemsServlet.java`**

After the JSPs and the servlet are implemented, they need to be added to the eClient. If you use the WebSphere Application Server Application Assembly Tool, do the following steps:

1. Open the eClient .ear file using the WebSphere Application Server Application Assembly Tool and expand the category **Web Modules -> eclient82 -> Files**. You see class files, JAR files and resource files.
2. Right-click **Resource Files** and select **Add Files**. The Add Files window appears.
3. Click **Browse**. Select the directory that contains the redbook samples for this chapter (it must appear in the File name field) and click **Select**.
4. From the upper-right pane in the Add Files window, select **IDMSearchToolbar.jsp** and the **pageComponents** directory containing Heading.jsp. Click **Add**. The JSP files appear in the Selected Files list.

5. Click **OK** and select **Yes** on the confirmation window to overwrite the existing JSP files.
6. Right-click **Class Files** and select **Add Files**. The Add Files window appears.
7. Click **Browse**. Select the directory that contains the redbook samples for this chapter (it must appear in the File name field) and click **Select**.
8. From the upper-right pane in the Add Files window, select **FindRelatedItemsServlet.class** and click **Add**. The file appears in the Selected Files list.
9. Click **OK**.
10. Right-click **Web Components** and select **New**.
11. In the New Web Component window, enter `FindRelatedItemsServlet` as the component name, select the component type **Servlet** and click **Browse**.
12. In the Select file for Class name window, expand the category **eclient82.war -> WEB-INF**. Select the **FindRelatedItemsServlet.class** file in the list on the right. Click **OK**.
13. The New Web Component window now looks like Figure 13-18 on page 303.

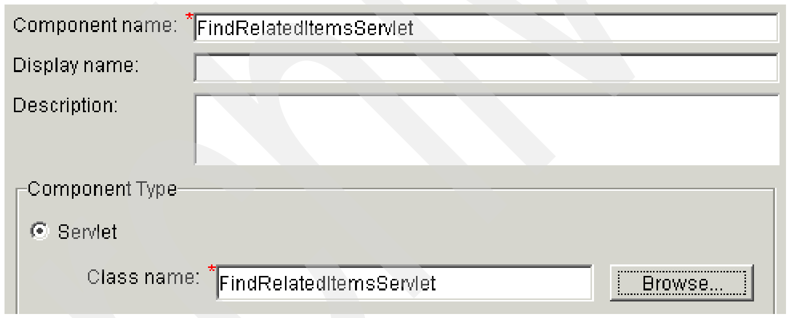

Component name: * FindRelatedItemsServlet
Display name:
Description:
Component Type
(●) Servlet
Class name: * FindRelatedItemsServlet

Figure 13-24 Adding new Web component FindRelatedItemsServlet

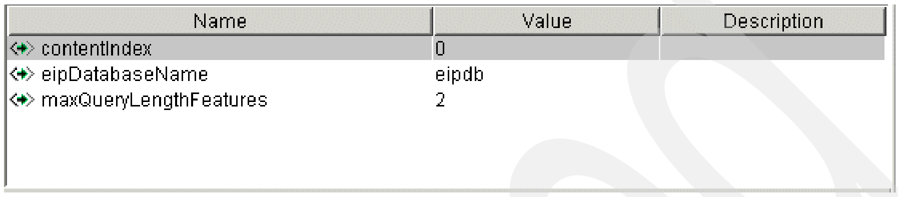
14. Click **OK**. The new servlet appears in the components list.
15. Expand the new **FindRelatedItemsItemServlet** category on the left, right-click **Initialization Parameters** and select **New**.
16. In the New Initialization Parameter window, enter `eipDatabaseName` as the parameter name and `EIPDB` as the parameter value. Click **Apply**. In the same way, add the remaining parameters of Table 13-1 on page 303.

Table 13-2 Initialization parameters for FindRelatedItemsServlet

Parameter name	Parameter value
eipDatabaseName	EIPDB

Parameter name	Parameter value
contentIndex	0
maxQueryLengthFeatures	2

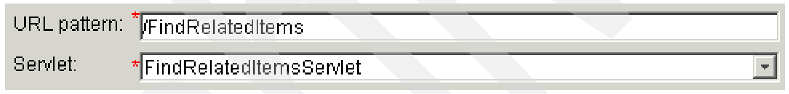
17. Close the New Initialization Parameter window. Now the parameter list looks like Figure 13-19 on page 304.



Name	Value	Description
contentIndex	0	
eipDatabaseName	eipdb	
maxQueryLengthFeatures	2	

Figure 13-25 Initialization parameters list box

18. Right-click **Servlet Mapping** and select **New**.
19. In the New Servlet Mapping window, specify `/FindRelatedItems` as the URL pattern and select the **servlet FindRelatedItemsServlet** as shown in Figure 13-20 on page 304.



URL pattern: * /FindRelatedItems

Servlet: * FindRelatedItemsServlet

Figure 13-26 Creating URL pattern for FindRelatedItemsServlet

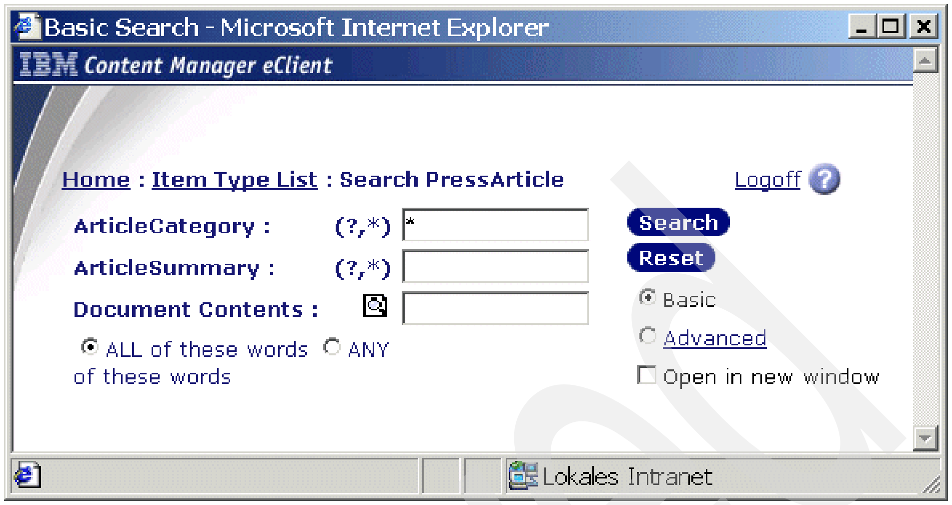
20. Click **OK**. The new pattern appears in the URL patterns list.
21. Select **File -> Save** to save the .ear file and then select **File -> Close**.

Use the WebSphere Application Server Administrative Console to uninstall eClient and install the new version.

The new Find related items action is now ready to be used.

13.3.4 Running new action

To run the new search results action, you first need to perform a search. Log on to eClient with the database name ICMNLSDB and the Content Manager connector (CM8). Perform a search with the PressArticle item type, for example a parametric search as shown in Figure 13-27 on page 315.


Basic Search - Microsoft Internet Explorer

IBM Content Manager eClient

Home : Item Type List : Search PressArticle

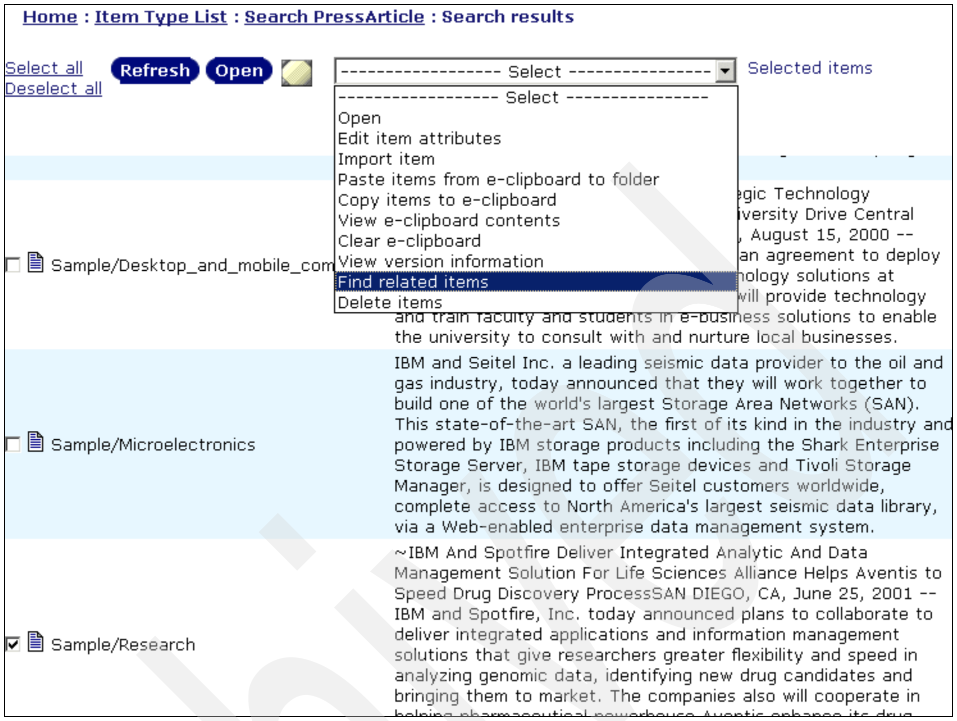
Logoff

ArticleCategory :	(?,*)	*
ArticleSummary :	(?,*)	
Document Contents :		

[✓ ALL of these words] ANY of these words

Figure 13-27 Running parametric search

If you specify * in the ArticleCategory field and click **Search**, you get a result list containing all press articles in the database. You can now select a document and then select **Find related items** action from the combo box in the toolbar as shown in Figure 13-28 on page 316.


Home : Item Type List : Search PressArticle : Search results

Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items
------------------ Select ----------------
Open
Edit item attributes
Import item
Paste items from e-clipboard to folder
Copy items to e-clipboard
View e-clipboard contents
Clear e-clipboard
View version information
Find related items
Delete items

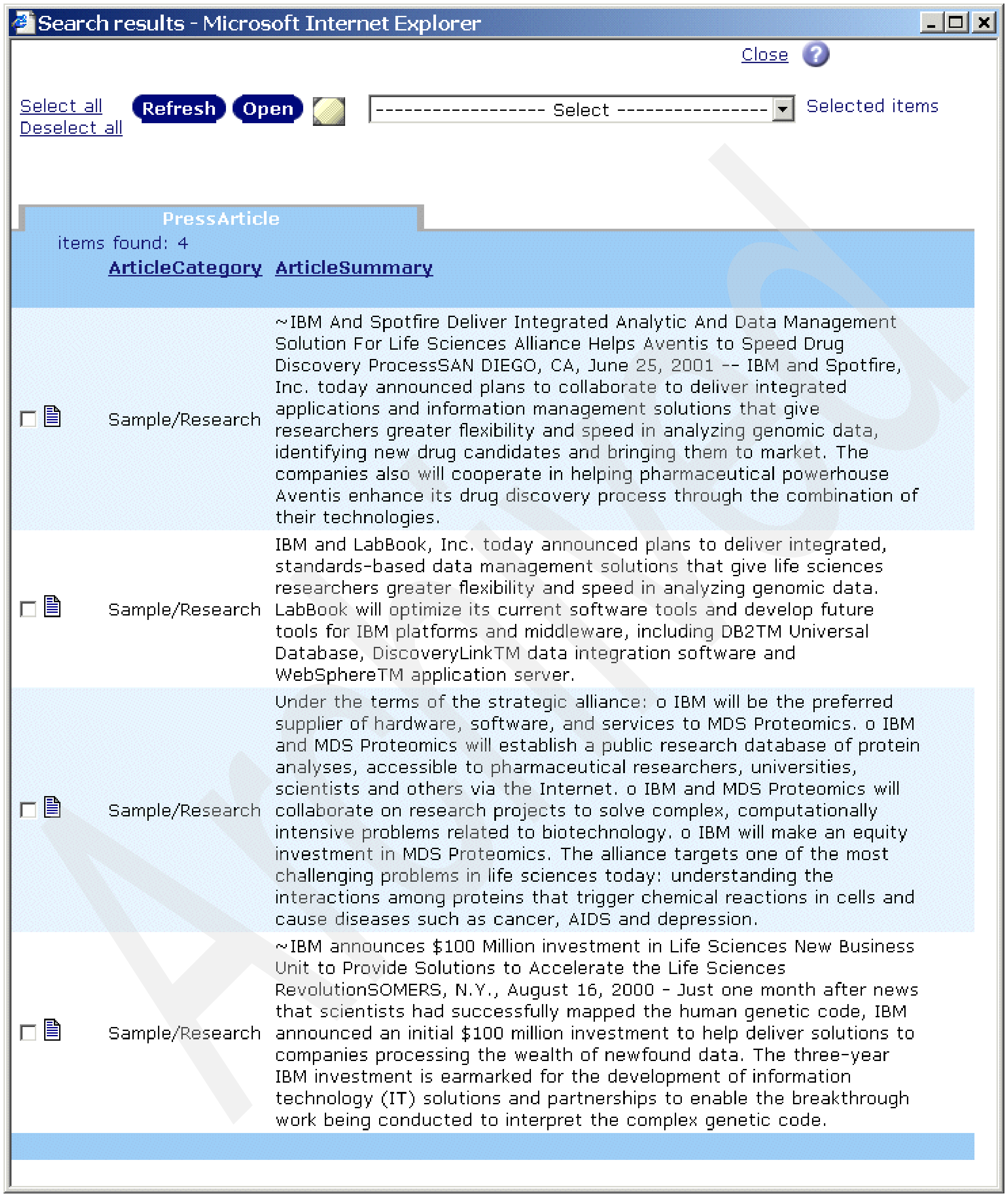
☐ Sample/Desktop_and_mobile_com	gic Technology iversity Drive Central , August 15, 2000 -- an agreement to deploy nology solutions at will provide technology and train faculty and students in e-business solutions to enable the university to consult with and nurture local businesses.
☐ Sample/Microelectronics	IBM and Seitel Inc. a leading seismic data provider to the oil and gas industry, today announced that they will work together to build one of the world's largest Storage Area Networks (SAN). This state-of-the-art SAN, the first of its kind in the industry and powered by IBM storage products including the Shark Enterprise Storage Server, IBM tape storage devices and Tivoli Storage Manager, is designed to offer Seitel customers worldwide, complete access to North America's largest seismic data library, via a Web-enabled enterprise data management system.
☑ Sample/Research	~IBM And Spotfire Deliver Integrated Analytic And Data Management Solution For Life Sciences Alliance Helps Aventis to Speed Drug Discovery ProcessSAN DIEGO, CA, June 25, 2001 -- IBM and Spotfire, Inc. today announced plans to collaborate to deliver integrated applications and information management solutions that give researchers greater flexibility and speed in analyzing genomic data, identifying new drug candidates and bringing them to market. The companies also will cooperate in

Figure 13-28 Running find related items for document

The related items are listed in a new window, as shown in Figure 13-29 on page 317.

Because we reused the search servlet and JSPs, this is a real eClient search result and you can apply all available functions to the items.

In the provided solution, the original item is part of the related items list.

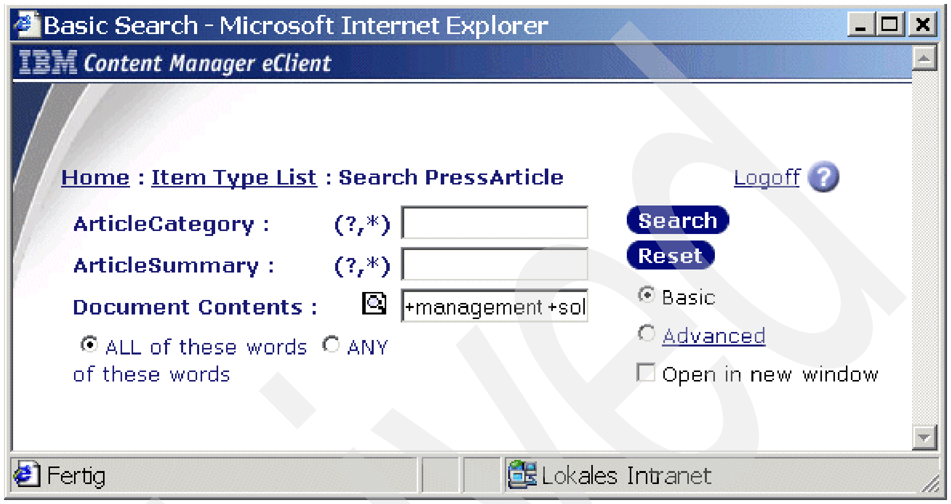

Search results - Microsoft Internet Explorer
Close
Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items
PressArticle

items found: 4

	ArticleCategory	ArticleSummary
☐	Sample/Research	~IBM And Spotfire Deliver Integrated Analytic And Data Management Solution For Life Sciences Alliance Helps Aventis to Speed Drug Discovery ProcessSAN DIEGO, CA, June 25, 2001 -- IBM and Spotfire, Inc. today announced plans to collaborate to deliver integrated applications and information management solutions that give researchers greater flexibility and speed in analyzing genomic data, identifying new drug candidates and bringing them to market. The companies also will cooperate in helping pharmaceutical powerhouse Aventis enhance its drug discovery process through the combination of their technologies.
☐	Sample/Research	IBM and LabBook, Inc. today announced plans to deliver integrated, standards-based data management solutions that give life sciences researchers greater flexibility and speed in analyzing genomic data. LabBook will optimize its current software tools and develop future tools for IBM platforms and middleware, including DB2TM Universal Database, DiscoveryLinkTM data integration software and WebSphereTM application server.
☐	Sample/Research	Under the terms of the strategic alliance: o IBM will be the preferred supplier of hardware, software, and services to MDS Proteomics. o IBM and MDS Proteomics will establish a public research database of protein analyses, accessible to pharmaceutical researchers, universities, scientists and others via the Internet. o IBM and MDS Proteomics will collaborate on research projects to solve complex, computationally intensive problems related to biotechnology. o IBM will make an equity investment in MDS Proteomics. The alliance targets one of the most challenging problems in life sciences today: understanding the interactions among proteins that trigger chemical reactions in cells and cause diseases such as cancer, AIDS and depression.
☐	Sample/Research	~IBM announces $100 Million investment in Life Sciences New Business Unit to Provide Solutions to Accelerate the Life Sciences RevolutionSOMERS, N.Y., August 16, 2000 - Just one month after news that scientists had successfully mapped the human genetic code, IBM announced an initial $100 million investment to help deliver solutions to companies processing the wealth of newfound data. The three-year IBM investment is earmarked for the development of information technology (IT) solutions and partnerships to enable the breakthrough work being conducted to interpret the complex genetic code.

Figure 13-29 List of related items

If you want to review or work with the automatically created query, you can click the **Search PressArticle** link at the top of the first search results window. This opens the search form for the item type where the query is displayed in the Document Contents field as shown in Figure 13-30.



Basic Search - Microsoft Internet Explorer

IBM Content Manager eClient

Home : Item Type List : Search PressArticle   Logoff

ArticleCategory :	(?,*)		Search
ArticleSummary :	(?,*)		Reset
Document Contents :		+management +sol	[✓ Basic] Advanced

[✓ ALL of these words] ANY of these words

☐ Open in new window

Fertig   Lokales Intranet

Figure 13-30 Automatically created query

Although this is a very simple implementation, the automatically created queries created for the press articles are good. For example, to find related documents, the servlet generates queries as follows:

- +information +technology +life +science
- +open +source +project +enterprise +system
- +cancer +treatment +information +system
- +ThinkPad +notebook +small +business
- +golf +club +product +development

13.4 Organizing existing items in Content Manager

In 13.1, "Using categories and summaries in eClient searches" on page 282, we discuss how you can prepare the data model and organize your items during upload by creating category and summary information.

It is most likely that the items are already stored in Content Manager and you want to organize them afterwards.

The OrganizeItems application allows you to create category, summary and language information for existing items. You can find the application in the organizeItems directory in the samples for this chapter.

Two steps are required to organize existing items:

1. Extend the item type with the attributes to store the additional information.
2. Run the OrganizeItems application.

Both steps are described in the following section. In the provided samples, we use the PressArticle item type and the items created in 13.1, "Using categories and summaries in eClient searches" on page 282.

The OrganizeItems application works with arbitrary item types.

13.4.1 Extending item type

Before you can use OrganizeItems, you need to extend the item type with the attributes to store the additional information. For example, if you want to add summary information to the items, you need to add a summary attribute to the item type. This can be done with the Content Manager System Administration Client. The attribute names can then be specified in the OrganizeItems.properties file.

For the PressArticle item type, the attributes for category and summary information are already there. As an example, we add an attribute for the language information:

1. Using the System Administration Client, log on to the Content Manager database ICMNLSDB.
2. Expand the **ICMNLSDB -> Data Modeling -> Item Types** category, right-click the **PressArticle** item type and select **Properties**.
3. In the Item Type Properties window, select the **Attributes** tab and click the icon on the left to open the New Attribute window (or press Alt+t).
4. In the New Attribute window, enter the name `ArticleLanguage`, attribute type `Character`, and a character length of 5. Click **OK**.
5. Back in the New Item Type Definition window, select the new attribute **ArticleLanguage** from the list of available attributes and click **Add**. The Attributes tab should now look like Figure 13-31 on page 320.

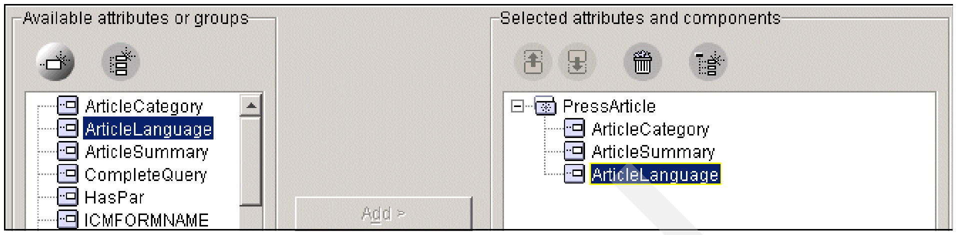


Figure 13-31 Adding new ArticleLanguage attribute

6. Click **OK** to update the item type.

For an example of how to add attributes for category and summary information, refer to 13.1.2, “Creating the data model” on page 283.

13.4.2 Running OrganizeItems application

The OrganizeItems application resides in the organizeItems directory in the samples directory for this chapter.

You need to edit the OrganizeItems.properties file to configure the application. It is also available in the samples directory.

Example 13-1 shows the content of the provided file.

Example 13-1 Content of OrganizeItems.properties

```
contenManagerServerName=icmnlsdb
eipServerName=eipdb
userId=icmadmin
password=password

query=(/PressArticle)

contentIndex=0

categoryAttributeName=
summaryAttributeName=ArticleSummary
languageAttributeName=ArticleLanguage

infoMiningCatalogName=Sample

infoMiningSummaryMaxLength=2
```

It consists of the following properties:

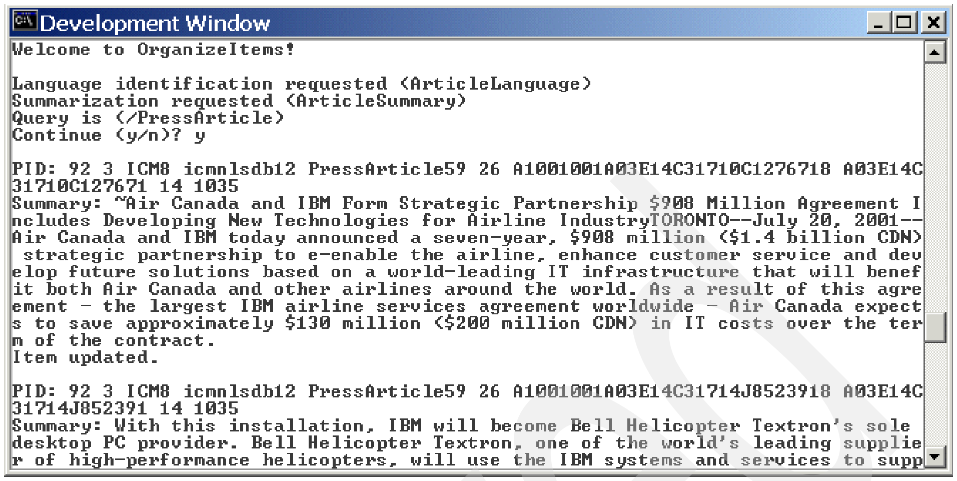
- **contenManagerServerName** is the name of the Content Manager server that contains the items to be updated.
- **eipServerName** is the name of the EIP server required to run the Information Mining Service.
- **userId** and **password** are used to connect to the specified servers.
- **query** is a XQuery Path Expressions (XQPE) that is used to search for the items to be updated. Information about the query language can be found in the EIP Information Center by selecting **Programming -> Working with Content Manager Version 8.2 -> Understanding the query language**.
- **contentIndex** specifies the index of the content object in the item.
- **categoryAttributeName**, **summaryAttributeName** and **languageAttributeName** specify the names of the attributes to store the appropriate information. You can leave the value blank if you do not want to create and store the information.
- **infoMiningCatalogName** specifies the name of the Information Mining Service catalog to be used for categorization. You need to specify a value only if you specified a value for the categoryAttributeName.
- **infoMiningSummaryMaxLength** specifies the maximum length of the summaries in sentences. You need to specify a value only if you specified a value for the summaryAttributeName.

To compile and run the OrganizeItems application:

1. Open a command window and switch to the organizeItems directory in the samples directory for this chapter.
2. To compile the application, run `compile`.
3. To run the application, enter `run`.

If you are using the OrganizeItems.properties file shown in Example 13-1 on page 320, each PressArticle item is updated with a new summary and language information. The category is not created because no value has been specified for the categoryAttributeName. The specified query returns all items of item type PressArticle.

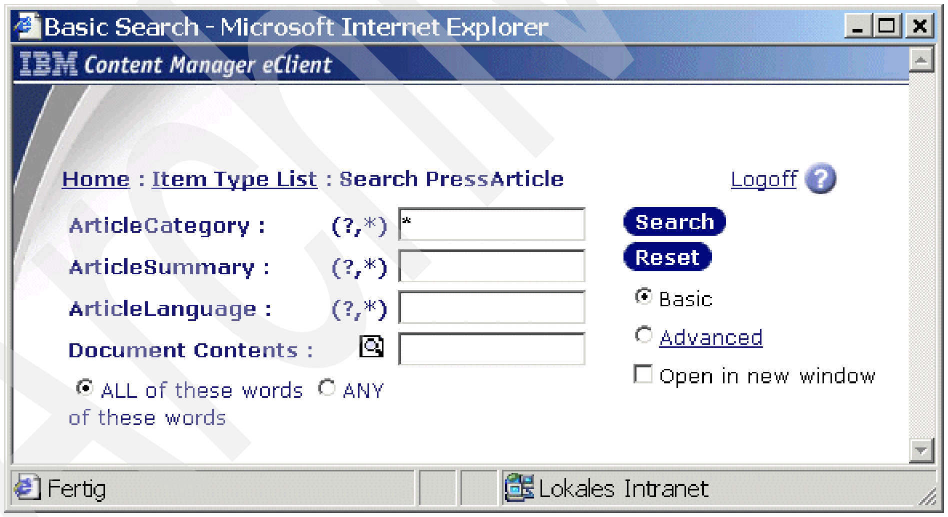
The application prints the created information as shown in Figure 13-32 on page 322.



```
Welcome to OrganizeItems!

Language identification requested (ArticleLanguage)
Summarization requested (ArticleSummary)
Query is (/PressArticle)
Continue (y/n)? y

PID: 92 3 ICM8 icmnlsdb12 PressArticle59 26 A1001001A03E14C31710C1276718 A03E14C
31710C127671 14 1035
Summary: ~Air Canada and IBM Form Strategic Partnership $908 Million Agreement I
ncludes Developing New Technologies for Airline IndustryTORONTO--July 20, 2001--
Air Canada and IBM today announced a seven-year, $908 million ($1.4 billion CDN)
 strategic partnership to e-enable the airline, enhance customer service and dev
elop future solutions based on a world-leading IT infrastructure that will benef
it both Air Canada and other airlines around the world. As a result of this agre
ement - the largest IBM airline services agreement worldwide - Air Canada expect
s to save approximately $130 million ($200 million CDN) in IT costs over the ter
m of the contract.
Item updated.

PID: 92 3 ICM8 icmnlsdb12 PressArticle59 26 A1001001A03E14C31714J8523918 A03E14C
31714J852391 14 1035
Summary: With this installation, IBM will become Bell Helicopter Textron's sole
desktop PC provider. Bell Helicopter Textron, one of the world's leading supplie
r of high-performance helicopters, will use the IBM systems and services to supp
```

Figure 13-32 Output of OrganizeItems

Now, if you search with the PressArticle item type in eClient, the search form also contains the language attribute as shown in Figure 13-33.


Basic Search - Microsoft Internet Explorer
IBM Content Manager eClient
Home : Item Type List : Search PressArticle
Logoff
Search
Reset
Fertig
Lokales Intranet

ArticleCategory :	(?,*)	*
ArticleSummary :	(?,*)	
ArticleLanguage :	(?,*)	
Document Contents :		

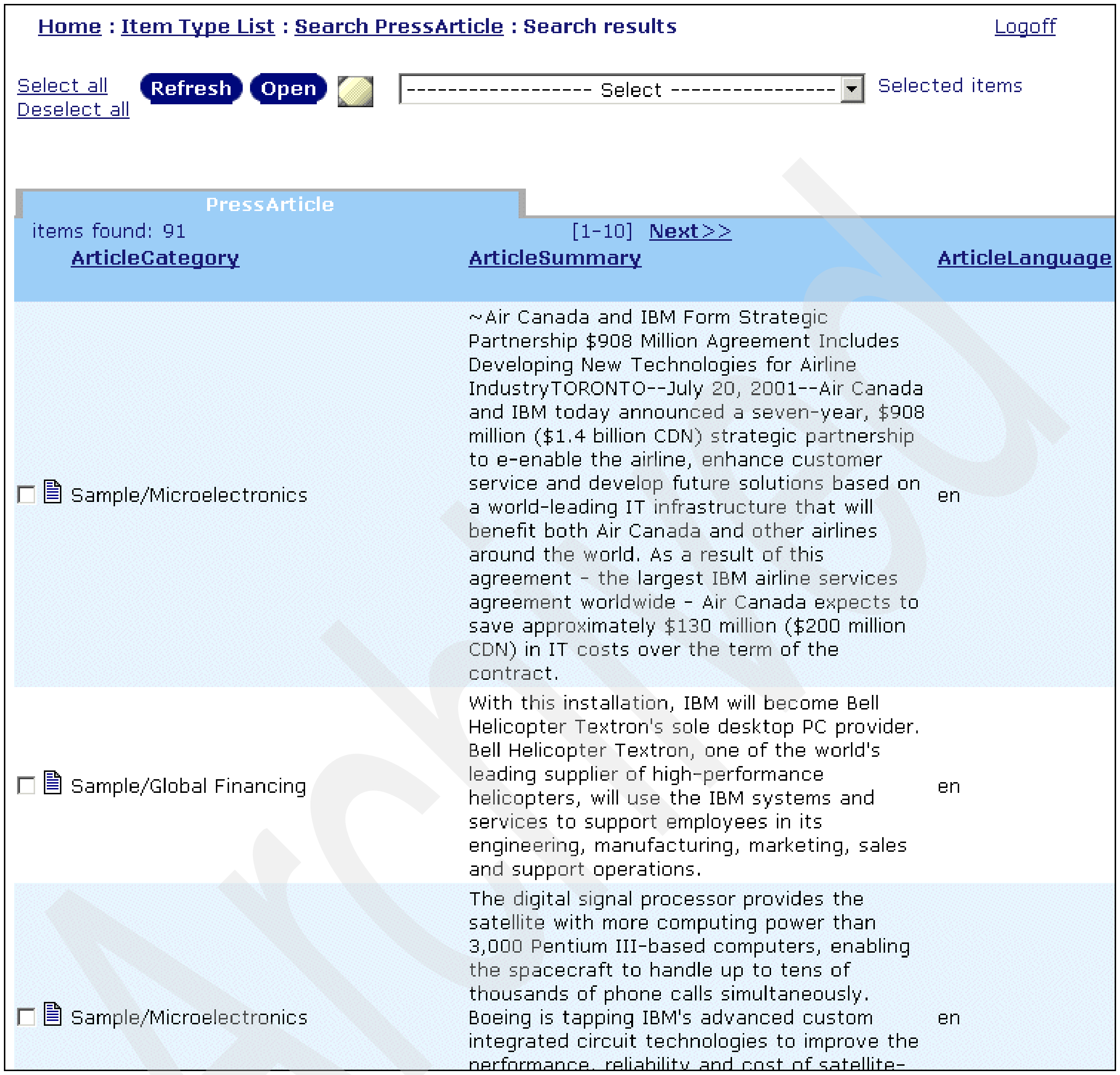
[✓ ALL of these words] ANY of these words

[✓ Basic] Advanced

Open in new window

Figure 13-33 Search form for PressArticle with language attribute

In the eClient Search results window, the values for the ArticleLanguage attribute are displayed in addition to the values for ArticleCategory and ArticleSummary. The results window is shown in Figure 13-34 on page 323.


Home : Item Type List : Search PressArticle : Search results   Logoff

Select all
Deselect all
Refresh Open ------------------ Select ----------------- Selected items

PressArticle

items found: 91   [1-10] Next>>

ArticleCategory	ArticleSummary	ArticleLanguage
☐ Sample/Microelectronics	~Air Canada and IBM Form Strategic Partnership $908 Million Agreement Includes Developing New Technologies for Airline IndustryTORONTO--July 20, 2001--Air Canada and IBM today announced a seven-year, $908 million ($1.4 billion CDN) strategic partnership to e-enable the airline, enhance customer service and develop future solutions based on a world-leading IT infrastructure that will benefit both Air Canada and other airlines around the world. As a result of this agreement - the largest IBM airline services agreement worldwide - Air Canada expects to save approximately $130 million ($200 million CDN) in IT costs over the term of the contract.	en
☐ Sample/Global Financing	With this installation, IBM will become Bell Helicopter Textron's sole desktop PC provider. Bell Helicopter Textron, one of the world's leading supplier of high-performance helicopters, will use the IBM systems and services to support employees in its engineering, manufacturing, marketing, sales and support operations.	en
☐ Sample/Microelectronics	The digital signal processor provides the satellite with more computing power than 3,000 Pentium III-based computers, enabling the spacecraft to handle up to tens of thousands of phone calls simultaneously. Boeing is tapping IBM's advanced custom integrated circuit technologies to improve the performance, reliability and cost of satellite-	en

Figure 13-34 New language attribute in eClient search result

14

Invoking eClient from another application

In this chapter, we integrate eClient with an existing application. Specifically, we show how to enable any application that has the capability of launching a Web browser to launch eClient, log on to EIP, and display search results for search criteria specified in a URL.

This chapter includes the following sections:

- ► Overview
- ► Servlet source code
- ► Configuring and using the servlet
- ► Servlet URL syntax and source
- ► Invoking the servlet from an application

14.1 Overview

There are many software application integration scenarios where users of an existing application need an easy way to view content available through eClient. For example, users of an insurance claims processing application may use a custom-built application to locate basic claim information from a database regarding insurance claims. After locating the claims, they may want to have an easy way to view documents related to the claim (claim forms, photos, policy documents, etc.) in eClient.

If using out-of-the box eClient to access the related documents, users have to launch eClient, log on, select search templates, and enter search criteria (such as claim numbers) to locate the documents. Users would have a better experience if they could simply select a custom button or menu from their Line of Business (LOB) application that would automatically show them a list of documents in an eClient window for the claims they are looking for without having to explicitly log on and enter the search criteria. The LOBIntegrator servlet and some custom code in the LOB application can provide this type of functionality.

The key component required to implement this functionality is the LOBIntegrator servlet. This servlet provides a new URL that is part of the eClient application and can be invoked with specific parameters to specify logon and search parameters. The servlet parses the parameters from the URL and automatically logs on and performs a search. The search results are displayed in the standard eClient window, allowing users to view and manipulate the documents.

The application you are integrating with may be written in one of many languages. For example, it may be written in Java, Visual Basic, C++, FoxPro, PowerBuilder, or perhaps it is a 3270 application running in an emulator. The LOBIntegrator servlet can be used with any of these types of applications. Any application that can launch a Web browser and pass a URL to it can use the functionality of the servlet. Later in this chapter, we provide more information on how to invoke the LOBIntegrator servlet from your LOB application.

14.2 Servlet source code

This section lists the source code for the LOBIntegrator servlet. The servlet can be added to eClient to allow other applications to perform automatic searches by generating a correctly formatted URL.

If you are integrating eClient with another J2EE application, you can add this servlet to your custom J2EE application instead of adding it to eClient. Either way, it does the same thing. Adding it to your custom J2EE application may be better because you avoid any customization of eClient.

Example 14-1 includes the complete source code for LOBIntegrator servlet. You can download a soft copy of the source code from Web. See Appendix B, “Additional material” on page 465 for download instructions.

The source code is fairly simple. The bulk of the code is in the ProcessRequest() method. It gets the parameters from the URL, performs the initialization and logon if necessary, and builds a URL for the standard IDMSearch servlet in eClient to perform the search and display the search results.

Example 14-1 LOBIntegrator servlet

```
/*
  This 'as is' sample source code is provided with the IBM Redbook
  SG246964.  The source code is for a Servlet that can be added to
  the eClient to allow an application to launch a browser with a
  specific URL that will logon, perform a search, and display the
  search results in the browser window.
*/

import java.io.IOException;
import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import java.net.URLEncoder;
import javax.servlet.http.HttpSession;

import com.ibm.mm.beans.CMBConnection;

public class LOBIntegrator extends HttpServlet {

   public void init() throws ServletException {
      super.init();
   }

   public void doGet(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {
        processRequest(req, resp);
   }

   public void doPost(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {
        processRequest(req, resp);
   }

   private String convertString(String strValue)
   {
         String strNewValue = "";

         for (int i = 0; i < strValue.length(); i++)
         {
            if (strValue.charAt(i) == ' ')
            {
               strNewValue = strNewValue + "%2B";
            }
            else if (strValue.charAt(i) == '(')
            {
               strNewValue = strNewValue + "%28";
            }
            else if (strValue.charAt(i) == ')')
            {
               strNewValue = strNewValue + "%29";
            }
            else
            {
               strNewValue = strNewValue + strValue.charAt(i);
            }
         }

         System.out.println("Old: " + strValue);
         System.out.println("New: " + strNewValue);

         return strNewValue;
      }

       public void processRequest(HttpServletRequest request, HttpServletResponse
response)
       throws ServletException, IOException {

           try {
            String strTemp = "";
            String url = "";

            // Connection parameters and default values
            String strServerName = "ICMNLSDB";
            String strServerType = "ICM";
            String strUserID = "icmadmin";
            String strPassword = "password";

            // Search parameters
            String strSearchType    = "";
            String strSearchEntity  = "";
            String strSearchAttr1   = "";
            String strSearchValue1  = "";

            //
```

```
          // Get parameters from the request
          //
          strTemp = request.getParameter("userid");
          if (strTemp != null && strTemp.length() > 0)
          {
             strUserID = strTemp;  // default is icmadmin
          }

          strTemp = request.getParameter("password");
          if (strTemp != null && strTemp.length() > 0)
          {
             strPassword = strTemp;  // default is password
          }

          strTemp = request.getParameter("serverName");
          if (strTemp != null && strTemp.length() > 0)
          {
             strServerName = strTemp;// default is ICMNLSDB
          }

          strTemp = request.getParameter("serverType");
          if (strTemp != null && strTemp.length() > 0)
          {
             strServerType = strTemp;// default is ICM
          }

          //
          // Make sure we have a session.  If not, call IDMInit and
          // set URLOut so this servlet will be called afterwards
          //
          HttpSession session = request.getSession(false);
          if(session == null) {
             System.out.println("session is null, calling IDMInit");
              request.setAttribute("URLout","/LOBIntegrator");

getServletContext().getRequestDispatcher("/IDMInit").forward(request,response);
          }

          //
          // Make sure we have a valid connection.  If not, call IDMConnection
          // and set URLOut so this servlet will be called afterwards
          //
          CMBConnection connection = (CMBConnection)
session.getAttribute("connection");

          if((connection == null) || (!connection.isConnected())) {
              request.setAttribute("URLout","/LOBIntegrator");

              url = "/IDMConnection?server=" +
                    URLEncoder.encode("name=" + strServerName +
                    ",type=" + strServerType) +
                    "&userid=" + strUserID +
                    "&password=" + strPassword +
                    "&action=Login";

                System.out.println("No connection, calling IDMConnection with url:
" + url);


getServletContext().getRequestDispatcher(url).forward(request,response);
          }

          // Remove URLout if still in request object so eClient will perform
          // default processing
          request.removeAttribute("URLout");

          System.out.println("Connected as user " + connection.getUserid());

          //
          // Get the search criteria related parameters
          //
          strTemp = request.getParameter("searchType");
          if (strTemp != null && strTemp.length() > 0)
          {
             strSearchType = strTemp;
          }

          strTemp = request.getParameter("searchEntity");
          if (strTemp != null && strTemp.length() > 0)
          {
             strSearchEntity = strTemp;
          }

          strTemp = request.getParameter("searchAttr");
          if (strTemp != null && strTemp.length() > 0)
          {
             strSearchAttr1 = strTemp;
          }

          strTemp = request.getParameter("searchValue");
          if (strTemp != null && strTemp.length() > 0)
          {
             strSearchValue1 = strTemp;
          }

          //
          // Build the search parameters based on the searchType param
          //
```

```
         System.out.println("strSearchType = " + strSearchType);
         if (strSearchType.equalsIgnoreCase("autoclaim"))
         {
            //
            // This special search type has the item type and entity
            // hardcoded in this servlet.  This means that only
            // searchType and searchValue need to be passed in on the
            // URL.
            //
            strSearchAttr1  = "Claimant Last Name (Content Manager V8.1 Sample 
Attribute)";
            strSearchEntity = "Auto Claim Form (Content Manager V8.1 Sample 
Item Type)";

            if (strSearchValue1.length() < 1)
            {
               System.out.println("searchValue param required for searchType 
of 'autoclaim'");
            }
         }
         else if (strSearchType.equalsIgnoreCase("generic"))
         {
            if (strSearchEntity.length() < 1)
            {
               System.out.println("searchEntity param required for searchType 
of 'generic'");
            }
            else if (strSearchAttr1.length() < 1)
            {
               System.out.println("searchAttr param required for searchType of 
'generic'");
            }
            else if (strSearchValue1.length() < 1)
            {
               System.out.println("searchValue param required for searchType 
of 'generic'");
            }
         }
         //
         // Anything that is passed to IDMBasicSearch must be
         // converted to a valid string for the servlet.  This
         // consists of:
         //   a) Converting spaces to %2B
         //
         strSearchEntity = URLEncoder.encode(strSearchEntity);
         strSearchAttr1 = convertString(strSearchAttr1);
         strSearchValue1 = convertString(strSearchValue1);

         //
```

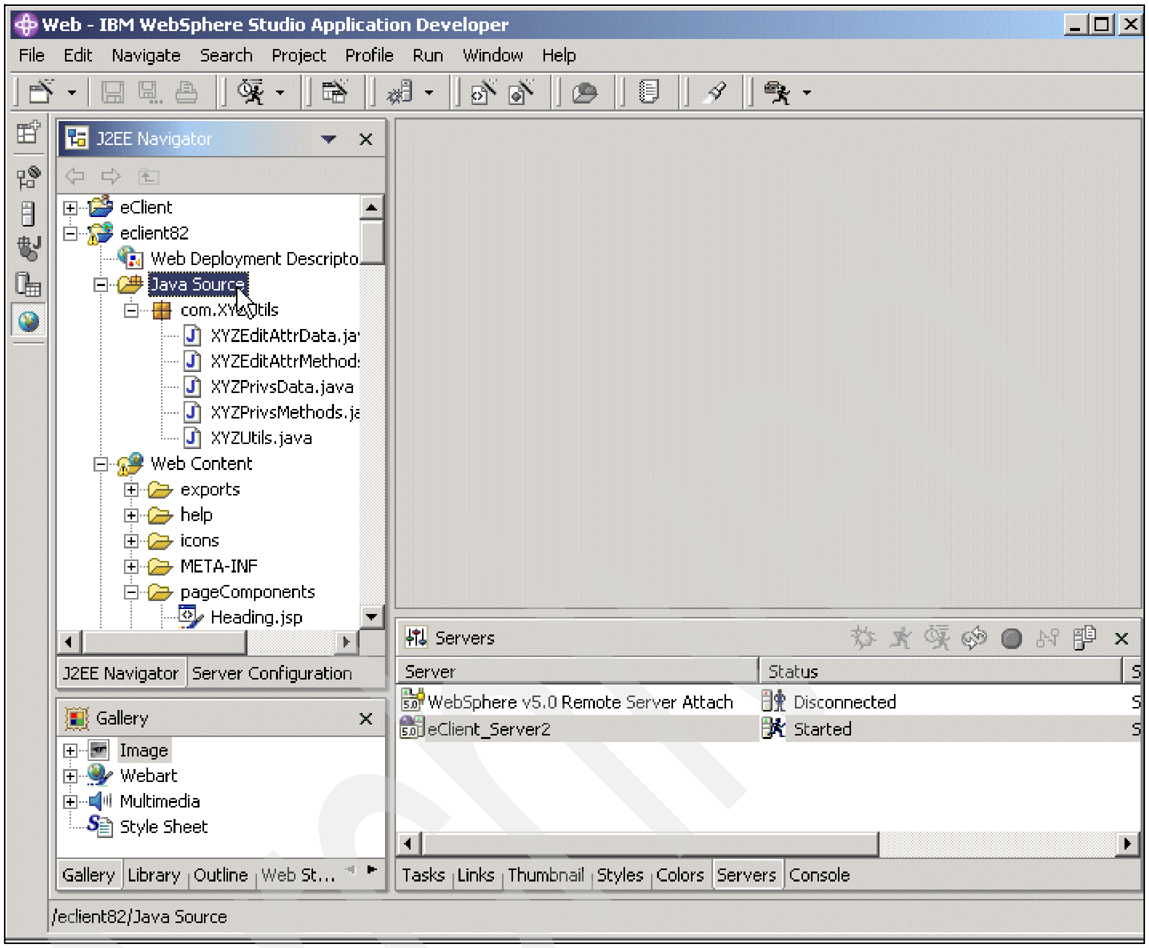
```
            // Perform the search
            //
            com.ibm.idm.beans.IDMUtilityBean cub =
            (com.ibm.idm.beans.IDMUtilityBean)session.getAttribute("cub");
            cub.setEntityNameValue(request);
            cub.setSearchType("IDMBasicSearch.jsp");  // ideas from
IDMBasicSearch.jsp

            url = "/IDMSearch?" +
                 strSearchAttr1 + "=" + strSearchValue1 +
                 "&radioAnyAll=ALL" +
                 "&attrCount=0" +
                 "&Entity=" + strSearchEntity +
                 "&NewWindow=true";

            System.out.println("Calling IDMBasicSearch with url of '" + url +
"'");

            getServletContext().getRequestDispatcher(url).forward(request,
response);
          }
          catch(Exception e) {
               throw new ServletException(e);
          }
     }
}
```

14.3 Configuring and using the servlet

In this section, we describe how to add the IDMIntegrator servlet to eClient using WebSphere Studio Application Developer. If you are not using WebSphere Studio Application Developer, you can use the WebSphere Application Server Application Assembly Tool to add the servlet to eClient.

Follow these steps to add the LOBIntegrator servlet to your application:

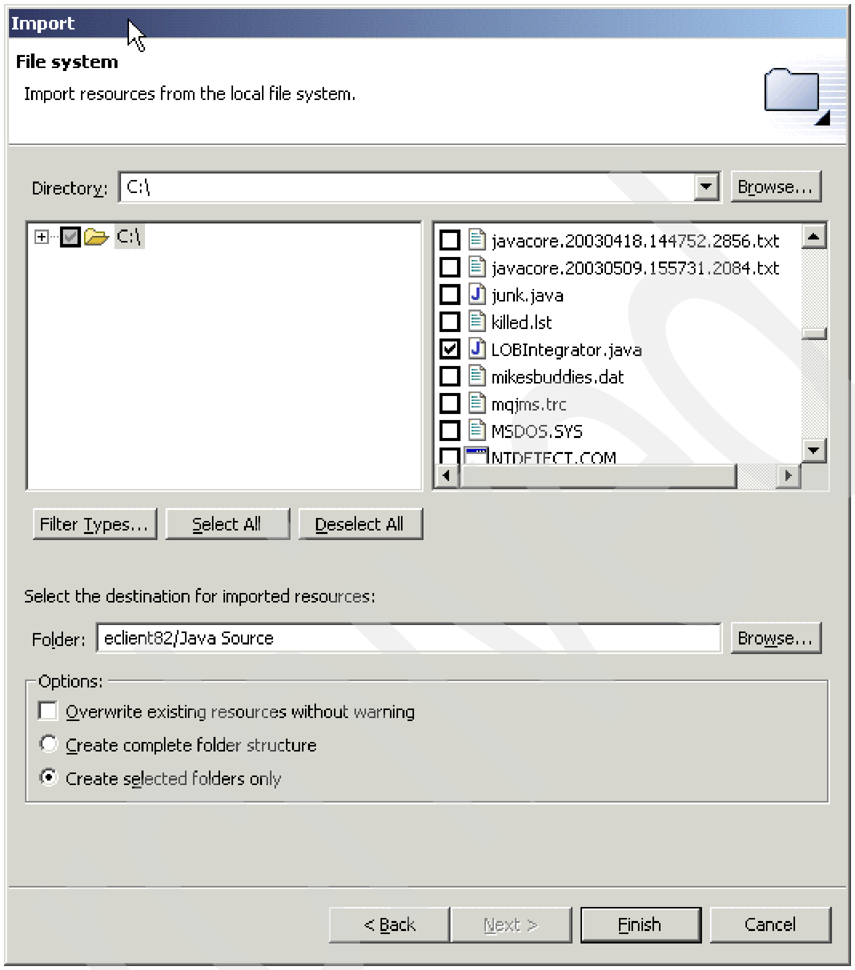
1. Open eClient in WebSphere Studio Application Developer, open the J2EE Navigator view, and highlight **Java Source** in the Navigator pane as shown in Figure 14-1 on page 333.



Server	Status
WebSphere v5.0 Remote Server Attach	Disconnected
eClient_Server2	Started

Figure 14-1 WebSphere Studio Application Developer J2EE Navigator view

2. Select **File -> Import**, select **File System** as the source and click **Next**.
3. Click **Browse** and select the directory that contains the sample code LOBIntegrator.java provided with the redbook. In the pane on the right, select **LOBIntegrator.java** as shown in Figure 14-2 on page 334.


Import

File system

Import resources from the local file system.

Directory: C:\

C:\

- [ ] javacore.20030418.144752.2856.txt
- [ ] javacore.20030509.155731.2084.txt
- [ ] junk.java
- [ ] killed.lst
- [x] LOBIntegrator.java
- [ ] mikesbuddies.dat
- [ ] mqjms.trc
- [ ] MSDOS.SYS
- [ ] NTDETECT.COM

Select the destination for imported resources:

Folder: eclient82/Java Source

Options:

- [ ] Overwrite existing resources without warning
- ( ) Create complete folder structure
- (●) Create selected folders only

Figure 14-2 WebSphere Studio Application Developer import file dialog

4. Click **Finish**. This imports the file into WebSphere Studio Application Developer. WebSphere Studio Application Developer analyzes the source code and recognizes that a servlet is being imported.
5. Go back to the J2EE Navigator window. It should now look similar to Figure 14-3 on page 335, with LOBIntegrator.java listed. Double-click **Web Deployment Descriptor** and select the **Servlets** tab in the pane on the right, as shown in Figure 14-3 on page 335.

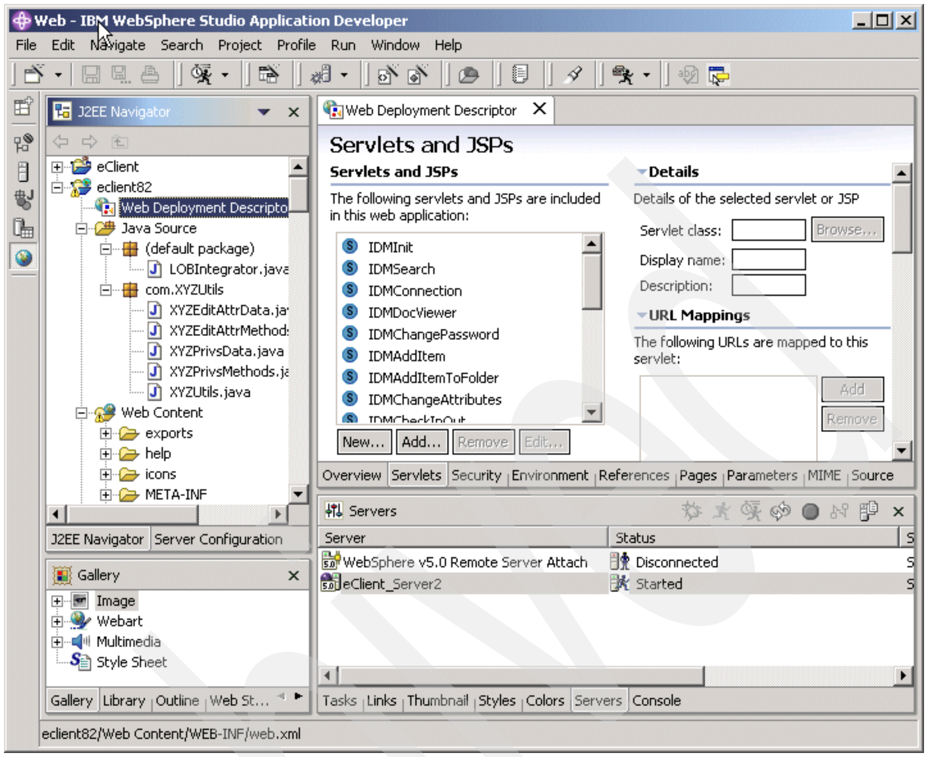

Web - IBM WebSphere Studio Application Developer
File Edit Navigate Search Project Profile Run Window Help
J2EE Navigator
eClient
eclient82
Web Deployment Descripto
Java Source
(default package)
LOBIntegrator.java
com.XYZUtils
XYZEditAttrData.ja
XYZEditAttrMethod
XYZPrivsData.java
XYZPrivsMethods.ja
XYZUtils.java
Web Content
exports
help
icons
META-INF
J2EE Navigator | Server Configuration
Gallery
Image
Webart
Multimedia
Style Sheet
Gallery | Library | Outline | Web St...
Web Deployment Descriptor
Servlets and JSPs
Servlets and JSPs
The following servlets and JSPs are included in this web application:
IDMInit
IDMSearch
IDMConnection
IDMDocViewer
IDMChangePassword
IDMAddItem
IDMAddItemToFolder
IDMChangeAttributes
IDMCheckInOut
New... Add... Remove Edit...
Details
Details of the selected servlet or JSP
Servlet class: Browse...
Display name:
Description:
URL Mappings
The following URLs are mapped to this servlet:
Add
Remove
Overview | Servlets | Security | Environment | References | Pages | Parameters | MIME | Source
Servers
Server | Status
WebSphere v5.0 Remote Server Attach | Disconnected
eClient_Server2 | Started
Tasks | Links | Thumbnail | Styles | Colors | Servers | Console
eclient82/Web Content/WEB-INF/web.xml

Figure 14-3 WebSphere Studio Application Developer Web Deployment Descriptor

6. Click **Add** in the Servlets and JSPs part of the window. The Add Servlet or JSP window, shown in Figure 14-4 on page 336, appears.

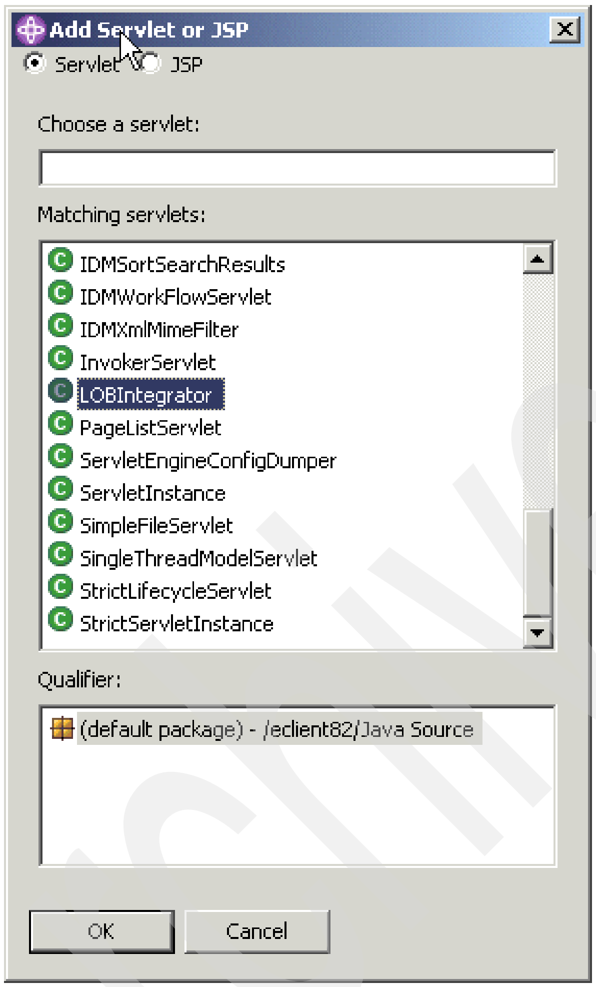
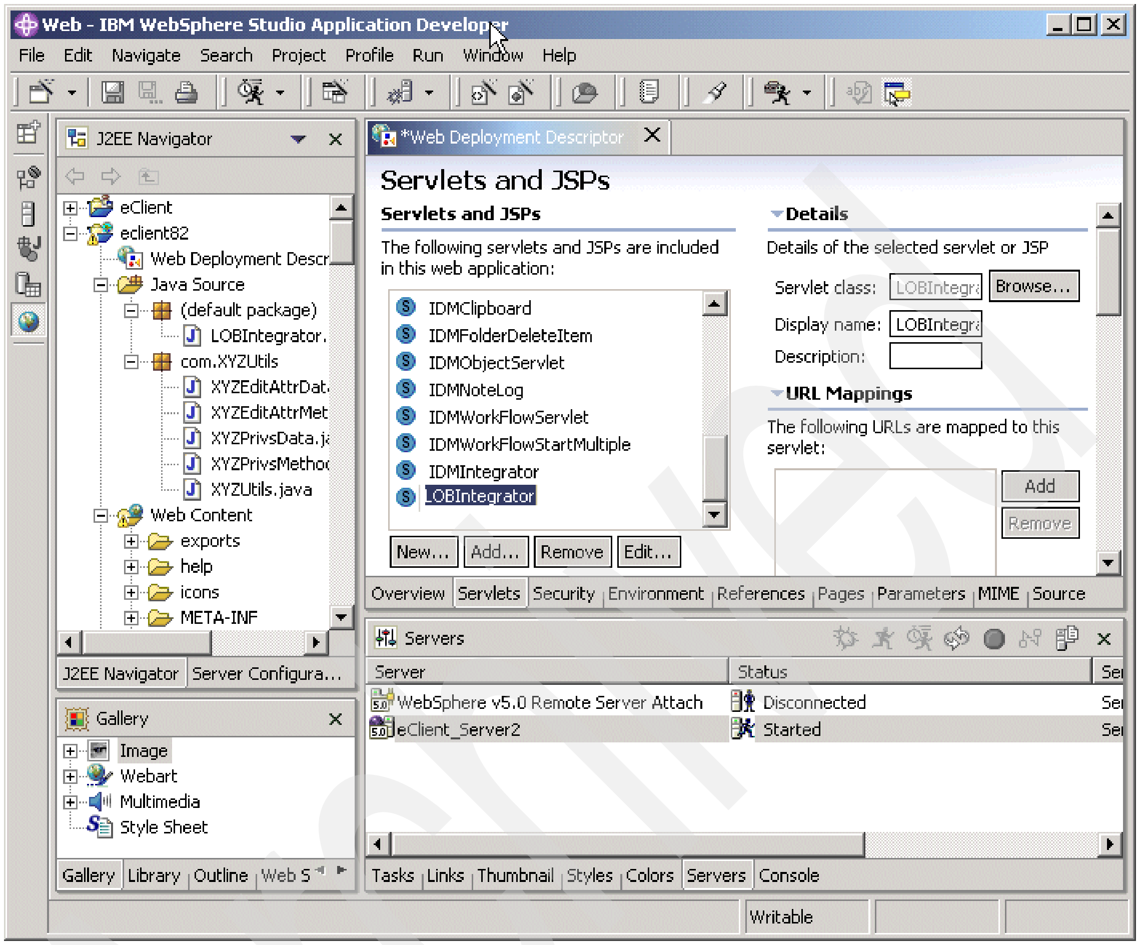


Figure 14-4 WebSphere Studio Application Developer - Add servlet window

7. Scroll down the servlet list until you find LOBIntegrator, as shown in Figure 14-4, and click **OK**.



Servlets and JSPs

Servlets and JSPs

The following servlets and JSPs are included in this web application:

Details

Details of the selected servlet or JSP

Servlet class: LOBIntegr

Display name: LOBIntegr

Description:

URL Mappings

The following URLs are mapped to this servlet:

Server	Status	Se
WebSphere v5.0 Remote Server Attach	Disconnected	Se
eClient_Server2	Started	Se

Figure 14-5 WebSphere Studio Application Developer - Add URL mapping for LOBIntegrator

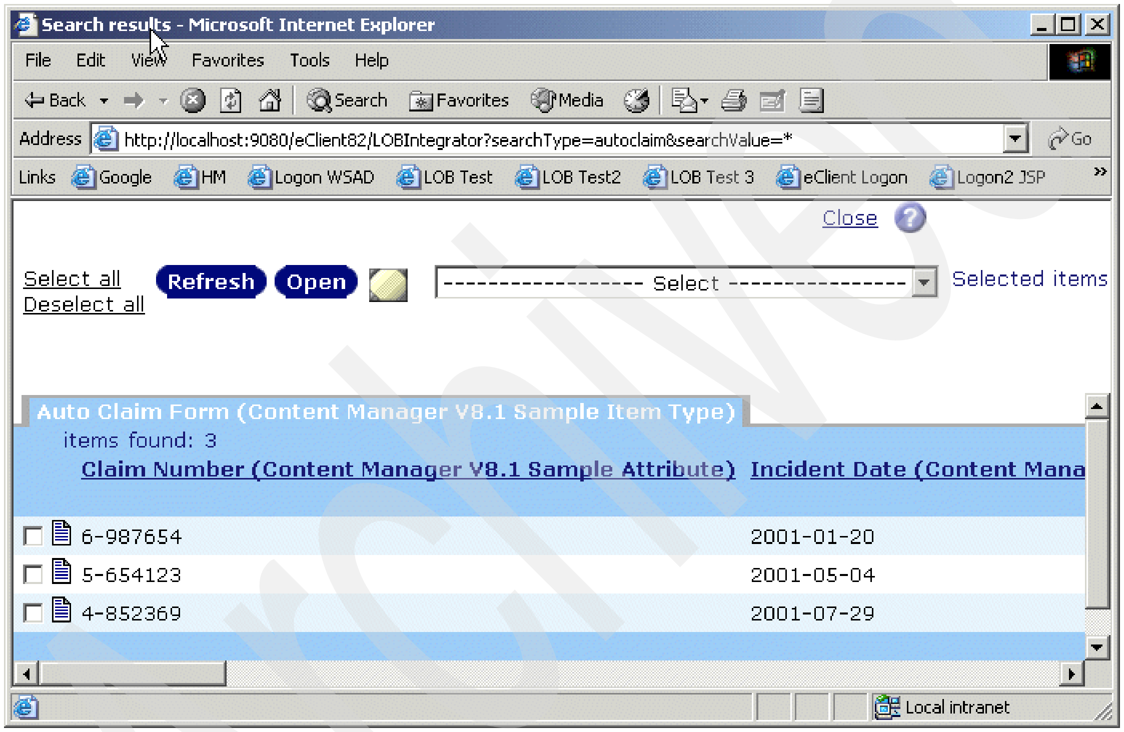
8. Click **Next**. Click **Add** in the URL Mappings section of the window (in the right pane in Figure 14-5). This adds \LOBIntegrator to the URL mappings window.
9. Close Web Deployment Descriptor window in the upper-right pane of the window by clicking the **X** in the tab along the top of the window. This saves the descriptor back to WebSphere Studio Application Developer.

At this point, you have added the LOBIntegrator.java source file, identified it as a servlet, and specified a URL mapping so that it can be accesseds from a Web browser.

Now you can test the installation. Start your test server. In Figure 14-5 on page 337, the servers are shown in the bottom-right pane. If your test server is not running, right-click it and click **Start**.

Open Internet Explorer and try one of the sample URLs described in the next section of this chapter. We entered the following URL and Figure 14-6 is displayed:

```
http://localhost:9080/eClient82/LOBIntegrator?searchType=autoclaim&searchVa
lue=*
```


Search results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Go
Links Google HM Logon WSAD LOB Test LOB Test2 LOB Test 3 eClient Logon Logon2 JSP
Close
Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items
Local intranet

http://localhost:9080/eClient82/LOBIntegrator?searchType=autoclaim&searchValue=*

Auto Claim Form (Content Manager V8.1 Sample Item Type)

items found: 3

Claim Number (Content Manager V8.1 Sample Attribute)	Incident Date (Content Mana
6-987654	2001-01-20
5-654123	2001-05-04
4-852369	2001-07-29

Figure 14-6 LOBIntegrator test

14.4 Servlet URL syntax and source

This section describes the URL syntax for the LOBIntegrator servlet.

14.4.1 Servlet parameters

Table 14-1 on page 339 lists and describes each parameter. The first four parameters specify connection information, and the other four parameters specify the search criteria. For our example, the userid, password, serverName, and serverType are hardcoded in the servlet.

The searchType parameter is very important. It describes the high-level type of search being requested. If the searchType is generic, then you must specify the SearchEntity, SearchAttr, and searchValue in the URL. If you do not want to specify the details of the search in the URL, you can create a custom searchType such as the autoclaim searchType in the example servlet. With a custom searchType, you just pass in the searchValue (such as last name or claim number) and the SearchEntity and SearchAttribute can be hardcoded in the servlet.

Table 14-1 LOBIntegrator servlet parameters

Parameter	Default	Description
userid	icmadmin	The user ID to log on with.
password	password	The password to log on with.
serverName	ICMNLSDB	The name of the server to log on to.
serverType	ICM	The server type. This can be ICM for CM8, FED for a federated connection, OD for OnDemand. It can be any of the back-end listed in the documentation for the setDSType method of the CMBConnection bean.
searchType	none	The searchType specifies the high-level type of the search. The servlet supports two types, autoclaim and generic. If autoclaim is used, then the Auto Claim Form entity is searched for a Claimant Last Name equals to the searchValue parameter. The SearchEntity and SearchAttr are hardcoded in the servlet for this searchType. If generic is used, then nothing is hardcoded and the SearchEntity, SearchAttr, and searchValue parameters are used to build the search string.
SearchEntity	none	The name of the entity to search. Required only for searchType = generic.
SearchAttr	none	The name of the attribute to search. Required only for searchType = generic.
searchValue	none	The value to search for in an attribute. If searchType is autoclaim, then this specifies the Claimant Last Name. Otherwise, it specifies the value for the SearchAttr parameter.

14.4.2 Sample URLs

We include some sample URLs that can be used with the LOBIntegrator servlet if you have documents in the NOINDEX index class or you have the Content Manager Version 8 sample data loaded.

The first URL searches against the Content Manager Version 8 sample data and the other two search against the NOINDEX index class.

The first example uses a searchType of autoclaim. The source code of the servlet handles this searchType by getting the searchValue and searching against a hardcoded item type and attribute.

The second and third examples use a searchType of generic. The generic searchType allows you to specify the entity you are searching, as well as the SearchAttribute and the searchValue.

You can modify the servlet to specify your own searchTypes as required. Using specific searchTypes can make your application integration more generic and put the logic to select the appropriate item type and attribute(s) to search on in the servlet instead of in the URL generated by the calling application.

Once you have the LOBServlet installed and working, and have the test data loaded in Content Manager Version 8, you can paste one of these URLs into Internet Explorer to test the servlet.

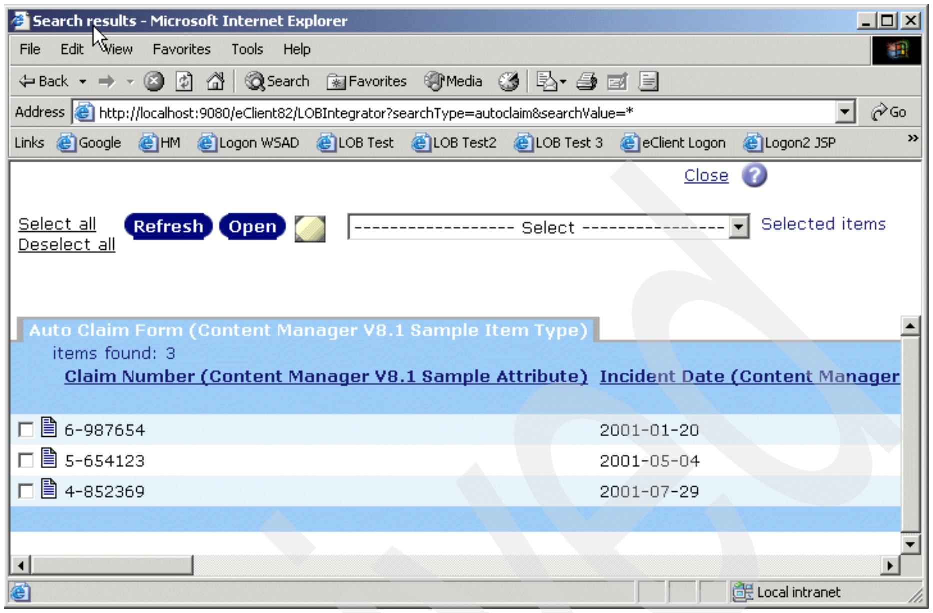
URL Example 1: Searching using the autoclaim searchType

This example uses a searchType of autoclaim. It logs on to Content Manager Version 8 using the hardcoded values in the servlet. It then searches against the item type and attribute specified in the servlet for the autoclaim searchType.

The example 1 URL is as follows:

```
http://localhost:9080/eClient82/LOBIntegrator?searchType=autoclaim&searchVa
lue=*
```

The output of the URL is shown in Figure 14-7 on page 341.


Search results - Microsoft Internet Explorer

Address: http://localhost:9080/eClient82/LOBIntegrator?searchType=autoclaim&searchValue=*

Auto Claim Form (Content Manager V8.1 Sample Item Type)

items found: 3

Claim Number (Content Manager V8.1 Sample Attribute)	Incident Date (Content Manager
6-987654	2001-01-20
5-654123	2001-05-04
4-852369	2001-07-29

Figure 14-7 Sample URL #1 displayed in Internet Explorer

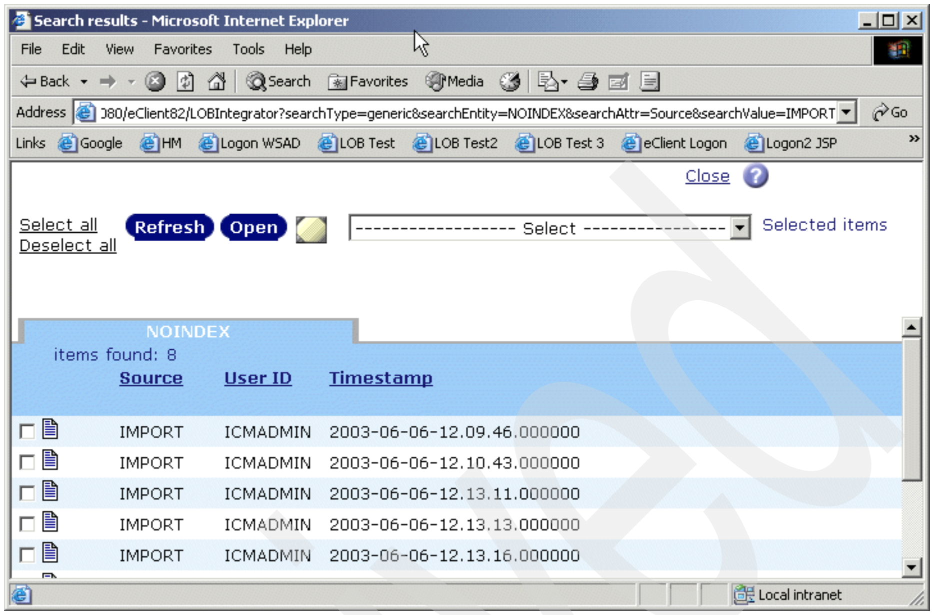
Example 2: Search using the generic searchType

This example uses a searchType of generic. It searches the NOINDEX item type, and search the Source attribute for a value of IMPORT.

Example 2 URL is as follows:

```
http://localhost:9080/eClient82/LOBIntegrator?searchType=generic&searchEnti
ty=NOINDEX&searchAttr=Source&searchValue=IMPORT
```

The output of the URL is shown in Figure 14-8 on page 342.


Search results - Microsoft Internet Explorer

Address:)80/eClient82/LOBIntegrator?searchType=generic&searchEntity=NOINDEX&searchAttr=Source&searchValue=IMPORT

NOINDEX

items found: 8

		Source	User ID	Timestamp
☐	🗎	IMPORT	ICMADMIN	2003-06-06-12.09.46.000000
☐	🗎	IMPORT	ICMADMIN	2003-06-06-12.10.43.000000
☐	🗎	IMPORT	ICMADMIN	2003-06-06-12.13.11.000000
☐	🗎	IMPORT	ICMADMIN	2003-06-06-12.13.13.000000
☐	🗎	IMPORT	ICMADMIN	2003-06-06-12.13.16.000000

Figure 14-8   Sample URL #2 displayed in Internet Explorer

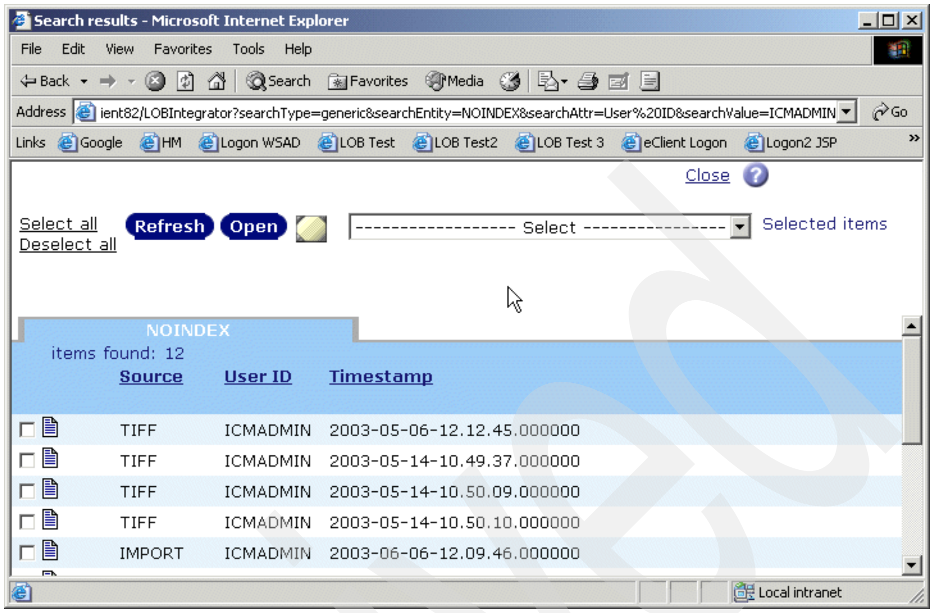
Example 3: Searching using the generic searchType

This example uses a searchType of generic. It searches the NOINDEX item type where the User ID attribute is ICMADMIN. You may need to manually create documents with matching attributes.

Example 3 URL is as follows:

```
http://localhost:9080/eClient82/LOBIntegrator?searchType=generic&searchEnti
ty=NOINDEX&searchAttr=User ID&searchValue=ICMADMIN
```

The output of the URL is shown in Figure 14-9 on page 343.


Search results - Microsoft Internet Explorer

Address: ient82/LOBIntegrator?searchType=generic&searchEntity=NOINDEX&searchAttr=User%20ID&searchValue=ICMADMIN

NOINDEX

items found: 12

	Source	User ID	Timestamp
☐	TIFF	ICMADMIN	2003-05-06-12.12.45.000000
☐	TIFF	ICMADMIN	2003-05-14-10.49.37.000000
☐	TIFF	ICMADMIN	2003-05-14-10.50.09.000000
☐	TIFF	ICMADMIN	2003-05-14-10.50.10.000000
☐	IMPORT	ICMADMIN	2003-06-06-12.09.46.000000

File Edit View Favorites Tools Help
Back Search Favorites Media
Go
Links Google HM Logon WSAD LOB Test LOB Test2 LOB Test 3 eClient Logon Logon2 JSP
Close
Select all
Deselect all
Refresh
Open
------------------ Select ----------------
Selected items
Local intranet

Figure 14-9 Sample URL #3 displayed in Internet Explorer

14.5 Invoking the servlet from an application

There are many ways to invoke the servlet described in this chapter. Any application that is capable of launching a Web browser and passing a URL to it can use this servlet integration.

Before proceeding, first make sure the servlet is working by invoking the URLs from a Web browser. Enter (or cut and paste) in one of the sample URLs or your own URL to invoke the servlet and display search results in eClient. Once the servlet works as you expected, you can invoke it through other applications.

A simple way to invoke the servlet from another application is to “shell” out to the operating system, run **`iExplore.exe`**, and then pass the URL on the command line. Visual Basic provides a **`shell()`** function to do this. It can be called as shown in the example below.

Integrating with Visual Basic application using shell() function

From Visual Basic, you can use the **`shell()`** command to launch a URL, as illustrated in Example 14-2 on page 344.

Example 14-2 Visual Basic sample code - Launch eClient using shell() command

```
Call Shell("c:\Program Files\Internet Explorer\iExplore.exe
""http://localhost:9080/eClient82/LOBIntegrator?searchType=generic&searchEntity
=NOINDEX&searchAttr=Source&searchValue=IMPORT""", vbNormalFocus)
```

There is a sample Visual Basic application described later that implements this code. Other application development languages usually have something similar to the Visual Basic **`shell()`** command to let you do the same thing.

Using ActiveX Automation (improving performance)

Performance may suffer if you open a new instance of a Web browser for each search, because the servlet has to log on to eClient for each request. You can improve performance by re-using the Web browser window, that is by using the same browser which re-uses the same eClient logon to send URLs for different search requests. One way to do this is with ActiveX Automation of Internet Explorer.

ActiveX Automation allows an application to create an instance of another application and interact with it programatically. Many applications provide an ActiveX Automation interface including standard desktop applications such as Microsoft Word, Microsoft Excel, and Microsoft Internet Explorer. Other applications, including the IBM Content Manager thick client application, provide ActiveX Automation programming interfaces.

As an example, a programmer can use ActiveX Automation from Visual Basic with Microsoft Word to open a document, update the contents, and print to a printer without requiring any user interaction. The ActiveX Automation interface for the Content Manager thick client can be used to import documents, export documents, re-index documents, or view documents. These tasks can be done from any application development environment that supports ActiveX Automation, including Microsoft Visual Basic, Microsoft Visual C++, PowerBuilder, and others.

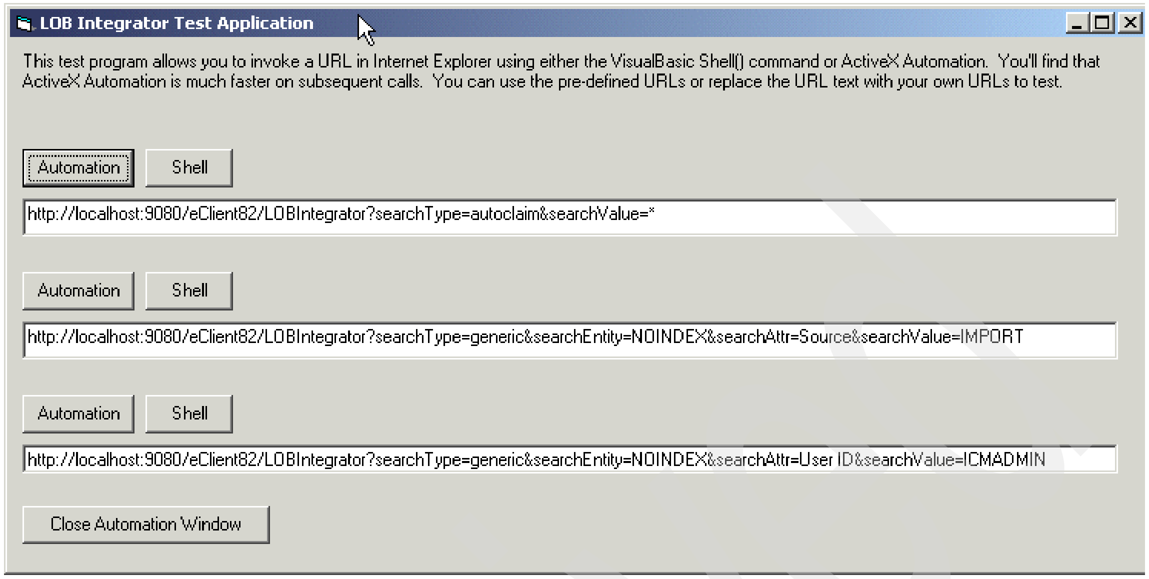
The code shown in Example 14-3 on page 345 is for a button-clicking event in Visual Basic. It uses a global variable called `goie` that contains the handle to an ActiveX Automation instance of Internet Explorer. The CreateObject() function creates an instance of Internet Explorer. The ActiveX object can then be used to interact with Internet Explorer as shown via the Visible() and Navigate() methods.

Using ActiveX Automation has a major advantage over using the **`shell()`** command, because the same instance of the eClient can be re-used over and over for different searches. This makes the display of the search results much faster than using the **`shell()`** command. In a stand-alone environment on a ThinkPad T30 as in our scenario, the response time for a simple search is about one second using ActiveX Automation and about five seconds using the **shell()** command to launch a new window each time.

Example 14-3 Visual Basic sample code - Re-use browser object

```
Dim goie As Object ‘ Global variable
Private Sub cmdExecute2_Click()

    On Error GoTo ErrorHandler

TryAgain:
    If (goie Is Nothing) Then
        Set goie = CreateObject("internetexplorer.application")
    End If
    goie.Visible = True
    goie.Navigate (frmMain.cmd2)
    Exit Sub

ErrorHandler:
    If (Err.Number = -2147417848) Then
        ' The user closed the internet explorer window
        Set goie = Nothing
        Resume TryAgain
    End If

   End Sub
```

We include a Visual Basic application that demonstrates the usage of the **shell()** command and ActiveX Automation. Figure 14-10 on page 346 shows a window from the sample application.



This test program allows you to invoke a URL in Internet Explorer using either the VisualBasic Shell() command or ActiveX Automation. You'll find that ActiveX Automation is much faster on subsequent calls. You can use the pre-defined URLs or replace the URL text with your own URLs to test.

```
http://localhost:9080/eClient82/LOBIntegrator?searchType=autoclaim&searchValue=*
http://localhost:9080/eClient82/LOBIntegrator?searchType=generic&searchEntity=NOINDEX&searchAttr=Source&searchValue=IMPORT
http://localhost:9080/eClient82/LOBIntegrator?searchType=generic&searchEntity=NOINDEX&searchAttr=User ID&searchValue=ICMADMIN
```

Figure 14-10 Visual Basic application for testing the LOBIntegrator servlet

This sample application specifies the three sample URLs discussed earlier in this chapter in the text boxes. For each URL, you can invoke the LOBIntegrator servlet using either ActiveX Automation or using the **`shell()`** command by clicking one of the buttons. If you use the Automation buttons, the first time you use a button, a new Internet Explorer window is created. Subsequent calls re-use the same window.

The sample code demonstrates invoking eClient from an external application. If you have a Visual Basic application that you want to integrate with eClient, you can add code similar to that for the Automate button to your application to get search results in eClient. You first need to know what the user is looking at in the application (in this case, claim number 12345), build a URL with a search criteria, and pass the URL to Internet Explorer using ActiveX Automation. You can do the similar modification with different syntax for other development languages that support ActiveX Automation (such as C++ and PowerBuilder).

To download the source code of the sample application, see Appendix B, “Additional material” on page 465 for download instructions.

15

Siebel Integration

In this chapter, we discuss how to integrate IBM DB2 Content Manager Version 8 eClient with a Siebel CRM application. The integration allows Siebel end users to retrieve and view documents stored in IBM DB2 Content Manager Version 8 system.

This chapter contains the following topics:

- Installing Siebel Integration for Content Manager
- Configuring eClient
- Configuring Siebel
- Setting up Content Manager and EIP
- Verification

15.1 Introduction

Siebel Integration for IBM Content Manager combines the strength of both the Content Manager product portfolio and a Siebel business application. The goal of the integration is to use the Siebel application to manage the business process, and use the Content Manager product portfolio to manage unstructured data for the Siebel server.

With Siebel Integration for IBM Content Manager, the unstructured data from Siebel may be stored in one of the following back-end servers:

- Content Manager Version 7.1
- Content Manager Version 8.1
- Content Manager Version 8.2
- Content Manager OnDemand for Multiplatforms Version 7.1
- Content Manager OnDemand for OS/390 Version 2.1, Version 7.1
- Content Manager OnDemand for iSeries™ Version 4.5, Version 5.1
- Content Manager ImagePlus for OS/390 Version 3.1

With Siebel Integration for IBM Content Manager, a Siebel end user is able to:

- Search and retrieve documents that are associated with a Siebel entity, such as a service request.
- Use an eClient viewer to view individual documents, such as attachment to a service request.
- While viewing a document, toggle on and off document annotations that were previously created, zoom in and out, rotate the current page, and print the document if the system is configured to enable the print function.

In this chapter, you learn how to use Siebel Integration for IBM Content Manager in the following environment:

- Content Manager V8.2 with WebSphere Application Server 5.0
- EIP V8.2
- eClient V8.2 with WebSphere Application Server 5.0
- Siebel V7.52 with WebSphere Application Server 4.0

15.2 Installing Siebel Integration for Content Manager

Siebel Integration for IBM Content Manager components are automatically installed when you install the IBM DB2 Content Manager Version 8 eClient. They include several servlets, JavaServer Pages (JSP), icons, one Cascading Style Sheet, two Siebel Web templates, and a sample Integration Properties (IP) file.

15.2.1 Installing Siebel Web templates

There are two Siebel Web templates (EIP81Applet.swt and EIP81Body.swt) in the C:\CMeClient\integration\siebel directory (/CMeClient/integration/siebel directory for AIX and Sun Solaris) after installing eClient, where /CMeClient is the directory in which eClient is installed.

To complete the installation of Siebel Integration, you must copy both Web templates into three directories on the Siebel machine:

- SIEBELROOT\siebsrvr\WEBTEMPL
- SIEBELROOT\client\WEBTEMPL
- SIEBELROOT\tools\WEBTEMPL

where SIEBELROOT is the directory in which Siebel is installed.

15.3 Configuring eClient

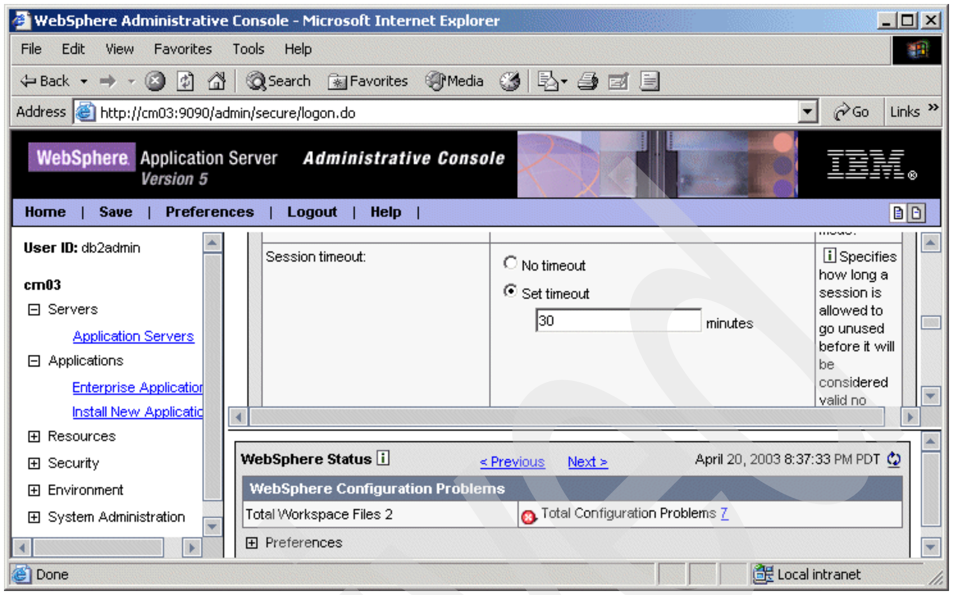
After you install Siebel Integration for IBM Content Manager, you must configure both the eClient and the Siebel CRM applications to make the two products work together.

15.3.1 Configuring eClient application server in WebSphere

Since eClient is a Web application server deployed in WebSphere Application Server, you should ensure that the time-out session value is appropriately set within WebSphere for the Siebel user community.

Before continuing the following steps, make sure that your WebSphere Application Server V5.0 server1 is running.

1. On eClient server, launch WebSphere Application Server Administrative console by selecting **Start -> Programs -> IBM WebSphere -> Application Server V5.0 -> Administrative Console**.
2. In the left Navigator pane, select **Servers -> Application Servers -> eClient_Server -> Web Container -> Session Management**.
3. Find the Session timeout field as shown in Figure 15-1 on page 350.


WebSphere Administrative Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://cm03:9090/admin/secure/logon.do
WebSphere Application Server Version 5 Administrative Console
Home | Save | Preferences | Logout | Help |
User ID: db2admin
cm03
Servers
Application Servers
Applications
Enterprise Applicatior
Install New Applicatic
Resources
Security
Environment
System Administration

Session timeout: ○ No timeout ◉ Set timeout 30 minutes

Specifies how long a session is allowed to go unused before it will be considered valid no

WebSphere Status < Previous Next > April 20, 2003 8:37:33 PM PDT

WebSphere Configuration Problems

Total Workspace Files 2	Total Configuration Problems 7

Preferences
Done
Local intranet

Figure 15-1 Set Session timeout in WebSphere Application Server

4. Select **Set Timeout**, and set the value in minutes. This value must be at least two minutes.
5. Click **OK** to apply the modification.
6. Click **Save** twice to save the change.

15.3.2 Configuring integration properties file

The Integration Properties file (hereafter referred to as the IP file) resides in the c:\CMeClient directory on eClient server. It specifies property values that configure your environment for integration with Siebel.

You may give the integration file any name you wish. The IP file name will be included in the calculated field when you configure Siebel. If you want to change the IP file name after configuring Siebel, make sure that you have the same name in the Siebel configuration. A sample IP file Siebel.properties is provided in the c:\CMeClient directory.

By specifying values for the required fields in the integration property file, you ensure that only the URL that originated from an authorized Siebel server has access to unstructured data in the Content Manager servers. It also gives a look and feel to Web pages generated by the eClient similar to the ones generated by Siebel.

1. Open the integration property file C:\CMeClient\Siebel.properties in a text editor.
2. Four properties in the IP file are required:
   - **eClientToken**

     The eClientToken property is used to control access to the eClient server. Siebel application uses an assembled URL to access eClient. The eClientToken property is part of the assembled URL. When the URL is sent to eClient, the eClient server then compares the token provided in the URL with the token in the IP file. eClient only allows access to unstructured data if these two tokens match.

     The eClientToken property is case-sensitive. Valid characters are any of the ISO8859-1 Latin 1 characters with the exception of the following characters, which are reserved for use within the query string of a URL:

     ```
     ;
     /
     ?
     :
     @
     &
     =
     +
     ,
     $
     ```

   In our scenario, we set the parameter to `token`.
   - **type**

     The type property specifies the look and feel of Web pages that are produced by the eClient JavaServer Pages. For Siebel Integration for IBM Content Manager, set the type to `1`.
   - **cssPrefix**

     The cssPrefix property specifies a file name prefix for the Cascading Style Sheet file used by the eClient JavaServer Pages for integration with Siebel. Set the cssPrefix property to `alt1`.
   - **iconPrefix**

     The iconPrefix property specifies a file name prefix for the icon files used by the eClient JavaServer Pages for integration with Siebel. Set the iconPrefix property to `alt1`.
3. For the server, userid, and password properties, you can either specify them in the IP file or specify them as arguments in the URL generated in Siebel. If

you specify them in both places, the values defined in the Siebel take precedence over the values in the IP file.

   – **server**

     The server property specifies the name of the federated server database. If your Content Manager database is ICMNLSDB and the EIP database is EIPDB, you must use EIPDB as the server property.

     > **Restriction:** At the time of this writing, Siebel application is allowed to connect only to the EIP federated database server. Direct connection to a back-end server is not available.

   – **userid**

     The userid property is used to access the federated server database.

   – **password**

     The password property is used to access the federated server database.

4. Optionally, you may specify the value for the printEnabled property. The printEnabled property specifies whether a print capability is included in the toolbar of the document viewers. It can be set to true or false. The default value is true.

Example 15-1 shows a sample integration property file.

Example 15-1 Sample Siebel.properties file

```
eClientToken=token
type=1
cssPrefix=alt1
iconPrefix=alt1
server=EIPDB
userid=icmadmin
password=password
printEnabled=false
```

15.3.3 Configuring browser

If you use the eClient viewer applet to view documents, you must configure your client browser properly. For more information, read Chapter 2, “Installing eClient” on page 17.

15.4 Configuring Siebel

There are two ways to configure Siebel Integration for IBM Content Manager depending on the version of Siebel you are using:

- Specify a URL in a calculated field of a Siebel business component. To complete configuration, you must do three tasks:
  - Customize the business objects layer.
  - Customize the user interface layer.
  - Configure the Siebel application.
- Use the Siebel portal framework to define a symbolic URL within the calculated field of a Siebel business component. This option also consists of three tasks:
  - Configure a business component to handle external data using a symbolic URL.
  - Display external content within an applet.
  - Configure the Siebel application.|

Siebel Integration for IBM Content Manager supports Siebel V7.0.4 and Siebel V7.5.2. If you are using Siebel V7.0.4, the first option is the only one available. If you are using Siebel V7.5.2, you may choose either option to configure the integration, although the second one is highly recommended.

Tip: Archive the Siebel repository objects (SRF files) before you start making any changes. If you need to remove the Siebel Integration for IBM Content Manager later, you can import these archived object definitions to restore your Siebel application environment to the level that existed prior to this configuration.

In this chapter, we discuss how to configure Siebel integration by using the second option (use the Siebel portal framework to define a symbolic URL within the calculated field of a Siebel business component). For the procedure of the first option (specifying a URL in a calculated field of a Siebel business component), refer to Chapter 4 of *IBM Content Manager for Multiplatforms / IBM Information Integrator for Content: Installing, Configuring, and Managing eClient*, SC27-1350.

Siebel Version V7.5.2 provides portal agents that allow you to integrate external data (for example, unstructured data managed by Content Manager) into the Siebel user interface. You configure a calculated field in the business component to handle external data using a symbolic URL. You then configure an applet to display the external HTML content inside of the applet container within a view.

In our scenario, we set up the environment with values shown in Table 15-1 and Table 15-2 to demonstrate how to configure Siebel integration.

Table 15-1 Environment on Siebel server machine

Product/Environment	Value
Host name	SiebelHost
Siebel	7.5.2
Siebel user	John Smith

Table 15-2 Environment on eClient server machine

Product/Environment	Setting
Host name	eClientHost
eClient	V8.2
eClient application server	eClient82
eClientToken	token
Integration property file name	Siebel.properties
EIP search template used for Siebel integration	SiebelAttachment
Search criteria #1	searchBySRNumber
Search criteria #2	searchByLastName
Content Manager	V8.2
EIP	V8.2
WebSphere Application Server	V5.0

15.4.1 Configuring business component

To configure Siebel business components to handle external data using a symbolic URL, you need to create a new calculated field in the business component, as follows:

1. Launch Siebel Tools by selecting **Start -> Programs -> Siebel Tools 7.5.2 -> Siebel Tools**.
2. In Object Explorer on the left, select **Siebel Objects -> Business Component**.

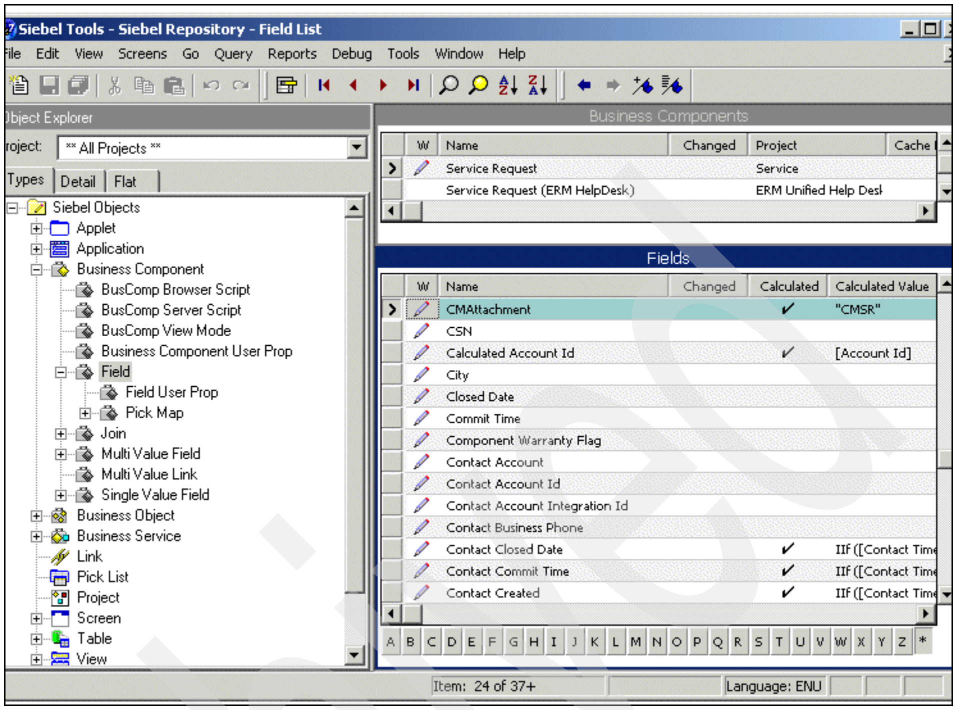
3. In the Business Components pane on the right, select the **Service Request** business component (or any business component in which you want to implement the integration).
4. From the menu bar, select **Tools -> Lock Project**.
5. In Object Explorer on the left, select **Business Component -> Field**. All of the fields for the Service Request business component are displayed in the Fields pane on the right.
6. In the Fields pane, right-click the window title bar and select **New Record**. A blank new record appears above all the existing records.

   Set the values for the fields found in Table 15-3 and keep the default values for the rest of the fields. See Figure 15-2 on page 356.

Table 15-3 New field definition

Field	**Value**
Name	CMAttachment
Calculated	True
Calculated Value	"CMSR"
Type	DTYPE_TEXT
Use Default Sensitivity	True

For the Name field, you may use any meaningful name. In the Calculated Value field, enter the name of the symbolic URL (enclosed in double quotes) that you want to use to submit the HTTP request for retrieving documents stored in Content Manager server. You define the symbolic URL in "Step 3. Defining a symbolic URL" on page 366.


Siebel Tools - Siebel Repository - Field List

File Edit View Screens Go Query Reports Debug Tools Window Help

Object Explorer

Project: ** All Projects **

Types | Detail | Flat

- Siebel Objects
  - Applet
  - Application
  - Business Component
    - BusComp Browser Script
    - BusComp Server Script
    - BusComp View Mode
    - Business Component User Prop
    - Field
      - Field User Prop
      - Pick Map
    - Join
    - Multi Value Field
    - Multi Value Link
    - Single Value Field
  - Business Object
  - Business Service
  - Link
  - Pick List
  - Project
  - Screen
  - Table
  - View

Business Components

W	Name	Changed	Project	Cache
	Service Request		Service	
	Service Request (ERM HelpDesk)		ERM Unified Help Desl	

Fields

W	Name	Changed	Calculated	Calculated Value
	CMAttachment		✓	"CMSR"
	CSN			
	Calculated Account Id		✓	[Account Id]
	City			
	Closed Date			
	Commit Time			
	Component Warranty Flag			
	Contact Account			
	Contact Account Id			
	Contact Account Integration Id			
	Contact Business Phone			
	Contact Closed Date		✓	IIf ([Contact Time
	Contact Commit Time		✓	IIf ([Contact Time
	Contact Created		✓	IIf ([Contact Time

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z *

Item: 24 of 37+ Language: ENU

Figure 15-2 Define new calculated field

7. Save the changes.

Important: You still have the Service Request business component locked at this time. You must keep it locked for the rest of the configuration.

15.4.2 Displaying external content within an applet

After you create the calculated field CMSR for the Service Request business component, you use a control in a form applet to expose it in the user interface.

Step 1. Creating a Web template object

Complete the following steps to create a Web template object:

1. Launch Siebel Tools by selecting **Start -> Programs -> Siebel Tools 7.5.2 -> Siebel Tools**.
2. In Object Explorer on the left, select **Siebel Objects -> Web Template**.
3. Right-click the Web template window title bar, and select **New Record**. A blank new record appears above all the existing records.

Set the values for the fields defined in Table 15-4 and keep the default values for the rest of fields.

Table 15-4 Define a new Web template

Field	Value
Name	CM Attachment Applet for Service Request
Project	Service
Type	Applet Template - Form

You may use any meaningful name for the new Web template. If you do not know the project name, click the drop-down box and select one from the list. Select the project that is associated to the business component being modified. For the demonstration in this chapter, we are modifying the business component Service Request and its project is Service. The project Service must be locked.

4. Save the changes.

Step 2. Creating a Web template file object

Complete the following steps to create a Web template file object:

1. In Siebel Tools, highlight the Web template object, **CM Attachment Applet for Service Request**, which you created in Step 1.
2. In Object Explorer on the left, select **Siebel Objects -> Web Template -> Web Template File**.
3. Right-click the title bar of the Web Template File window and select **New Record**. A blank new record appears above all the existing records.

   Set the values for the fields found in Table 15-5 and keep the default values for all other fields.

Table 15-5 Define a Web template file

Field	Value
Name	CM Attachment Applet for Service Request
File Name	EIP81Applet.swt

Important: You must use the same name for both the Web template file and the Web template.

4. Save the changes.

Step 3. Creating an applet

Complete the following steps to create an applet:

1. In Siebel Tools, select **Siebel Objects -> Applet** in Object Explorer on the left.
2. Right-click the Applet window title bar on the right and select **New Record**. A blank new record appears above all the existing records.

   Set the values for the fields found in Table 15-6 and keep the default values for all other fields.

Table 15-6 Define an applet

Field	Value
Name	CM Attachment Applet
Project	Service
Business Component	Service Request
Class	CSSFrameBase
Title	CM Attachment Applet
Type	Standard

   You may use any meaningful name for the new applet. If you do not know the project name, select one from the drop-down box. Select the project that is associated to the business component being modified.

3. Save the changes.

Step 4. Creating an applet control

Complete the following steps to create an applet control:

1. In Siebel Tools, highlight the applet **CM Attachment Applet** that you created in “Step 3. Creating an applet” on page 358.
2. In Object Explorer on the left, select **Siebel Objects -> Applet -> Control**.
3. Right-click the Control window title bar and select **New Record**. A blank new record appears above all the existing records.

   Set the values in Table 15-7 on page 359 and keep the default values for all other fields.

Table 15-7 Define an applet control

Field	Value
Name	`CMAttachment`
Display Format	`HTML Text`
Field	`CMAttachment`
Field Retrieval Type	`Symbolic URL`
HTML Display Mode	`DontEncodeData`
HTML Only	`True`
HTML Row Sensitive	`True`
HTML Type	`Field`
Read Only	`True`
Sort	`True`
Text Alignment	`Left`
Visible	`True`

Important: You must use the same name as the one for the calculated field defined in Table 15-3 on page 355. And you must also use the same value for the Field field.

4. Save the changes.

Step 5. Creating an applet Web template

Complete the following steps to create an applet Web template:

1. In Siebel Tools, highlight the applet CM **Attachment Applet** that you created in "Step 3. Creating an applet" on page 358.
2. In Object Explorer on the left, select **Siebel Objects -> Applet -> Applet Web Template**.
3. Right-click the Applet Web Templates window title bar and select **New Record**. A blank new record appears above all existing records.

   Set the values for the fields found in Table 15-8 on page 360 and keep the default values for all other fields.

Table 15-8 Define an applet Web template

Field	**Value**
Name	Base
Type	Base
Web template	CM Attachment Applet for Service Request

Important: When you enter data in Web template field, use the name defined in "Step 1. Creating a Web template object" on page 356.

4. Save the changes.

Step 6. Creating an applet Web template item

Complete the following steps to create an applet Web template item:

1. In Siebel Tools, highlight the applet Web template that you created in "Step 5. Creating an applet Web template" on page 359.
2. In Object Explorer on the left, select **Siebel Objects -> Applet -> Applet Web Template -> Applet Web Template Item**.
3. Right-click the Applet Web Template Item window title bar and select **New Record**. A blank new record appears above all existing records.

   Set the values for the fields found in Table 15-9 and keep the default values for all other fields.

Table 15-9 Define an applet Web template item

Field	**Value**
Name	CMAttachment
Control	CMAttachment
Item Identifier	1301
Type	Control

Important: For the Name field, use the same name as the calculated field defined in "Configuring business component" on page 354. For the Control field, use the one created in "Step 4. Creating an applet control" on page 358.

4. Save the changes.

Step 7. Creating a view

Complete the following steps to create a view:

1. In Siebel Tools, select **Siebel Objects -> View** in the Object Explorer on the left.
2. Right-click the View window title bar and select **New Record**. A blank new record appears above all existing records.

   Set the values for the fields found in Table 15-10 and keep the default values for all other fields.

Table 15-10 Define a view

Field	Value
Name	CMAttachment View for Service Request
Project	Service
Business Object	Service Request
Thread Applet	CM Attachment Applet
Thread Field	SR Number
Thread Title	SR Number:

   You may use any meaningful name. The project contains the name of the business component you are modifying. The newly created view will become a new tab for the business object Service Request.

   The SR Number is the Service Request Number field defined in Siebel. It is used as the search criteria while retrieving documents from the EIP search template.

3. Save the changes.

Step 8. Creating a view Web template

Complete the following steps to create a view Web template:

1. In Siebel Tools, highlight the view **CMAttachment View for Service Request** that you created in "Step 7. Creating a view" on page 361.
2. In the Object Explorer on the left, select **Siebel Objects -> View -> View Web Template**.
3. Right-click the title bar of the View Web Template window and select **New Record**. A blank new record appears above all existing records.

   Set the values for the fields found in Table 15-11 on page 362 and keep the default values for all other fields.

Table 15-11 Define a view Web template

Field	Value
Name	Base
Web Template	View Detail

4. Save the changes.

Step 9. Creating view Web template items.

In this section, we create two view Web template items since we want to display two applets in the view. For each applet which is displayed in the view, you must create a corresponding Web template item.

1. In Siebel Tools, highlight the view Web template that you created in “Step 8. Creating a view Web template” on page 361.
2. In the Object Explorer on the left, select **Siebel Objects -> View -> View Web Template -> View Web Template Item**.
3. Right-click the View Web Template Items window title bar and select **New Record**. A blank new record appears above all existing records.

   Set the values for the fields found in Table 15-12 and keep the default values for all other fields.

Table 15-12 Define first view Web template item

Field	Value
Name	Service Request Detail Applet
Item Identifier	1
Applet	Service Request Detail Applet
Applet Mode	Edit

   You may name the View Web Template Item anything you wish. The Item Identifier maps the item to a control within a Siebel Web template (.swt) file. The Applet field provides the name of the applet being included in this view. The Applet Mode field provides the mode to be used for the applet when rendering the view.

4. Save the changes.
5. Right-click the View Web Template Items window title bar and select **New Record**. A blank new record appears above all existing records.

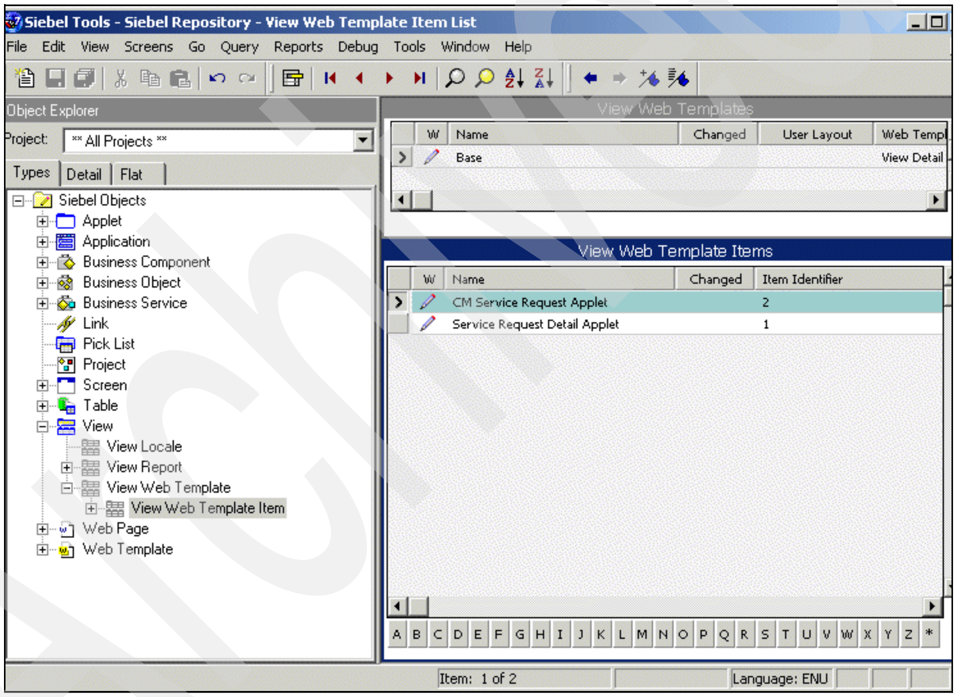
   Set the values for the fields found in Table 15-13 on page 363 and keep the default values for all other fields. See Figure 15-3 on page 363.

Table 15-13   *Define second view Web template item*

Field	Value
Name	CM Service Request Applet
Item Identifier	2
Applet	CM Attachment Applet
Applet Mode	Base

The applet CM Attachment Applet was defined in “Step 3. Creating an applet” on page 358.

6. Save the changes.



View Web Templates

W	Name	Changed	User Layout	Web Templ
	Base			View Detail

View Web Template Items

W	Name	Changed	Item Identifier
	CM Service Request Applet		2
	Service Request Detail Applet		1

Item: 1 of 2   Language: ENU

Figure 15-3   Define view Web template items

Step 10. Updating the screen object

Complete the following steps to update the screen object:

1. In Siebel Tools, select **Siebel Objects -> Screen** in Object Explorer on the left.
2. Select the Service Request window. This is where you are going to add the view created in “Step 7. Creating a view” on page 361.

3. From the menu bar, select **Tools -> Lock Project**.
4. In Object Explorer on the left, expand **Screen** and select **Screen View**.
5. Right-click the title bar of the Screen Views window and select **New Record**. A blank new record appears above all existing records.

   Set the values for the fields found in Table 15-14.

Table 15-14 Update the screen object

Field	Value
View	CMAttachment for Service Request
Category Menu Text	CM Attachment
Category Viewbar Text	CM Attachment
Menu Text	CM Attachment
Viewbar Text	CM Attachment
Sequence	1

   The view CMAttachment for Service Request was created in “Step 7. Creating a view” on page 361. The Category Menu Text field provides the text you want to display in the category menu in Siebel. The Category View bar Text field provides the text you want to display in the category tab in Siebel. The Menu Text field provides the text you want to display in the menu in Siebel. The Viewbar Text field provides the text you want to display on the viewbar tab in Siebel. The Sequence controls the location of the view tab in the screen of the Siebel application. 1 is used in the example because we want the integration modification to be highly visible.

6. Save the changes.

Step 11. Compiling

Before running the following steps, make sure that Siebel client is not running. Otherwise, the compiling will fail. The compiling process makes all modifications available to Siebel client.

1. In Siebel Tools, select **Tools -> Compile Project** from the menu bar.
2. Under Projects, select **Locked Projects**.
3. Enter the target directory for the SRF file. In our scenario, we entered `c:\sea752\client\OBJECTS\ENU\`.
4. Click **Compile**.
5. From the menu bar, select **Tools -> Unlock Project** to unlock the locked projects.

6. Exit Siebel Tools.

15.4.3 Configuring Siebel application

Before being able to retrieve documents from a Content Manager server in the Siebel Call Center, you need to configure the Siebel application. Follow these steps to complete your configuration.

Step 1. Logging on to the Siebel Call Center

Complete the following steps to log on to the Siebel Call Center:

1. Select **Start -> Programs -> Siebel Client 7.5 -> Siebel Call Center-ENU**.
2. Enter your administrator user ID and password.

Important: You must log in to Siebel as administrator.

3. Select **Connect to -> Server**.

Step 2. Defining the external data host

Complete the following steps to define the external data host:

1. In the Siebel Call Center, select **View -> Site Map -> Integration Administration -> Host Administration** from the Siebel menu bar (not the browser menu bar).
2. Click **New**.

   Set the values for the fields found in Table 15-15.

Table 15-15 Define external data host

Field	Value
Name	eClientHost.ibm.com®
Virtual Name	eClientHost
Authentication Type	leave this field blank
Authentication Value	leave this field blank

   The Name field specifies the host name for the machine where the eClient server is installed. The host name should include the domain suffix.

3. Click **Menu** and select **Save Record**.

Step 3. Defining a symbolic URL

In this step, you define the symbolic URL referenced in the calculated value field defined in 15.4.1, “Configuring business component” on page 354.

1. In the Siebel Call Center, select **View -> Site Map -> Integration Administration -> Symbolic URL Administration** from the Siebel menu bar.
2. In the Symbolic URL Administration view, click **New**.

   Set the values for the fields found in Table 15-16.

Table 15-16 Define a symbolic URL

Field	Value
Name	CMSR
URL	http://eClientHost:80/eClient82/IDMIntegrator
Host Name	eClientHost
Fixup Name	Default
Multivalue Treatment	leave this field blank
SSO Disposition	IFrame
Web Application	leave this field blank

   The Name field defines the name of the symbolic URL that is used in “Configuring business component” on page 354 when you defined the calculated field. The URL field defines the URL for invoking the eClient Integration servlet. This URL format is:

   ```
   URL scheme://virtual host name/eClient application name/IDMIntegrator
   ```

   In the Host Name field, you enter the virtual name defined in the “Step 2. Defining the external data host” on page 365.

3. Click **Menu** and select **Save Record**.

Step 4. Defining required symbolic URL arguments

Complete the following steps to define required symbolic URL arguments:

1. In the Siebel Call Center, select **Site Map -> Integration Administration -> Symbolic URL Administration** from the Siebel menu bar.
2. Select the symbolic URL CMSR that you created in “Step 3. Defining a symbolic URL” on page 366.
3. In the Symbolic URL Arguments view, click **New**.

   Set the values for the fields found in Table 15-17 on page 367.

Table 15-17 Define first symbolic URL argument

Field	**Value**
Name	method
Required	True
Argument Type	Command
Argument Value	PostRequest
Append as Argument	True
Sequence	1

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `PostRequest` in the first Argument value field, and leave the second value field blank.

4. In the Symbolic URL Arguments view, click **New**.

   Set the values for the fields found in Table 15-18.

Table 15-18 Define second symbolic URL argument

Field	**Value**
Name	eClientToken
Required	True
Argument Type	Constant
Argument Value	token
Append as Argument	True
Sequence	2

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `token` in the first Argument value field, and leave the second value field blank. `token` is the value assigned to the eClientToken field in the integration properties file Siebel.properties.

5. In the Symbolic URL Arguments view, click **New**.

   Set the values for the fields found in Table 15-19.

Table 15-19 Define third symbolic URL argument

Field	**Value**
Name	IPFile

Field	Value
Required	True
Argument Type	Constant
Argument Value	Siebel
Append as Argument	True
Sequence	3

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `Siebel` in the first Argument value field, and leave the second value field blank.

Important: Even though the integration properties file name is Siebel.properties, we enter `Siebel` as the argument value. The file extension should not be entered as part of the argument value.

6. In the Symbolic URL Arguments view, click **New**.

   Set the values for the fields found in Table 15-20.

Table 15-20 Define fourth symbolic URL argument

Field	Value
Name	ReleaseLevel
Required	True
Argument Type	Constant
Argument Value	SIEBELV75
Append as Argument	True
Sequence	4

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `SIEBELV75` in the first Argument value field, and leave the second value field blank.

7. In the Symbolic URL Arguments view, click **New**.
8. Set the values for the fields found in Table 15-21 on page 369.

Table 15-21 Define fifth symbolic URL argument

Field	Value
Name	Entity
Required	True
Argument Type	Constant
Argument Value	SiebelAttachment
Append as Argument	True
Sequence	5

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `SiebelAttachment` in the first Argument value field, and leave the second value field blank. `SiebelAttachment` is a federated search template that you define in the EIP database server.

9. In the Symbolic URL Arguments view, click **New**.

   Set the values for the fields found in Table 15-22.

Table 15-22 Define sixth symbolic URL argument

Field	Value
Name	searchBySRNumber
Required	True
Argument Type	Field
Argument Value	SR Number
Append as Argument	True
Sequence	6

searchBySRNumber is one of the federated search criteria of the search template SiebelAttachment that you define in the EIP database server. SR Number is a Siebel field that is presented in the Siebel service request application. This URL symbolic argument takes the value in the SR Number Siebel field and passes it to the EIP search template.

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `SR Number` in the first Argument value field, and leave the second value field blank.

10. In the Symbolic URL Arguments view, click **New**.
11. Set the values for the fields found in Table 15-23 on page 370.

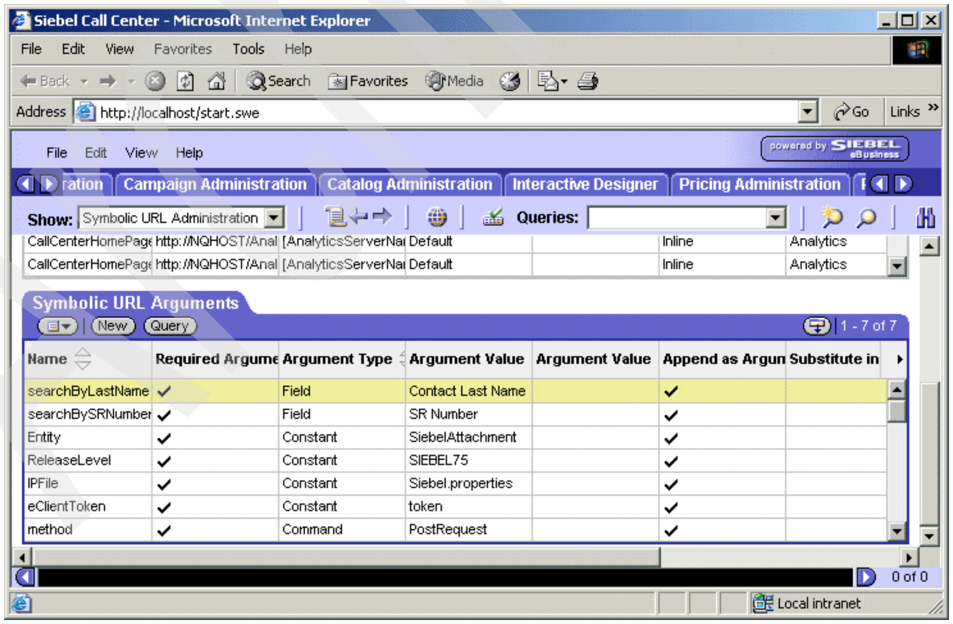
Table 15-23   Define seventh symbolic URL argument

Field	Value
Name	searchByLastName
Required	False
Argument Type	Field
Argument Value	Contact Last Name
Append as Argument	True
Sequence	7

searchByLastName is another of the federated search criteria of the search template SiebelAttachment that you define in the EIP database server. Contact Last Name is a Siebel field that is presented in the Siebel service request application. This URL symbolic argument takes the value in the Contact Last Name Siebel field and passes it to the EIP search template.

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `searchByLastName` in the first Argument value field, and leave the second value field blank.

See Figure 15-4.



Siebel Call Center - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/start.swe
Go Links
File Edit View Help
powered by SIEBEL eBusiness
ration Campaign Administration Catalog Administration Interactive Designer Pricing Administration
Show: Symbolic URL Administration
Queries:
Symbolic URL Arguments
New Query
1 - 7 of 7
0 of 0
Local intranet

CallCenterHomePag	http://NQHOST/Anal	[AnalyticsServerNa	Default		Inline	Analytics
CallCenterHomePag	http://NQHOST/Anal	[AnalyticsServerNa	Default		Inline	Analytics

Name	Required Argume	Argument Type	Argument Value	Argument Value	Append as Argun	Substitute in
searchByLastName	✓	Field	Contact Last Name		✓	
searchBySRNumber	✓	Field	SR Number		✓	
Entity	✓	Constant	SiebelAttachment		✓	
ReleaseLevel	✓	Constant	SIEBEL75		✓	
IPFile	✓	Constant	Siebel.properties		✓	
eClientToken	✓	Constant	token		✓	
method	✓	Command	PostRequest		✓	

Figure 15-4   Define symbolic URL arguments

12. Click **Menu** and select **Save Record**.

Step 5. Defining symbolic URL arguments (optional)

There are three optional symbolic URL arguments: server, userid, and password. You do not have to define them in the symbolic URL if they are specified as the server, userid, and password properties in the integration properties file. If you specify them in both the symbolic URL and the IP file, the values in the symbolic URL take precedence over the values in the IP file.

1. In the Siebel Call Center, select **Site Map -> Integration Administration -> Symbolic URL Administration** from the Siebel menu bar.
2. Select the symbolic URL CMSR that you created in "Step 3. Defining a symbolic URL" on page 366.
3. In the Symbolic URL Arguments view, click **New**.

   Set the values for the fields found in Table 15-24.

Table 15-24 Define eighth symbolic URL argument

Field	Value
Name	server
Required	True
Argument Type	Constant
Argument Value	EIPDB
Append as Argument	True
Sequence	8

   EIPDB is the EIP database server name. Do not enter the server host name as the argument value.

   The Siebel Symbolic URL Administration offers two Argument value fields. Enter `EIPDB` in the first Argument value field, and leave the second value field blank.

4. In the Symbolic URL Arguments view, click **New**.

   Set the values for the fields found in Table 15-25.

Table 15-25 Define ninth symbolic URL argument

Field	Value
Name	userid
Required	True

Field	Value
Argument Type	Constant
Argument Value	icmadmin
Append as Argument	True
Sequence	9

icmadmin must be a valid EIP user ID.

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `icmadmin` in the first Argument value field, and leave the second value field blank.

5. In the Symbolic URL Arguments view, click **New**.

   Set the values for the fields found in Table 15-26.

Table 15-26 Define tenth symbolic URL argument

Field	Value
Name	password
Required	True
Argument Type	Constant
Argument Value	password
Append as Argument	True
Sequence	10

The Siebel Symbolic URL Administration offers two Argument value fields. Enter `password` in the first Argument value field, and leave the second value field blank.

6. Click **Menu** and select **Save Record**.

Step 6. Specifying Siebel single sign-on (optional)

Siebel V7.5.2 provides a single sign-on capability. You do not need to perform this step unless you are using the Siebel single sign-on capability. In our scenario, we do not do this step.

Step 7. Setting up a new view in the Siebel Call Center

Follow these steps to set up a new view in the Siebel Call Center:

1. In the Siebel Call Center, select **View -> Site Map -> Application Administration** from the Siebel menu bar.

2. Select **Views**.
3. Click **New** in the View applet.
4. Set the values for the fields found in Table 15-27.

Table 15-27 Set up a new view

Field	value
Name	CMAttachment View for Service Request
Description	CM/Siebel integration view

You have defined the view CMAttachment View for Service Request in "Step 7. Creating a view" on page 361. In this step, you make it available in the Siebel Call Center.

5. Click **Menu** and select **Save Record**.

Step 8. Creating a responsibility

Follow these steps to create a responsibility:

1. In the Siebel Call Center, select **View -> Site Map -> Application Administration -> Responsibilities** from the Siebel menu bar.
2. Click **New** in the Responsibilities applet.
3. Set the values for the fields found in Table 15-28.

Table 15-28 Create a responsibility

Field	Value
Name	CM Attachment Manager
Description	CM Attachment Manager
Organization	Default Organization

4. Save the changes.
5. From the User applet, click **New**.
6. Select the user **John Smith**, and click **OK**.
7. From the View applet, click **New**.
8. Select the view **CMAttachment View for Service Request**, and click **OK**.
9. Save the changes.
10. Log off from the Siebel Call Center.

15.5 Setting up Content Manager and EIP

The goal of the Siebel and eClient integration is to use Siebel application to manage the business process, and use Content Manager to manage service request attachment files for Siebel. To manage unstructured data for Siebel, we must have the infrastructure in Content Manager and EIP server.

In this section, we create an item type in Content Manager to hold Siebel service request attachment files. Since the Siebel integration only works with EIP federated server, you must also create a federated entity and a federated search template in EIP.

15.5.1 Creating attributes in Content Manager

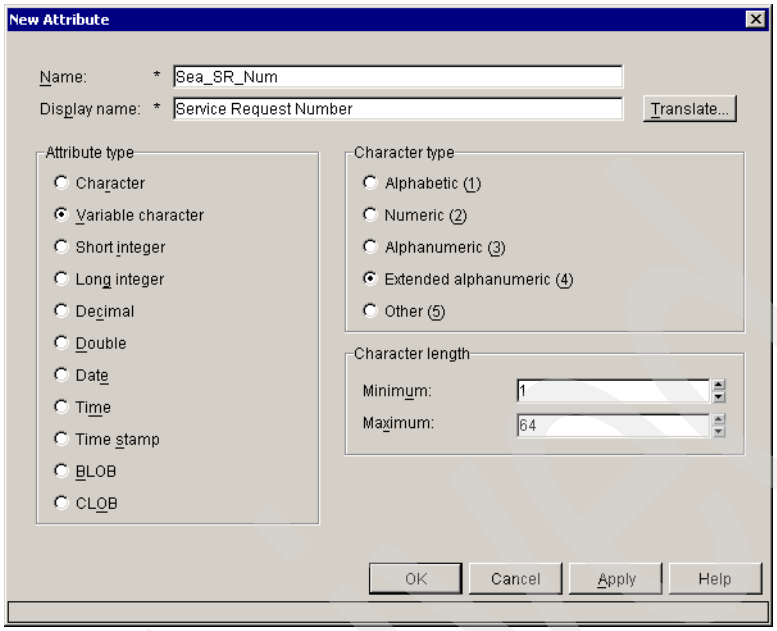
In this section, we create two Content Manager attributes that have the same properties as the fields SR Number and Contact Last Name defined in Siebel. In addition, we create the third attribute to present the service request document sequence number.

1. Log on to Content Manager system administration client by selecting **Start -> Programs -> IBM Content Manager for Multiplatforms V8.2 -> System Administration Client**.
2. In the Navigation pane on the left, select **Content Manager -> ICMNLSDB -> Data Modeling -> Attributes**.
3. Right-click **Attributes**, and select **New** to create the first field.

   Set the values for the fields found in Table 15-29. See Figure 15-5 on page 375.

Table 15-29 Define attribute Sea_SR_Num

Field	**Value**
Name	Sea_SR_Num
Display name	Service Request Number
Attribute type	Variable character
Character type	Extended alphanumeric
Minimum	1
Maximum	64



Name: * Sea_SR_Num

Display name: * Service Request Number

Attribute type: Character, [✓ Variable character], Short integer, Long integer, Decimal, Double, Date, Time, Time stamp, BLOB, CLOB

Character type: Alphabetic (1), Numeric (2), Alphanumeric (3), [✓ Extended alphanumeric (4)], Other (5)

Character length

Minimum: 1

Maximum: 64

Figure 15-5 Define attribute in Content Manager

4. Click **OK** to save the attribute.
5. Right-click **Attributes**, and select **New** to create the second field.

   Set the values for the fields found in Table 15-30.

Table 15-30 Define attribute Sea_LastName

Field	**Value**
Name	Sea_LastName
Display name	Last Name
Attribute type	Variable character
Character type	Extended alphanumeric
Minimum	1
Maximum	50

6. Click **OK** to save the attribute.
7. Right-click **Attributes**, and select **New** to create one more field.

   Set the values for the fields found in Table 15-31 on page 376.

Table 15-31 Define attribute Sea_SR__Doc_Num

Field	Value
Name	Sea_SR__Doc_Num
Display name	Service Request Document Sequence Number
Attribute type	Short integer
Minimum	0
Maximum	32767

8. Click **OK** to save the attribute.

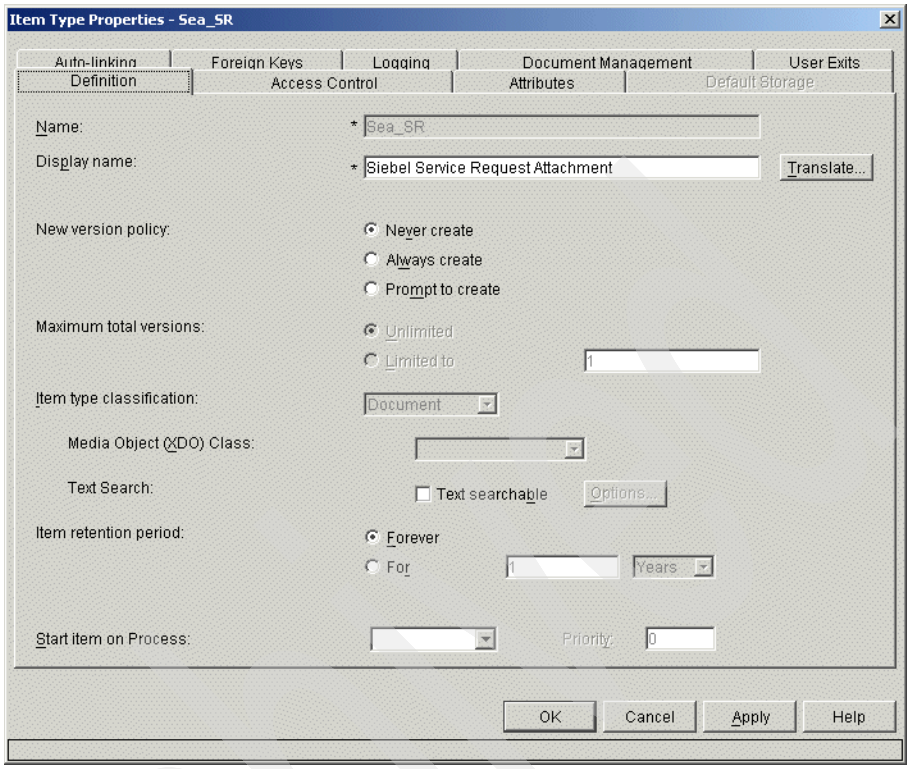
15.5.2 Creating item type in Content Manager

In this section, we create an item type in Content Manager system administration client. This item type is used to hold unstructured data (service request attachment) for Siebel application.

1. Log on to Content Manager system administration client by selecting **Start -> Programs -> IBM Content Manager for Multiplatforms V8.2 -> System Administration Client**.
2. In the Navigation pane on the left, select **Content Manager -> ICMNLSDB -> Data Modeling -> Item Types**.
3. Right-click **Item Types**, and select **New**.
4. On the **Definition** tab, set the values for the fields found in Table 15-32. See Figure 15-6 on page 377.

Table 15-32 Define item type - Definition

Field	Value
Name	Sea_SR
Display name	Siebel Service Request Attachment
New version policy	Never create
Item type classification	Document
Text Search	False



Item Type Properties - Sea_SR

Auto-linking | Foreign Keys | Logging | Document Management | User Exits | Definition | Access Control | Attributes | Default Storage

Name: * Sea_SR

Display name: * Siebel Service Request Attachment [Translate...]

New version policy: [✓ Never create] Always create Prompt to create

Maximum total versions: [✓ Unlimited] Limited to 1

Item type classification: Document

Media Object (XDO) Class:

Text Search: ☐ Text searchable [Options...]

Item retention period: [✓ Forever] For 1 Years

Start item on Process: Priority: 0

OK | Cancel | Apply | Help

Figure 15-6 Define item type Sea_SR

5. On the Attributes window, set the values for the fields found in Table 15-33.

Table 15-33 Define item type - Attributes

Attribute	**Required**
Sea_SR_Num	True
Sea_LastName	False
Sea_SR_Doc_Num	False

6. On the Document Management window, set the values for the fields found in Table 15-34.

Table 15-34 Define item type - Document Management

Part type	**Access control list**	**RMDB/Collection**	**Version**
ICMBASE	PublicReadACL	take default	Never create

Part type	Access control list	RMDB/Collection	Version
ICMANNOTATION	PublicReadACL	take default	Never create
ICMNOTELOG	PublicReadACL	take default	Never create

7. Click **OK** to save the item type.

15.5.3 Populating unstructured data in Content Manager

Before Siebel users are able to retrieve the unstructured data (Siebel service request attachment) from a Content Manager server, you must import these files into Content Manager. To import these files, you must find out the service request number and other related information that associates to these files.

Find attribute information before populating Content Manager

1. Log on to the Siebel Call Center as user John Smith by selecting **Start -> Programs -> Siebel Client 7.5.2 -> Siebel Call Center - ENU**.
2. Select the **Service** tab.
3. In the Show drop-down list, select **All Service Requests**. All service requests are displayed in the Service Request applet.
4. Record the SR_Num and Contact Last Name for a service request. For demonstration purposes, we use "`1-229`" as SR_Num and “`Smith`” as Contact Last Name.

Import service request attachment files into Content Manager

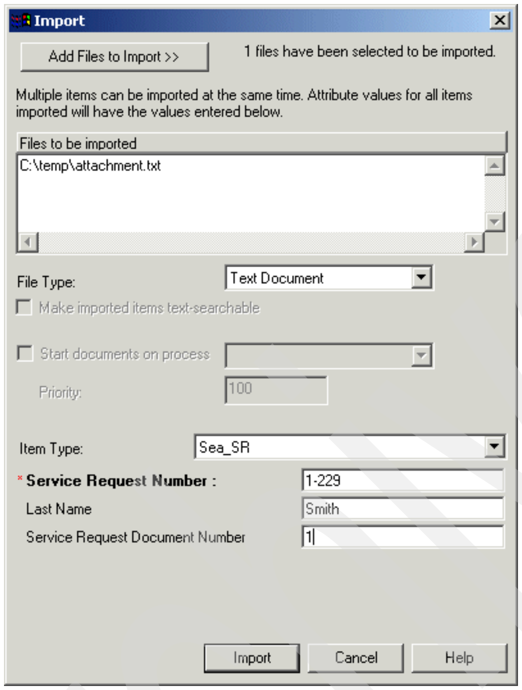
5. Log on to the Content Manager window client with the user ID icmadmin by selecting **Start -> Programs -> IBM Content Manager V8 -> Client for Windows**.
6. Select **File -> Import** from the menu bar.

   Set the values for the fields found in Table 15-35. See Figure 15-7 on page 379.

Table 15-35 Import service request attachment file

Field	Value
Files to be imported	c:\temp\attachment.txt
File Type	Text Document
Item Type	Sea_SR
Service Request Number	1-229

Field	Value
Last Name	Smith
Search Request Document Number	1



1 files have been selected to be imported.

Multiple items can be imported at the same time. Attribute values for all items imported will have the values entered below.

Files to be imported
C:\temp\attachment.txt

File Type: Text Document
Make imported items text-searchable
Start documents on process
Priority: 100

Item Type: Sea_SR
* **Service Request Number :** 1-229
Last Name: Smith
Service Request Document Number: 1

Figure 15-7 Import service request attachment

7. Repeat step 7 to import more attachments for service request 1-229.

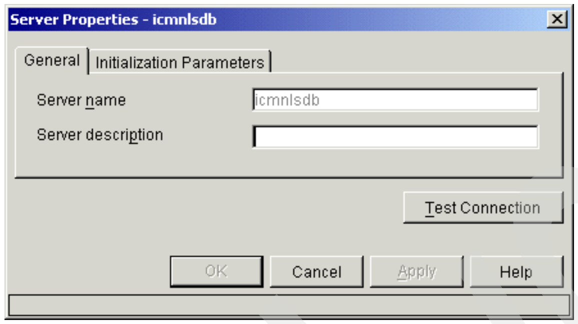
15.5.4 Preparing EIP server

You must make the Content Manager server available to the EIP server before you continue to set up the federated entity and federated search template. Follow these steps to add a new Content Manager server to EIP and refresh the Content Manager inventory:

1. Log on to EIP administration client by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Administration**.
2. In the Navigation pane on the left, select **Enterprise Information Portal -> EIPDB -> Servers**.
3. Right-click **Servers**, select **New -> Content Manager v8**.

4. Enter `icmnlsdb` in the Server name field. See Figure 15-8.

Important: You must enter the Library Server database name of the Content Manager server in the Server name field. Do not use the host name of the Content Manager server.



Server name: icmnlsdb
Server description:

Figure 15-8 Add Content Manager server in EIP administration client

5. Click **Test Connection**.
6. Enter a valid user ID and password of the Library Server, and click **OK**.

   You should see a message: "`The connection to icmnlsdb was successful`."
7. Click **OK** twice to save the changes.
8. Right-click the **icmnlsdb** server entry in the Contents of servers pane on the right, and select **Refresh Server Inventory**.
9. A message is displayed: "`Refreshing the inventory can take awhile. A message will be issued when the refresh is completed. Do you want to continue with the refresh?`". Click **Yes**.
10. Another message is displayed: "`The inventory was refreshed for icmnlsdb. No changes were found that affect existing search templates`." Click **OK**.
11. Select **Tools -> Server Inventory Viewer** from the menu bar.
12. The Server inventory viewer window is displayed. Scroll down the list and make sure that there are three entries for the item type Sea_SR.

15.5.5 Creating EIP federated entity

When you define the EIP federated entity, you map EIP federated attributes to native attributes in the back-end server, such as Content Manager.

Complete the following steps to define the EIP federated entity.

1. Log on to EIP administration client by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Administration**.
2. In the Navigation pane on the left, select **Enterprise Information Portal -> EIPDB -> Federated Entities**.
3. Right-click **Federated Entities**, and select **New -> Nonwizard**.

   Set the values of the Name and Description fields as shown in Table 15-36. Keep the default values for the other fields.

Table 15-36 Define EIP federated entity

Field	Value
Name	SiebelAttachment
Description	SiebelAttachment

4. Click **Add** to define a new EIP attribute.

   Set the values of the fields as found in Table 15-37.

Table 15-37 Define EIP federated attribute - SR_Num

Field	Value
Add federated attribute	True
Name	SR_Num
Type	Variable character
Length	64
Nullable	False
Queryable	True
Updateable	True
Text Searchable	False

5. Click **OK** to save the EIP federated attribute.
6. Click **Add** to define a new EIP attribute.

   Set the values of the fields as found in Table 15-38.

Table 15-38 Define EIP federated attribute - LastName

Field	Value
Add federated attribute	True

Field	Value
Name	SR_Num
Type	Variable character
Length	50
Nullable	True
Queryable	True
Updateable	True
Text Searchable	False

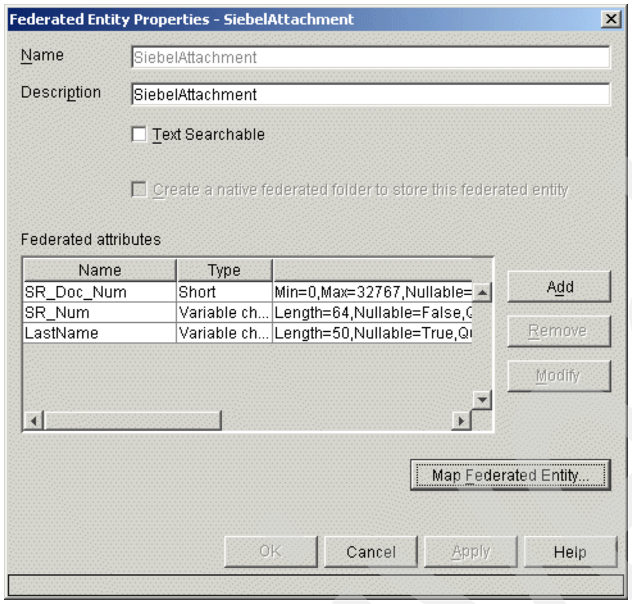
7. Click **OK** to save the EIP federated attribute.
8. Click **Add** again.

   Set the values of the fields found in Table 15-39.

Table 15-39 Define EIP federated attribute - SR_Doc_Num

Field	Value
Add federated attribute	True
Name	SR_Doc_Num
Type	Short
Minimum	0
Maximum	32767
Nullable	True
Queryable	True
Updateable	True
Text Searchable	False

9. Click **OK** to save the EIP federated attribute.
10. Now, your Federated Entity Properties window should look like Figure 15-9 on page 383.


Federated Entity Properties - SiebelAttachment

Name: SiebelAttachment

Description: SiebelAttachment

☐ Text Searchable

☐ Create a native federated folder to store this federated entity

Federated attributes

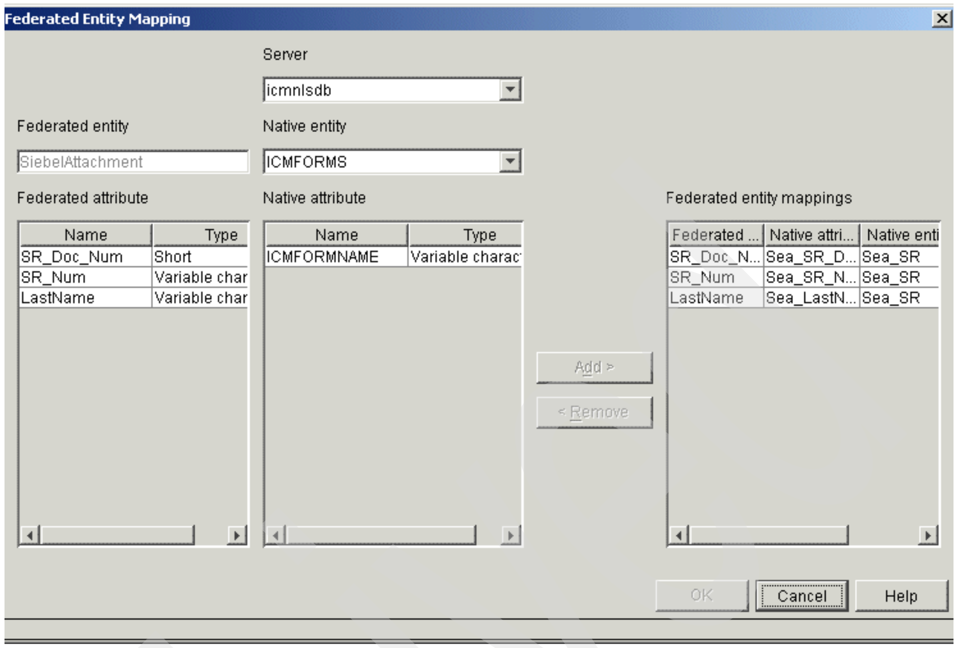
Name	Type	
SR_Doc_Num	Short	Min=0,Max=32767,Nullable=
SR_Num	Variable ch...	Length=64,Nullable=False,(
LastName	Variable ch...	Length=50,Nullable=True,Q

Figure 15-9 Define EIP federated entity

11. Click **Map Federated Entity**. The Federated Entity Mapping window is displayed.
12. Select **SiebelAttachment** in the Federated entity field.
13. Select **icmnlsdb** in the Server field.
14. Select **Sea_SR** in the Native entity field.
15. There are three federated attributes and three native attributes. Map the federated attribute to the native attribute as shown in Table 15-40. Your federated entity mapping window looks like Figure 15-10 on page 384.

Table 15-40 Map the federated attributes to the native attributes

Federated attribute	Native attribute
SiebelAttachment.**SR_Num**	Sea_SR.**Sea_SR_Num**
SiebelAttachment.**LastName**	Sea_SR.**Sea_LastName**
SiebelAttachment.**SR_Doc_Num**	Sea_SR.**Sea_SR_Doc_Num**



Federated Entity Mapping

Server: icmnlsdb

Federated entity: SiebelAttachment

Native entity: ICMFORMS

Federated attribute

Name	Type
SR_Doc_Num	Short
SR_Num	Variable char
LastName	Variable char

Native attribute

Name	Type
ICMFORMNAME	Variable charac

Federated entity mappings

Federated ...	Native attri...	Native enti
SR_Doc_N...	Sea_SR_D...	Sea_SR
SR_Num	Sea_SR_N...	Sea_SR
LastName	Sea_LastN...	Sea_SR

Figure 15-10 Map federated attributes to native attributes

16. Click **OK** to save the federated attribute mapping.
17. Click **OK** to save the federated entity.

15.5.6 Creating EIP federated search template

In order to search and retrieve documents through eClient, you must define the EIP federated search template. Complete the following steps to create a EIP federated search template.

1. Log on to EIP administration client by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Administration**.
2. In the Navigation pane on the left, select **Enterprise Information Portal -> EIPDB -> Search Templates**.
3. Right-click **Search Templates**, and select **New -> Nonwizard**.

   Set the values of the Name and Description fields as shown in Table 15-41.

Table 15-41 Define EIP search template

Field	Value
Name	SiebelAttachment

Field	Value
Description	Attachment in IBM Content Manager - Service Request

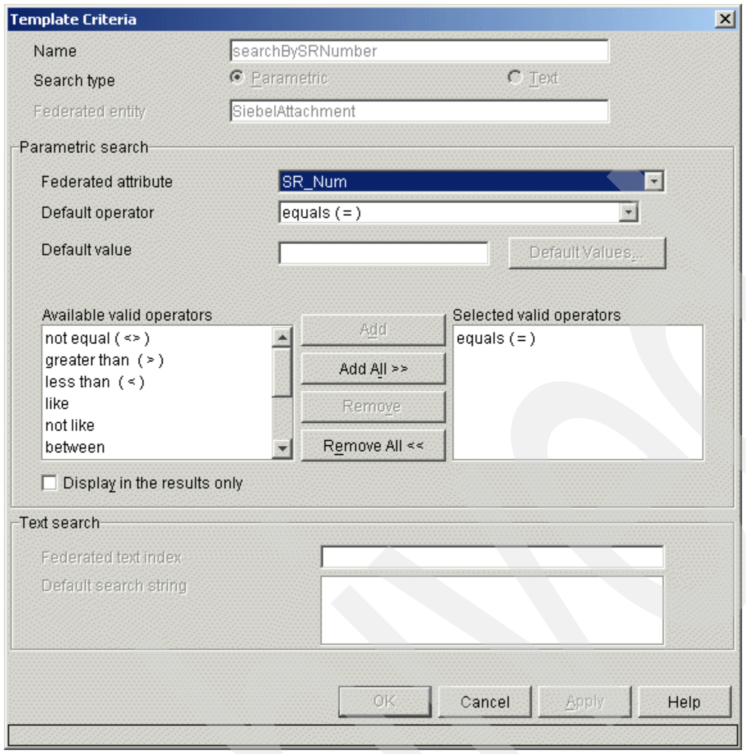
4. Since you are using user ID icmadmin in the integration properties file, you must grant icmadmin access to this search template. Highlight user ID icmadmin in the Available groups/users field and click **Add**.
5. Click **Add** next to the Template criteria field.

   Set the values for the fields found in Table 15-42. See Figure 15-11 on page 386.

Table 15-42 Define template criteria - searchBySRNumber

Field	Value
Name	searchBySRNumber
Type	Parametric
Federated entity	SiebelAttachment
Federated attribute	SR_Num
Default operator	equals (=)
Selected valid operators	equals (=)
Display in the results only	False

Restriction: The Siebel integration only supports the equal (=) operator in the EIP search template.


Template Criteria

Name: searchBySRNumber

Search type: [✓ Parametric] Text

Federated entity: SiebelAttachment

Parametric search

Federated attribute: SR_Num

Default operator: equals (=)

Default value:

Available valid operators: not equal (<>), greater than (>), less than (<), like, not like, between

Selected valid operators: equals (=)

[ ] Display in the results only

Text search

Federated text index:

Default search string:

Figure 15-11 Define template criteria

6. Click **OK** to save the template criteria.
7. Click **Add** next to the Template criteria field.

   Set the values for the fields found in Table 15-43.

Table 15-43 Define template criteria - searchByLastName

Field	**Value**
Name	searchByLastName
Type	Parametric
Federated entity	SiebelAttachment
Federated attribute	LastName
Default operator	equals (=)
Selected valid operators	equals (=)
Display in the results only	False

8. Click **OK** to save the template criteria.
9. Click **Add** next to the Template criteria field.

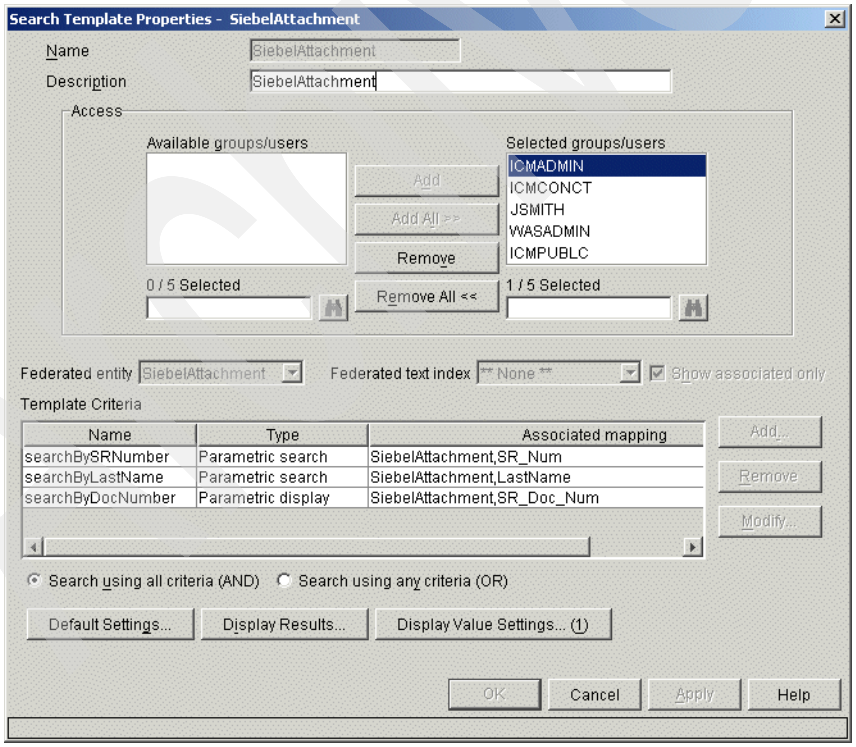
   Set the values for the fields found in Table 15-44.

Table 15-44 Define template criteria - searchByDocNumber

Field	Value
Name	searchByDocNumber
Type	Parametric
Federated entity	SiebelAttachment
Federated attribute	SR_Doc_Num
Display in the results only	True

10. Click **OK** to save the template criteria.

   The New Search Template window looks like Figure 15-12.


Search Template Properties - SiebelAttachment

Name: SiebelAttachment
Description: SiebelAttachment

Access

Available groups/users: (none) — 0 / 5 Selected

Selected groups/users: ICMADMIN, ICMCONCT, JSMITH, WASADMIN, ICMPUBLC — 1 / 5 Selected

Federated entity: SiebelAttachment
Federated text index: ** None **
Show associated only: ✓

Template Criteria

Name	Type	Associated mapping
searchBySRNumber	Parametric search	SiebelAttachment,SR_Num
searchByLastName	Parametric search	SiebelAttachment,LastName
searchByDocNumber	Parametric display	SiebelAttachment,SR_Doc_Num

[✓ Search using all criteria (AND)] Search using any criteria (OR)

Figure 15-12 Define search template

11. Click **OK** to save the search template.

15.6 Verification

After completing the installation and configuration of Siebel Integration for IBM Content Manager, we need to verify if it is working as we designed.

During the configuration, we:

- Modified the business component Service Request for a Siebel entity Service Request.
- Created a view CMAttachment View for Service Request.

These modifications allow Siebel end users to retrieve and display documents stored in Content Manager.

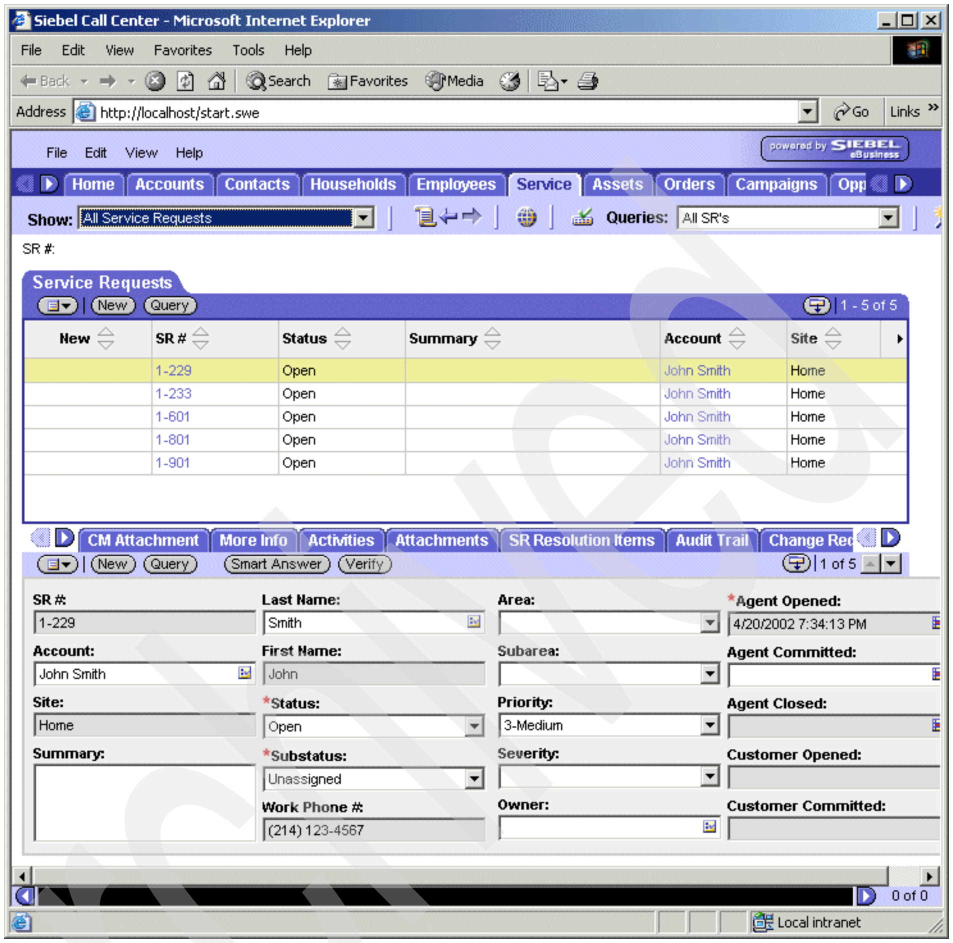
Before continuing the verification steps, make sure that you completed the following setup, as described in 15.5, “Setting up Content Manager and EIP” on page 374:

- Created a Content Manager item type Sea_SR to store attachments for the Service Request in Siebel.
- Loaded attachment files into the Content Manager item type Sea_SR for service request 1-229.
- Created an EIP entity SiebelAttachment that is mapped to the Content Manager item type Sea_SR.
- Created an EIP search template SiebelAttachment to retrieve stored attachments in Content Manager server.

Follow these steps to validate your configuration:

1. Log on to the Siebel Call Center as user John Smith by selecting **Start -> Programs -> Siebel Client 7.5 -> Siebel Call Center-ENU**.
2. Select the **Service** tab.
3. From the Show drop-down list, select **All Service Requests**. All service requests are displayed in the Service Request applet.
4. Select service request **1-229**. See Figure 15-13 on page 389.

   Notice that CM Attachment is the first tab in the bottom half of the window. It was added in “Step 10. Updating the screen object” on page 363.


Siebel Call Center - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/start.swe Go Links
File Edit View Help
powered by SIEBEL eBusiness
Home Accounts Contacts Households Employees Service Assets Orders Campaigns Opp
Show: All Service Requests
Queries: All SR's
SR #:
Service Requests
New Query
1 - 5 of 5

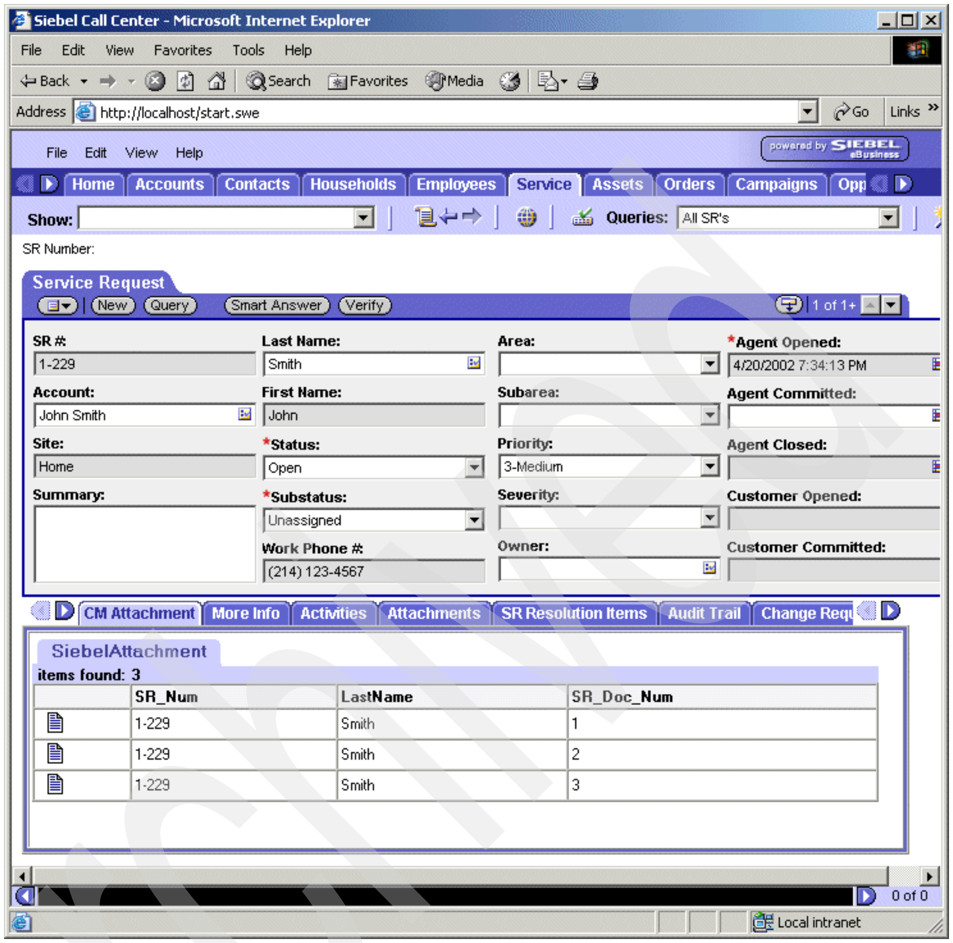
New	SR #	Status	Summary	Account	Site
	1-229	Open		John Smith	Home
	1-233	Open		John Smith	Home
	1-601	Open		John Smith	Home
	1-801	Open		John Smith	Home
	1-901	Open		John Smith	Home

CM Attachment More Info Activities Attachments SR Resolution Items Audit Trail Change Req
New Query Smart Answer Verify
1 of 5
0 of 0
Local intranet

SR #: 1-229	Last Name: Smith	Area:	*Agent Opened: 4/20/2002 7:34:13 PM
Account: John Smith	First Name: John	Subarea:	Agent Committed:
Site: Home	*Status: Open	Priority: 3-Medium	Agent Closed:
Summary:	*Substatus: Unassigned	Severity:	Customer Opened:
	Work Phone #: (214) 123-4567	Owner:	Customer Committed:

Figure 15-13 Service request in the Siebel Call Center

5. Click the **CM Attachment** tab. A list of attachment for service request 1-229 is displayed in the SiebelAttachment pane. See Figure 15-14 on page 390.


Siebel Call Center - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/start.swe
Go Links
File Edit View Help
powered by SIEBEL eBusiness
Home Accounts Contacts Households Employees Service Assets Orders Campaigns Opp
Show: Queries: All SR's
SR Number:
Service Request
New Query Smart Answer Verify 1 of 1+
CM Attachment More Info Activities Attachments SR Resolution Items Audit Trail Change Requ
0 of 0
Local intranet

SR #:	1-229
Last Name:	Smith
Area:	
*Agent Opened:	4/20/2002 7:34:13 PM
Account:	John Smith
First Name:	John
Subarea:	
Agent Committed:	
Site:	Home
*Status:	Open
Priority:	3-Medium
Agent Closed:	
Summary:	
*Substatus:	Unassigned
Severity:	
Customer Opened:	
Work Phone #:	(214) 123-4567
Owner:	
Customer Committed:	

SiebelAttachment

items found: 3

	SR_Num	LastName	SR_Doc_Num
🗎	1-229	Smith	1
🗎	1-229	Smith	2
🗎	1-229	Smith	3

Figure 15-14 Unstructured data is retrieved through eClient

6. Click the icon next to the Service Request Number to open the attachment.

   If the attachment opens as expected, then the verification process is complete.

Integrating with e-mail server

In this chapter, we describe the e-mail feature in eClient. With the e-mail feature enabled in eClient, a user is able to mail documents stored in Content Manager server to other users.

This chapter covers the following topics:

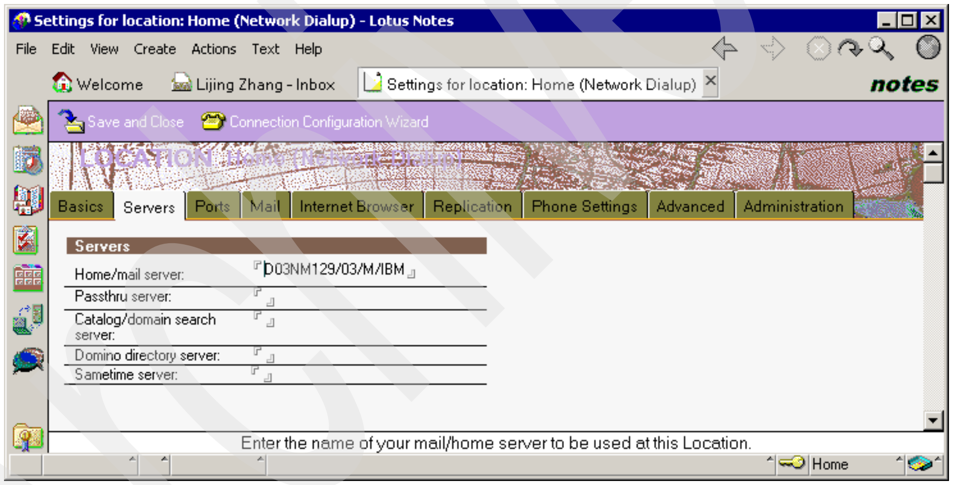
- Identifying e-mail server
- Configuring eClient to enable the e-mail feature
- E-mailing documents as attachment

16.1 Identifying e-mail server

Before configuring eClient, you need to identify an e-mail server that the eClient application server uses to send stored documents. Contact the e-mail system administrator to obtain the e-mail server name.

In this section, we discuss a quick way to identify the e-mail server that you are connecting to if you are using Lotus® Notes® as your e-mail server.

1. Log on to Lotus Notes by selecting **Start -> Programs -> Lotus Applications -> Lotus Notes**.
2. From the menu bar, select **File -> Preferences -> Location Preferences**.
3. Go to **Servers** tab.
4. The mail server name is in the Home/mail server field as shown in Figure 16-1. In this example, D03NM129 is the mail server.


Settings for location: Home (Network Dialup) - Lotus Notes
File Edit View Create Actions Text Help
Welcome
Lijing Zhang - Inbox
Settings for location: Home (Network Dialup)
notes
Save and Close
Connection Configuration Wizard
LOCATION Home (Network Dialup)
Basics
Servers
Ports
Mail
Internet Browser
Replication
Phone Settings
Advanced
Administration
Servers
Home

Home/mail server: D03NM129/03/M/IBM
Passthru server:
Catalog/domain search server:
Domino directory server:
Sametime server:

Enter the name of your mail/home server to be used at this Location.

Figure 16-1 Mail server in Lotus Notes

5. Open a command window.
6. Run the command `ping D03NM129`.
7. This returns the fully qualified host name and IP address for the mail server. In our example, it is d03nm129.boulder.ibm.com.

16.2 Configuring eClient to enable the e-mail feature

You must set three parameters in the IDM.properties file to enable e-mailing documents through eClient. By default, the IDM.properties file is in the C:\CMeClient directory.

- **emailenabled**: Set to true to enable e-mail; set to false to disable it.
- **mailUser**: Set to a valid user ID on the mail server; returned mail goes to the user ID.
- **mailHost**: Set to the IP address or fully qualified host name of the SMTP mail server.

Example 16-1 is a portion of a sample IDM.properties file.

Example 16-1 Sample IDM.properties file

```
emailEnabled=true
mailUser=admin@us.ibm.com
mailHost=d03nm129.boulder.ibm.com
```

16.3 E-mailing documents as attachment

After configuration changes are made in the IDM.properties file, you need to make the new changes effective. If you do not have Property Daemon enabled, you must stop the eClient application server and restart it:

1. Open a command window.
2. Run the command **`cd c:\CMeClient\Save`**.
3. Run the command **`stopIDMServer.bat`** to stop the server.
4. Run the command **`cd c:\CMeClient\Save`**.
5. Run the command **`startIDMServer.bat`** to restart the server.

Now, you are ready to e-mail stored documents in Content Manager as attachments.

1. Open a browser and enter `http://<hostname>/eClient82/IDMInit`.
2. Enter the user ID and password. Select **serverICMNLSDB(CM8)**. Click **OK**.
3. Log on to eClient.
4. Click **Search** on the eClient home page.
5. Select the **NOINDEX** item type.

6. Perform a search and a list of items is displayed on the Search results window.
7. Select the first document by selecting the check box to its left.
8. Choose the **E-mail document** option from the drop-down box at the top, as shown in Figure 16-2.

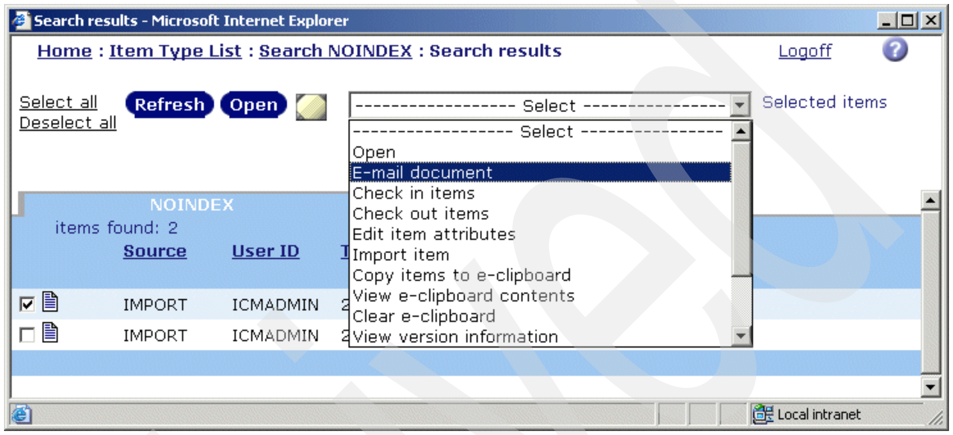
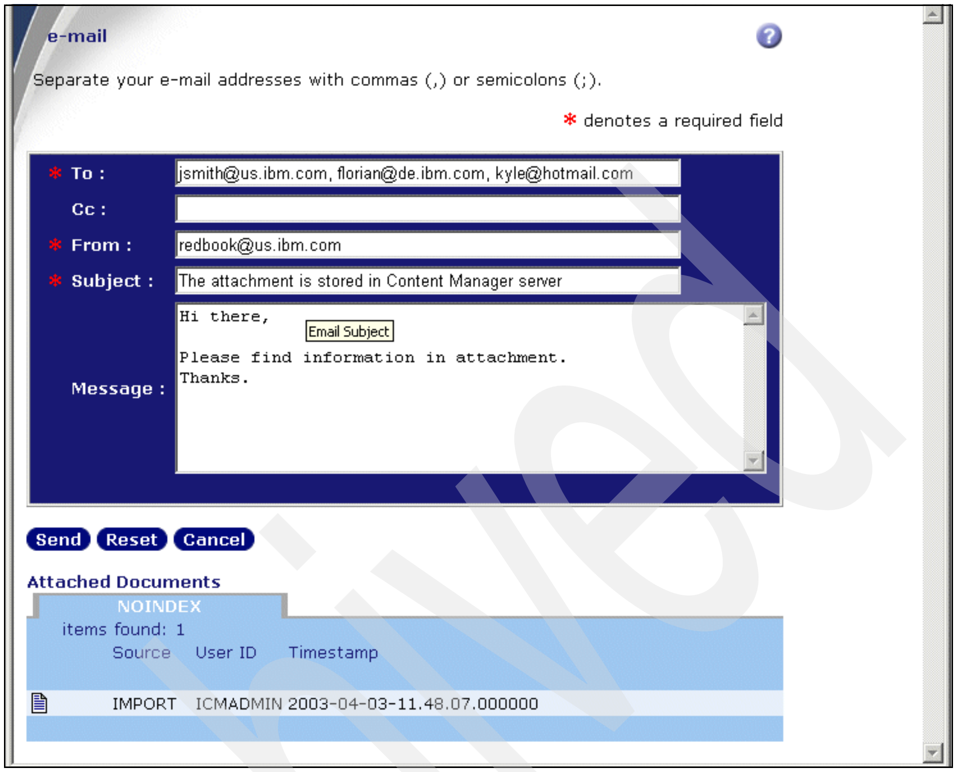


Figure 16-2 Choose the e-mail document option

9. Enter valid values fo the To, From, and Subject fields. An e-mail user should be familiar with this window. You can write an e-mail as you do in your regular e-mail client application. See Figure 16-3 on page 395.



Separate your e-mail addresses with commas (,) or semicolons (;).

* denotes a required field

* To :	jsmith@us.ibm.com, florian@de.ibm.com, kyle@hotmail.com
Cc :	
* From :	redbook@us.ibm.com
* Subject :	The attachment is stored in Content Manager server
Message :	Hi there, Please find information in attachment. Thanks.

Attached Documents

NOINDEX

items found: 1

Source	User ID	Timestamp
IMPORT	ICMADMIN	2003-04-03-11.48.07.000000

Figure 16-3 Send mail with attachment

10. Click **Send** to send the mail with an attachment. In this example, the e-mail is sent to three persons. The e-mail contains a short message and one attachment. The attachment is stored in and retrieved from the Content Manager server.

If you receive the error message “`java.lang.NullPointerException`” in the browser while e-mailing documents stored in the Content Manager V8 server, check if the Resource Manager (on which the document is stored) is running. You may do so by retrieving and displaying the same document in the eClient for a quick verification. If you fail to retrieve the same document, it is very likely that the Resource Manager is not running. To start the Resource Manager:

1. On the Resource Manager machine, open a command window.
2. Run the command **`cd C:\WebSphere\AppServer\bin`**.
3. Run the command **`startServer icmrm`**.

17

Single sign-on

In this chapter, we describe how to configure IBM Directory Server Version 5.1, WebSphere Application Server V5.0, and Content Manager Server V8.2 to enable the single sign-on feature in eClient V8.2 server.

This chapter covers the following topics:

- Installing and configuring Directory Server V5.1
- Configuring Content Manager V8.2 for LDAP
- Configuring Content Manager V8.2 for SSO
- Importing LDAP users into Content Manager V8.2
- Configuring WebSphere Application Server V5.0
- Verification

17.1 Introduction

The goal of single sign-on is for an enterprise to be able to have one network identity for each user, which allows centralized management of the various roles that a user may have in different applications, so that correct rules can be applied without duplication of the user data and without requiring multiple identities for the user.

Single sign-on is the process whereby users provide their credentials, user identity, and password and/or token, once within a session. These credentials are available to all enterprise applications for which single sign-on is enabled without prompting the user to re-enter a user name and password.

The Lightweight Third Party Authentication (LTPA) mechanism developed by IBM enables single sign-on between various application servers. A token, the transient cookie LtpaToken, is generated by the authenticating server. The cookie is encrypted using LTPA keys that must be shared among all single sign-on participating servers. The cookie contains user authentication information, the network domain in which it is valid for single sign-on, and an expiration date.

The token, issued to the Web user in a cookie, is called a transient cookie. This means that the cookie resides in the browser memory, is not stored on the user's computer system, and expires when the user closes the browser.

The general requirements for enabling single sign-on using LTPA are as follows:

- All single sign-on participating servers have to use the same user registry (for example the LDAP server).
- All single sign-on participating servers must be in the same DNS domain (cookies are issued with a domain name and will not work in a domain other than the one for which it was issued).
- The browser must be configured to accept cookies.
- Server time and time zone must be correct. The single sign-on token expiration time is absolute.
- All servers participating in the single sign-on scenario must be configured to share LTPA keys.

17.1.1 Introducing the scenario

For our scenario, we perform the following procedures:

- Use the IBM Directory Server V5.1 as a central repository to store user information.
- Enable the single sign-on feature in the Content Manager V8.2 server.

- Import user information into the Content Manager V8.2 server from the LDAP server.
- Configure LDAP user registry and enable security in the WebSphere Application Server V5.0.

After completing the above configuration, the single sign-on feature will be enabled in the WebSphere Application Server and the Content Manager server. If you have two applications in WebSphere Application Server, and both of them are single sign-on enabled, you only have to enter a user name and password once to log on to both applications. Also, you do not need a password to log on to the Content Manager server, since you have enabled its single sign-on.

For example, you have installed both eClient and WebSphere Portal Server. You log on to the WebSphere Portal Server first. At this time, you must submit the user ID and password to log on. After you log on to the WebSphere Portal Server, log on to eClient. Since the WebSphere Application Server already has the user credentials, you do not have to enter the password again in order to log on to eClient.

17.2 Installing and configuring Directory Server V5.1

This section provides the steps to install and configure the IBM Directory Server V5.1 on a Windows platform. For our scenario, we use it as the central repository to store user information.

17.2.1 Installing Directory Server V5.1

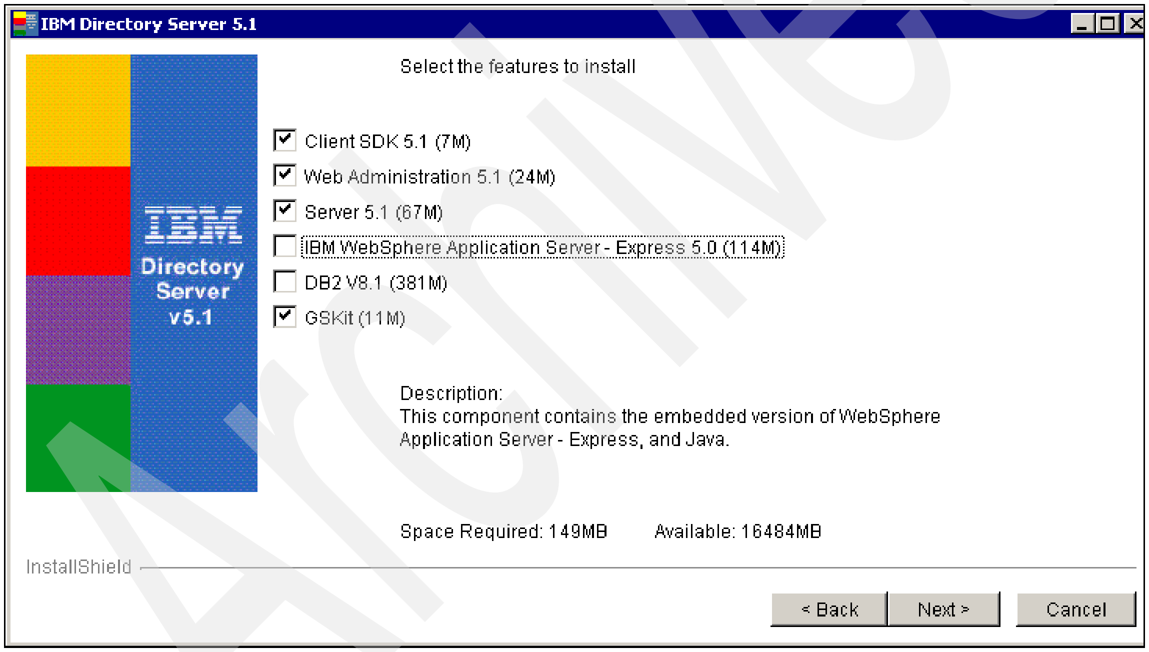
If you are running a previous version of Directory Server, refer to the *IBM Directory Server V5.1 Installation and Configuration Guide* for the upgrading procedure. This guide may be found at:

http://publib.boulder.ibm.com/tividd/td/IBMDirectoryServer5.1.html

This section demonstrates the standard, first-time Directory Server V5.1 installation steps:

1. Reboot the computer where you are installing the IBM Directory Server.
2. Log on using an administrator ID.
3. Insert the CD in your CD-ROM drive. If the CD-ROM does not automatically start the installation program:
   a. Go to \ids_ismp directory on the CD.
   b. Double-click **`setup.exe`**.

4. Select the language which you want to use during the installation. Click **OK**.
5. On the Welcome window, click **Next**.
6. After reviewing the software license agreement, click **I accept the terms** in the license agreement. Click **Next**.
7. Any previously installed components are displayed (including DB2). Click **Next**.
8. Enter `C:\LDAP` in the Directory Name field. This is the installation directory. Click **Next**.
9. Select the language you want to use in IBM Directory Server 5.1. Click **Next**.
10. Select **Custom** and click **Next**.
11. All available components are displayed on the next window as shown in Figure 17-1.



Select the features to install

- [✓] Client SDK 5.1 (7M)
- [✓] Web Administration 5.1 (24M)
- [✓] Server 5.1 (67M)
- [ ] IBM WebSphere Application Server - Express 5.0 (114M)
- [ ] DB2 V8.1 (381M)
- [✓] GSKit (11M)

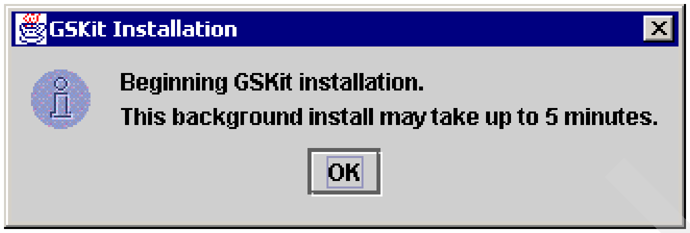
Description:
This component contains the embedded version of WebSphere Application Server - Express, and Java.

Space Required: 149MB   Available: 16484MB

Figure 17-1 Directory Server V5.1 components

Take the default components selection in Figure 17-1 and click **Next**.

12. Review the installation option and click **Next** to start copying files.
13. Click **OK** on the GSKit Installation window in Figure 17-2 on page 401.


GSKit Installation

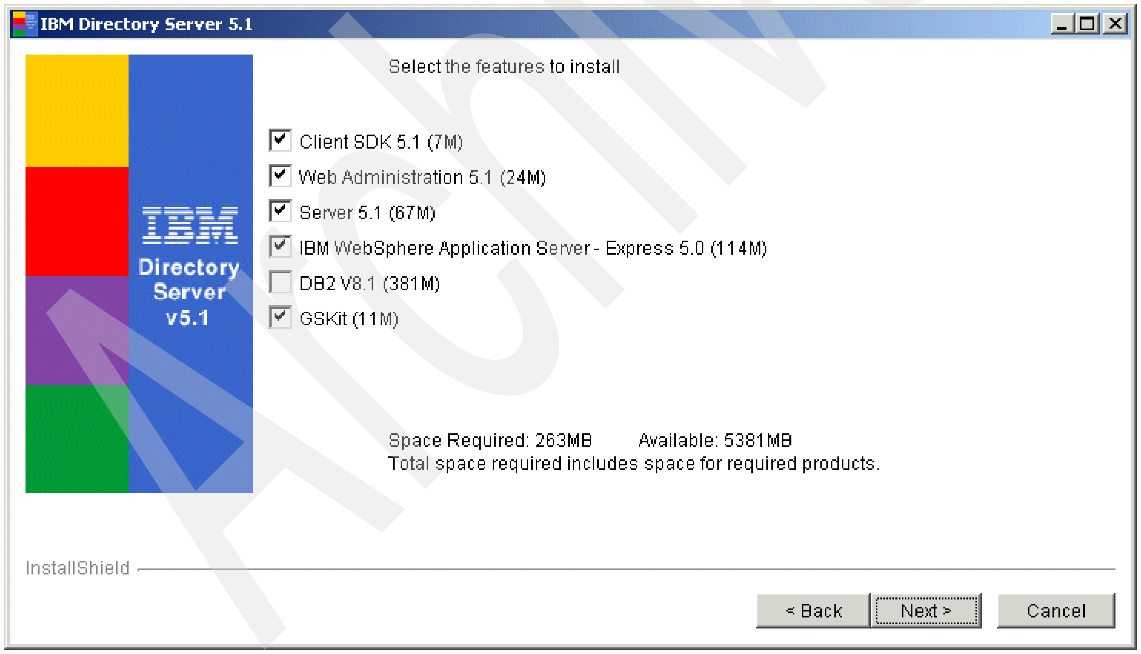
Beginning GSKit installation.
This background install may take up to 5 minutes.

Figure 17-2 Installing GSKit

Be patient. It may take a few minutes.

Important: Do not move the mouse and type on the keyboard during the installation.

14. The installation procedure installs the embedded version of WebSphere Application Server - Express per your selection. It is required by the Directory Server Web Administration Tool. See Figure 17-3.


IBM Directory Server 5.1

Select the features to install

- [✓] Client SDK 5.1 (7M)
- [✓] Web Administration 5.1 (24M)
- [✓] Server 5.1 (67M)
- [✓] IBM WebSphere Application Server - Express 5.0 (114M)
- [ ] DB2 V8.1 (381M)
- [✓] GSKit (11M)

Space Required: 263MB Available: 5381MB
Total space required includes space for required products.

Figure 17-3 Installing WebSphere Application Server - Express

Be patient. It may take a few minutes. Do not touch the mouse or the keyboard!

15. Read the client Readme file and click **Next**.
16. Read the server Readme file and click **Next**.
17. Select **Yes, restart my system**, and click **Next**.
18. Click **Finish** to complete the installation and re-boot the system.
19. When the machine is booted up, the Directory Server Configuration Tool is started automatically.

Before continuing with the rest of this section, check the following:

- IBM Directory Admin Daemon service is running in the Windows Control Panel.
- IBM Directory Server V5.1 service is running in the Windows Control Panel.
- IBM HTTP Server 1.3.26 service is running in the Windows Control Panel.

17.2.2 Creating administrator DN and password

Complete the following steps to create an administrator DN and password:

1. In the Directory Server Configuration Tool window, select **Administrator DN/password** in the Navigation pane on the left.
2. Set the values of the fields in Table 17-1.

Table 17-1 Creating administrator DN and password

Field	Value
Administrator DN	cn=root
Administrator password	password
Confirm password	password

3. Click **OK**.
4. Click **OK** in confirmation window.

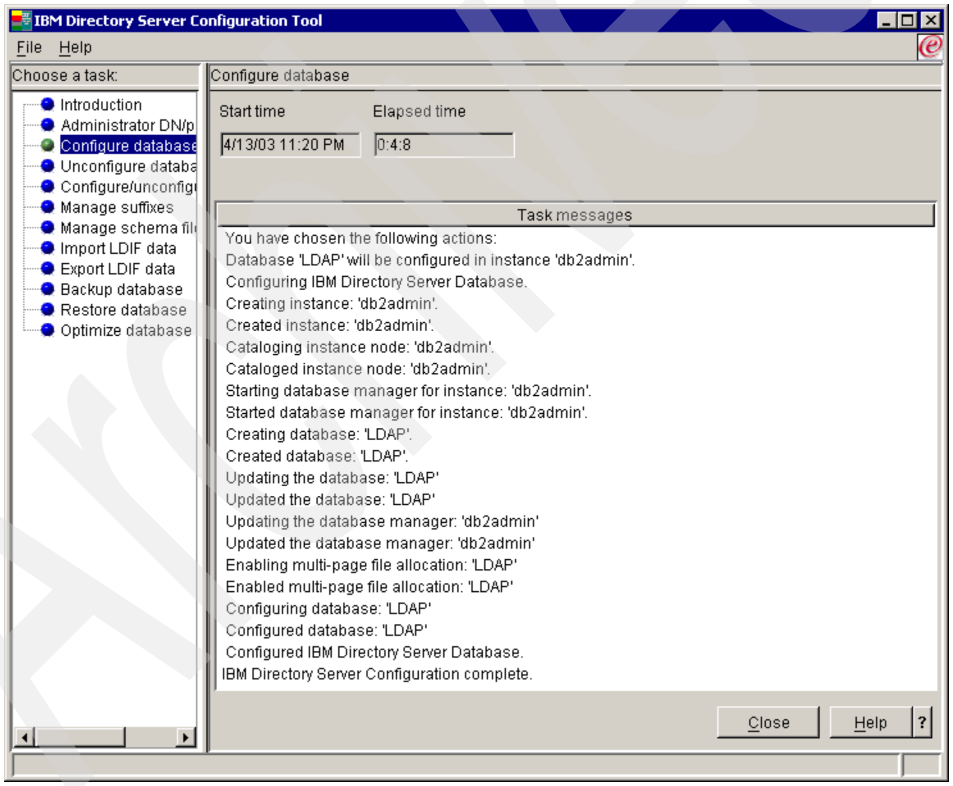
17.2.3 Configuring database

To configure Directory Server database, complete the following steps:

1. In the Directory Server Configuration Tool window, select **Configure database** in the Navigation pane on the left.
2. Select **Create a new database** and click **Next**.
3. Enter the database owner user ID and password. This user ID should have been created on the system and it should be part of the DBA administrator

group. By default, the DBA administrator group is the same as the system administrator group on the Windows system.

4. Click **Next**.
5. Enter `LDAP` in the Database name field for our scenario.
6. Click **Next**.
7. Select a code page and click **Next**.
8. Enter a database location and click **Next**.
9. Review the database configuration option and click **Finish** to start configuration.
10. Review the configuration log as shown in Figure 17-4.
11. Click **Close**.



4/13/03 11:20 PM
0:4:8

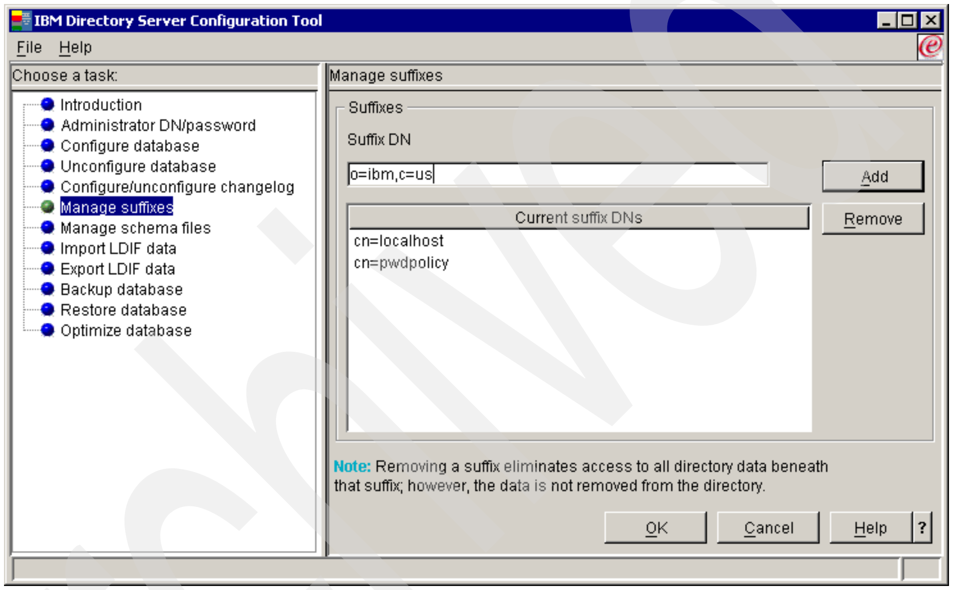
```
You have chosen the following actions:
Database 'LDAP' will be configured in instance 'db2admin'.
Configuring IBM Directory Server Database.
Creating instance: 'db2admin'.
Created instance: 'db2admin'.
Cataloging instance node: 'db2admin'.
Cataloged instance node: 'db2admin'.
Starting database manager for instance: 'db2admin'.
Started database manager for instance: 'db2admin'.
Creating database: 'LDAP'.
Created database: 'LDAP'.
Updating the database: 'LDAP'
Updated the database: 'LDAP'
Updating the database manager: 'db2admin'
Updated the database manager: 'db2admin'
Enabling multi-page file allocation: 'LDAP'
Enabled multi-page file allocation: 'LDAP'
Configuring database: 'LDAP'
Configured database: 'LDAP'
Configured IBM Directory Server Database.
IBM Directory Server Configuration complete.
```

Figure 17-4 Directory Server database configuration

17.2.4 Creating a suffix

Complete the following steps to create a new suffix:

1. Launch the IBM Directory Server Configuration Tool by selecting **Start -> Programs -> IBM Directory Server 5.1 -> Directory Configuration**.
2. Select **Manage suffixes** in the Navigation pane on the left.
3. Enter `o=ibm,c=us` in the Suffix DN field. This sets the organizational unit to IBM and the country code to US. See Figure 17-5.



Suffix DN: o=ibm,c=us

Current suffix DNs:
cn=localhost
cn=pwdpolicy

Note: Removing a suffix eliminates access to all directory data beneath that suffix; however, the data is not removed from the directory.

Figure 17-5 Create new suffix

4. Click **Add** to add it to the Currents suffix DN list.
5. Click **OK** to save it.
6. Exit the IBM Directory Server Configuration Tool application.

17.2.5 Registering the LDAP server

Complete the following steps to register the server in Directory Server V5.1:

1. Start the LDAP application server:
   a. Open a command window.
   b. Run the command **`cd c:\LDAP\appsrv\bin`**.

   c. Run the command **`serverStatus -all`**. This command returns the running status of all application servers.
   d. If server1 is not running, enter the command **`startServer.bat server1`** to start it.
2. Log on to the LDAP Web Administration Tool:
   a. Open a browser.
   b. Enter `http://<LDAPhostname>:9080/IDSWebApp/IDSjsp/Login.jsp`. This brings up the IBM Directory Server Web Administration logon window. For information about using the Web Administration Tool, refer to the *IBM Directory Server Version 5.1 Administration Guide*, found at:

      `http://publib.boulder.ibm.com/tividd/td/IBMDirectoryServer5.1.html`
3. Registering the LDAP server:
   a. At this time, you only have Console Admin entry in LDAP Hostname field on the logon window. You need to register your LDAP server in LDAP Web Administration Tool.
   b. On the logon window, enter the values of the fields in Table 17-2.

Table 17-2 Logon to LDAP Web Administration Tool

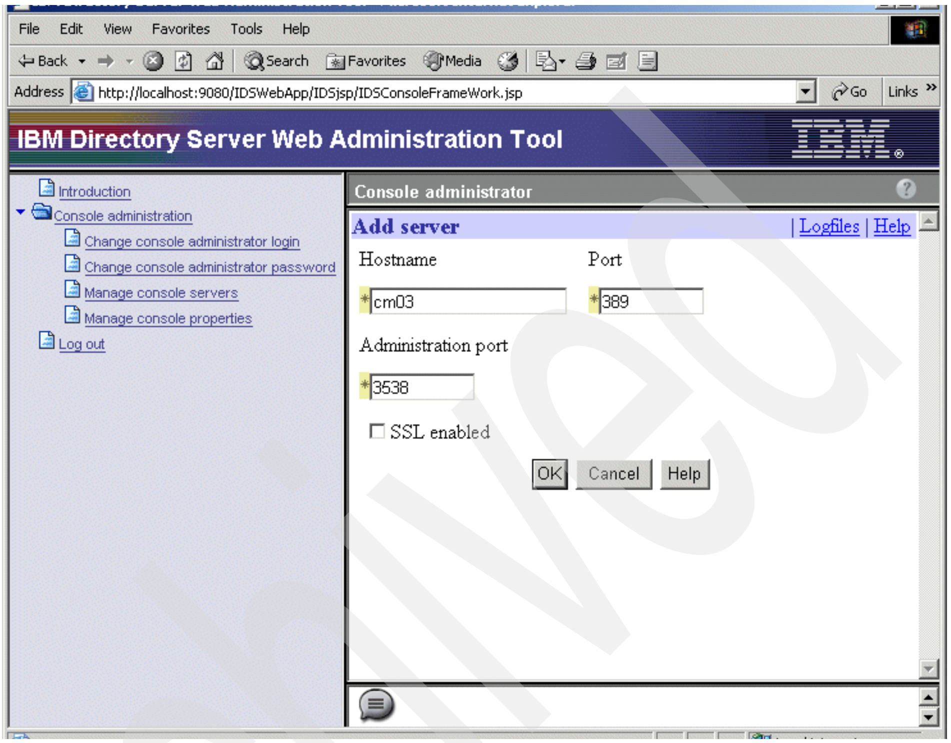
Field	Value
LDAP Hostname	Console Admin
Username	superadmin
Password	secret

   c. Click **Login**.
   d. Select **Console administration -> Manage console servers** in Navigation pane on the left.
   e. Click **Add** in the Manage console servers window.
   f. Enter the values of the fields in Table 17-3 for our scenario. See Figure 17-6 on page 406.

Table 17-3 Add LDAP server

Field	Value
Hostname	cm03
Port	389
Administration port	3538
SSL enabled	False

In the Hostname field, enter the LDAP server host name. For demonstration purposes in our scenario, we entered `cm03`.


File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:9080/IDSWebApp/IDSjsp/IDSConsoleFrameWork.jsp
Go Links
IBM Directory Server Web Administration Tool
IBM
Introduction
Console administration
Change console administrator login
Change console administrator password
Manage console servers
Manage console properties
Log out
Console administrator
Add server
| Logfiles | Help
OK Cancel Help

Hostname: cm03
Port: 389
Administration port: 3538
☐ SSL enabled

Figure 17-6 Add LDAP server

g. Click **OK** to save the change.

h. Click **Log out** in the Navigation pane on the left.

17.2.6 Creating LDAP user

Complete the following steps to create users in Directory Server:

1. Log on to the LDAP Web Administration Tool:

   a. Open a browser.

   b. Enter `http://<LDAPhostname>:9080/IDSWebApp/IDSjsp/Login.jsp`.

   c. On the logon window, enter the values of the fields in Table 17-4 on page 407.

Table 17-4 Logon values for the LDAP Web Administration Tool

Field	Value
LDAP Hostname	cm03
Username	cn=root
Password	password

d. Click **Login**.

2. Create the LDAP user wasadmin:

   a. In the Navigation pane on the left, click **Directory management -> Manage entries**.

   b. In the Manage entries window on the right, select entry **o=ibm,c=us**.

   c. Click **Add** in the Manage entries window.

   d. Select **inetOrgPerson** in the Structural object class field on the Select object class window.

   e. Click **Next**.

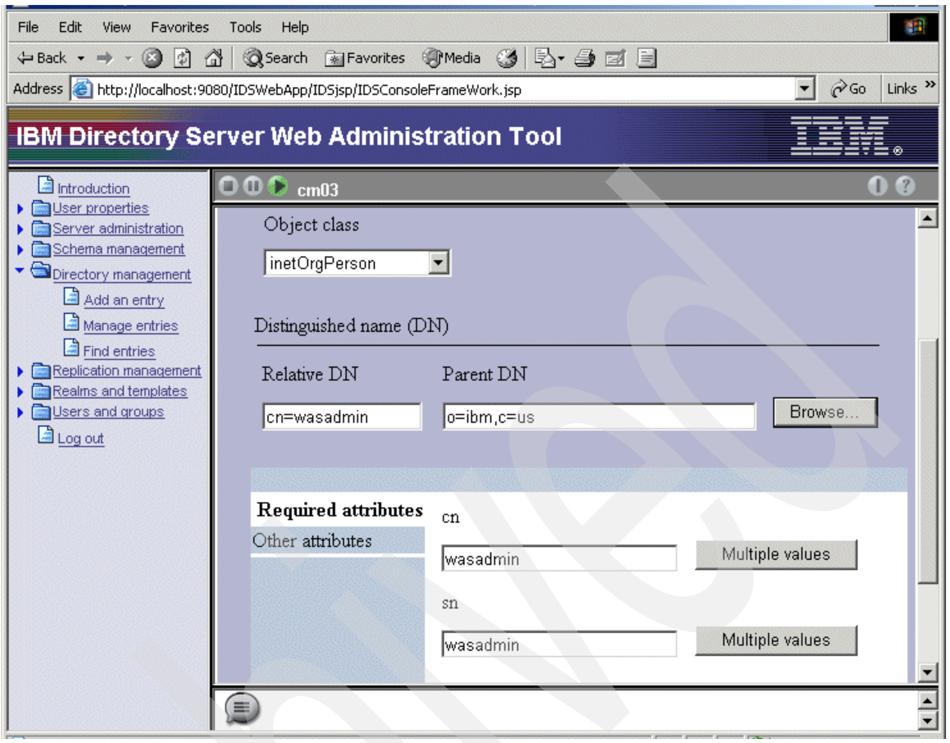
   f. Click **Next** on the Select auxiliary object classes window.

   g. Set the values for the fields in Table 17-5 for our scenario. See Figure 17-7 on page 408.

Table 17-5 Set the required fields for user wasadmin

Field	Value
Relative DN	cn=wasadmin
cn	wasadmin
sn	wasadmin

For our scenario, the user cn=wasadmin,o=ibm,c=us is not a regular user. It will be used to connect to IBM Directory Server while you configure security in the WebSphere Application Server.



Object class: inetOrgPerson

Distinguished name (DN)

Relative DN: cn=wasadmin
Parent DN: o=ibm,c=us

Required attributes
Other attributes

cn: wasadmin

sn: wasadmin

Figure 17-7 Set required fields for LDAP user

h. Select the **Other attributes** tab.

i. Set the values for the fields in Table 17-6.

Table 17-6 Set the optional fields for user wasadmin

Field	Value
uid	wasadmin
userPassword	password

j. Click **Finish** to save the user wasadmin.

3. Create an organizational unit:

   a. In the Navigation pane on the left, click **Directory management -> Add an entry**.

   b. In the Select object class window on the right, select the entry **organizationalUnit**. Click **Next**.

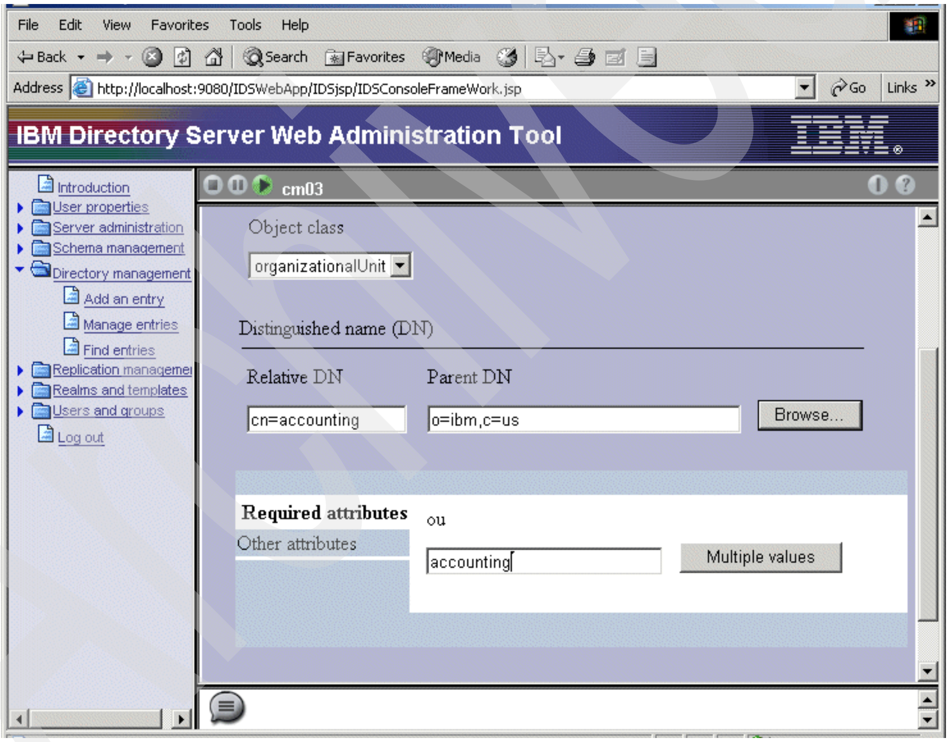
   c. Click **Next** on the Select auxiliary object classes window.

d. Set the values for the fields in Table 17-7 for our scenario. See Figure 17-8.

Table 17-7 Set the required fields for organizational unit

Field	Value
Relative DN	ou=accounting
Parent DN	o=ibm,c=us
ou	accounting

We are creating a tree structure for the accounting department in IBM Directory Server. This helps better manage user information in the LDAP server.



Object class: organizationalUnit

Distinguished name (DN)

Relative DN: cn=accounting

Parent DN: o=ibm,c=us

Required attributes

Other attributes

ou: accounting

Figure 17-8 Add organizational unit

e. Click **Finish** to add the entry.

4. Adding an LDAP user in the accounting department:

   a. In the Navigation pane on the left, click **Directory management -> Add an entry**.

b. In the Select object class window on the right, select the entry **inetOrgPerson**.
c. Click **Next**.
d. Click **Next** on the Select auxiliary object classes window.
e. Set the values for the fields in Table 17-8 for our scenario.

Table 17-8 Set the required fields for user John Smith

Field	Value
Relative DN	cn=john
Parent DN	ou=accounting,o=ibm,c=us
cn	john
sn	smith

This adds the user John Smith in the accounting department.

f. Select the **Other attributes** tab.
g. Set the values for the fields in Table 17-9.

Table 17-9 Set the optional fields for user wasadmin

Field	Value
uid	john
userPassword	password

h. Click **Finish** to save the user wasadmin.
i. Exit the Directory Server Web Administration Tool by clicking **Log out** in the Navigation pane on the left.

17.3 Configuring Content Manager V8.2 for LDAP

Single sign-on requires synchronized user profiles between different applications. In this chapter, you use IBM Directory Server V5.1 to manage a single centralized user repository. Before you import users from the LDAP server to Content Manager, you must enable the LDAP feature in the Content Manager Library Server.

In this section, we show how to configure the Content Manager V8.2 for LDAP on a Windows platform. Before continuing with the rest of this section, make sure that IBM Directory Server V5.1 service is running in the Windows Control Panel.

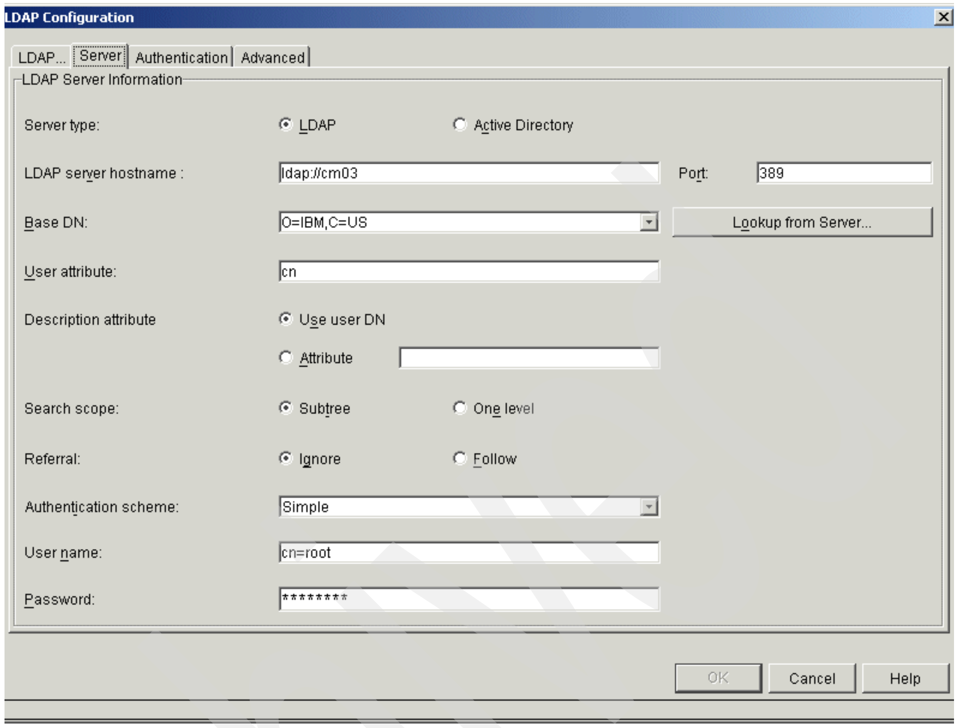
17.3.1 Generating the properties file

Complete the following steps to generate the properties file:

1. Launch a Content Manager System Administration Client by selecting **Start -> Programs -> IBM Content Manager for Multiplatforms V8.2 -> System Administration Client**.
2. Log on by entering `icmadmin` and `password` in the appropriate fields.
3. Select **Tools -> LDAP Configuration** from the menu bar.
4. Select **Enable LDAP User import and authentication** on the LDAP Information tab.
5. Select the **LDAP Server Information** tab.
6. Set the value of the fields in Table 17-10. See Figure 17-9 on page 412.

Table 17-10 Configure LDAP server information in Content Manager

Field	Value
Server Type	LDAP
LDAP server hostname	cm03 for our scenario (host name of LDAP server)
Port	389
Base DN	o=ibm,c=us (click **Lookup from Server** and select from the list)
User Attribute	cn
Description Attribute	Use user DN
Search Scope	Select Subtree
Referral	Ignore
Authentication Schema	simple
User name	cn=root
Password	password


LDAP Configuration

LDAP... | Server | Authentication | Advanced

LDAP Server Information

Server type:	[✓ LDAP]	Active Directory		
LDAP server hostname :	ldap://cm03		Port:	389
Base DN:	O=IBM,C=US		Lookup from Server...	
User attribute:	cn			
Description attribute	[✓ Use user DN]			
	Attribute			
Search scope:	[✓ Subtree]	One level		
Referral:	[✓ Ignore]	Follow		
Authentication scheme:	Simple			
User name:	cn=root			
Password:	********			

Figure 17-9 Configure LDAP server information in Content Manager

7. Click **OK** to save the configuration.

When the configuration is completed and saved, the cmbcmenv.properties file is generated in the directory pointed to by the CMCOMMON environment variable. By default, this is in C:\Program Files\IBM\Cmgmt. Example 17-1 shows a sample cmbcmenv.properties file.

Example 17-1 Sample cmbcmenv.properties file

```
CMCFGDIR=C:\\Program Files\\IBM\\Cmgmt
CMCOMMON_LDAP=enabled
LDAP_DATASOURCES=disabled
LDAP_USER_AUTHENTICATION=enabled
LDAP_INITIAL_CONTEXT_FACTORY=com.sun.jndi.ldap.LdapCtxFactory
LDAP_SERVER_TYPE=STANDARD_LDAP
LDAP_PROVIDER_URL=ldap://cm03
LDAP_REFERRAL=ignore
LDAP_SECURITY_AUTHENTICATION=Simple
LDAP_SECURITY_PRINCIPAL=cn=root
LDAP_SECURITY_CREDENTIALS=(MDoPbWQ9IVQDERMR)
LDAP_ROOT_DN=O=IBM,C=US
```

```
LDAP_SEARCH_SCOPE=SUBTREE_SCOPE
LDAP_AUTHENTICATION_ATTRIBUTE=uid
LDAP_SECURITY_PROTOCOL=none
LDAP_PORT=389
LDAP_DESC_ATTR=DN
LDAP_IBM_SSL_KEYRING=none
LDAP_IBM_SSL_PASSWORD=none
LDAP_IBM_SSL_CIPHERS=SSL_RSA_EXPORT_WITH_RC4_40_MD5
SSL_RSA_EXPORT_WITH_DES40_CBC_SHA SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
LDAP_MAX_RECORDS=5000
LDAP_SERVER_TIMEOUT=15
```

17.3.2 Installing properties file

The file cmbcmenv.properties is used by the Content Manager System Administration Client to import users from the LDAP server (the IBM Directory Server). Because the Library Server and the Resource Manager also use the information stored in the file for user authentication, the file must be copied to both the Library Server and the Resource Manager.

Installing the properties file on the Library Server

This step is only necessary if the LDAP integration was set up using a remote Content Manager System Administration Client. Otherwise, go straight to “Installing the properties file on the Resource Manager” on page 413.

To install the properties file on the Library Server, copy the cmbcmenv.properties file generated in “Generating the properties file” on page 411 to the CMCOMMON directory on the Library Server machine. By default, this is C:\Program Files\IBM\Cmgmt.

Installing the properties file on the Resource Manager

Complete the following steps to install the properties file on the Resource Manager:

1. Copy the cmbcmenv.properties file generated in “Generating the properties file” on page 411 to the Resource Manager application server directory:

   ```
   C:\WebSphere\AppServer\installedApps\cm03\icmrm.ear\icmrm.war\WEBINF\classe
   s\com\ibm\mm\icmrm
   ```

   where `icmrm` is the default Resource Manager application server name and `cm03` is the host name where the WebSphere Application Server is installed.
2. Open the cmbcmenv.properties file that resides in the Resource Manager application server directory.

3. Replace all the encrypted passwords to plain text passwords. When you restart the Resource Manager application server, the passwords will be re-encrypted.
4. Restart the Resource Manager application server.
5. Repeat the above steps for each Resource Manager if there are multiple ones.

17.3.3 Installing user exit

The user exit file ICMXLSLG.DLL must be installed to enable LDAP integration in the Content Manager server. The file is located in the LDAP directory of the Content Manager installation directory. By default, this directory is C:\ICMROOT\LDAP.

The user exit file ICMXLSLG.DLL must be copied into the Library Server DLL directory of the Content Manager installation directory. By default, this directory is C:\ICMROOT\ICMNLSDB\DLL.

17.4 Configuring Content Manager V8.2 for SSO

In order to have the single sign-on feature working in the Content Manager server, three things need to happen:

- Enable the single sign-on feature in the Content Manager server.
- Create a new privilege set, which must contain the following two privileges and any other necessary privileges:
  - AllowConnectToLogon, which allows the Content Manager user to log on with a different DB2 connection user.
  - AllowTrustedLogon, which allows the Content Manager user to log on with a different DB2 connection user and without a password.
- User IDs that are used to log on to the Content Manager server with single sign-on must use the newly created privilege set.

17.4.1 Enabling single sign-on

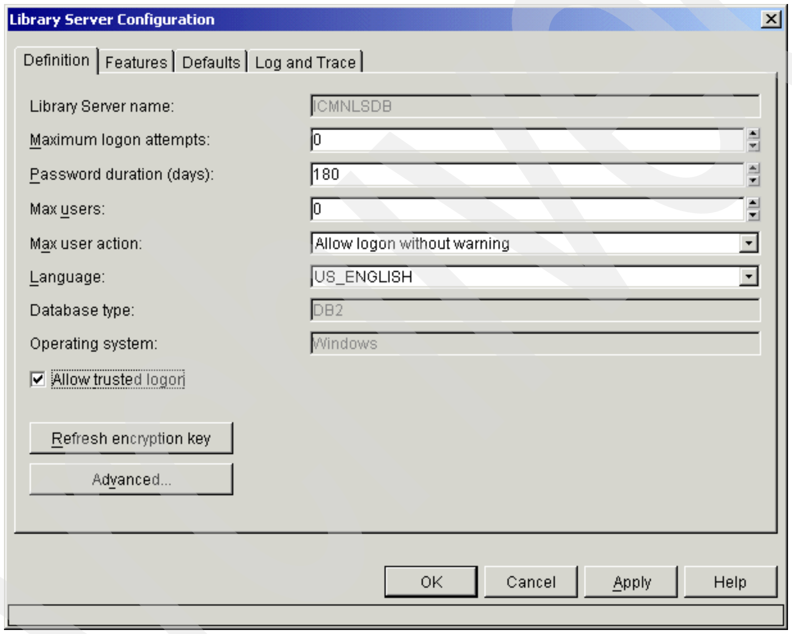
You can enable Content Manager single sign-on feature during installation or after installation. In the following, we show how to turn on this feature after installation:

1. Launch a Content Manager System Administration Client by selecting **Start -> Programs -> IBM Content Manager for Multiplatforms V8.2 -> System Administration Client**.

2. In the Navigation pane on the left, select **Library server parameters -> Configuration**.
3. The Library Server configuration is displayed in the Contents of configuration pane on the right. Right-click it and select **Properties**.
4. Set the values of the fields in Table 17-11. See Figure 17-10.

Table 17-11 Enable single sign-on in Library Server

Field	Value
Max user action	Allow logon without warning
Allow trusted logon	True

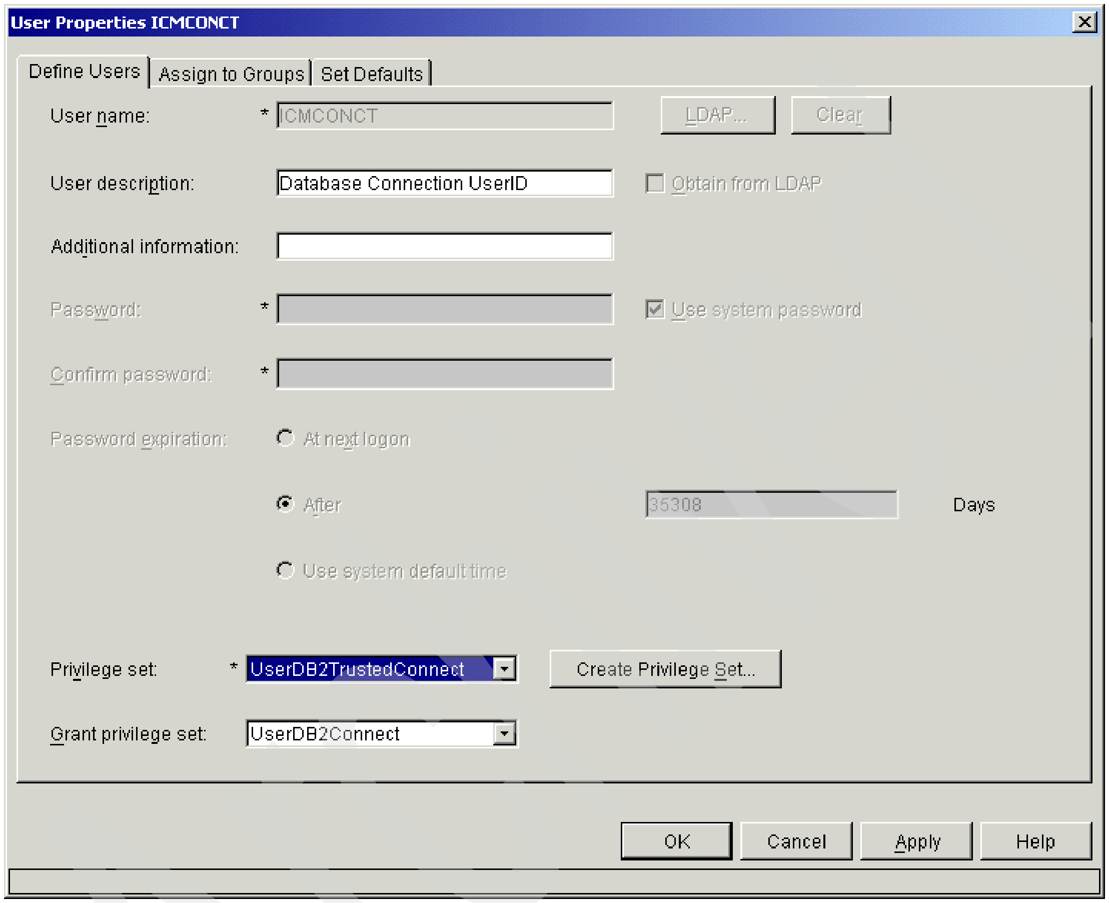


Library Server name: ICMNLSDB
Maximum logon attempts: 0
Password duration (days): 180
Max users: 0
Max user action: Allow logon without warning
Language: US_ENGLISH
Database type: DB2
Operating system: Windows
[✓] Allow trusted logon

Figure 17-10 Enable single sign-on in Library Server

5. Click **OK** to save the change.
6. In the Navigation pane on the left, select **Authentication -> Users**. A list of users is displayed in the Contents of users pane on the right. One of the users is the connection user. By default, it is icmconct.
7. Right-click user **icmconct** and select **Properties**. The User properties window is displayed with user icmconct's definition.
8. Change the Privilege set field to UserDB2TrustedConnect. See Figure 17-11 on page 416. The privilege set UserDB2TrustedConnects allows Content

Management users to connect to the DB2 database without having their own DB2 user IDs. These users are also not required to have a password in the Content Manager.


User Properties ICMCONCT

Define Users | Assign to Groups | Set Defaults

User name: * ICMCONCT

User description: Database Connection UserID

Additional information:

Password: *

Confirm password: *

[✓ Use system password]

Obtain from LDAP

Password expiration: At next logon [✓ After] 35308 Days; Use system default time

Privilege set: * UserDB2TrustedConnect

Grant privilege set: UserDB2Connect

Figure 17-11 Grant UserDB2TrustedConnect privilege set to user icmconct

9. Click **OK** to save the change.

17.4.2 Creating new privilege set

Complete the following steps to create a new privilege set:

1. Launch a Content Manager System Administration Client by selecting **Start -> Programs -> IBM Content Manager for Multiplatforms V8.2 -> System Administration Client**.
2. In the Navigation pane on the left, select **Authorization -> Privilege Set**.

3. Right-click **Privilege Set** and select **New**. The New privilege Set Definition window is displayed.
4. Enter `ClientSSO` in the Name field for our scenario.
5. Select the **ClientTaskLogon** entry in the Privilege groups field.
6. Check the **Select all** box at the top of the Privileges field. Both AllowConnectToLogon and AllowTrustedLogon entries are then included in the Selected privileges field. See Figure 17-12.

Important: Steps 5 and 6 must be done for every privilege set used for single sign-on.

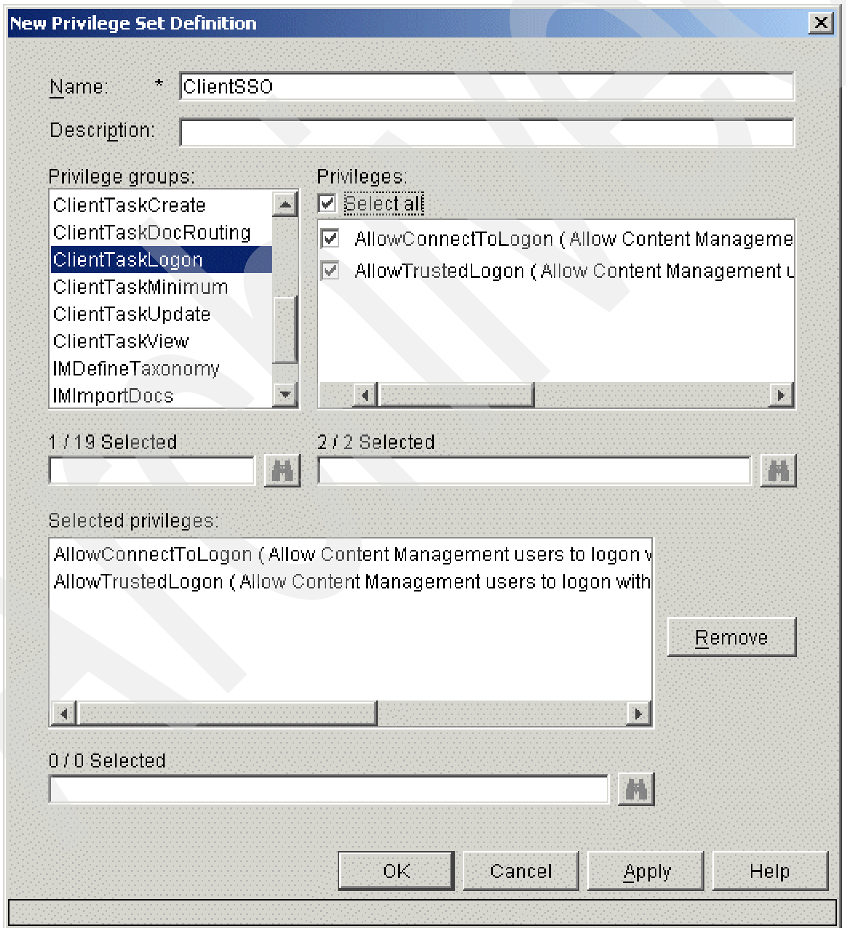


Figure 17-12 Adding privileges to privilege set

7. Select the **ClientTaskMinimum** entry in the Privilege groups field.

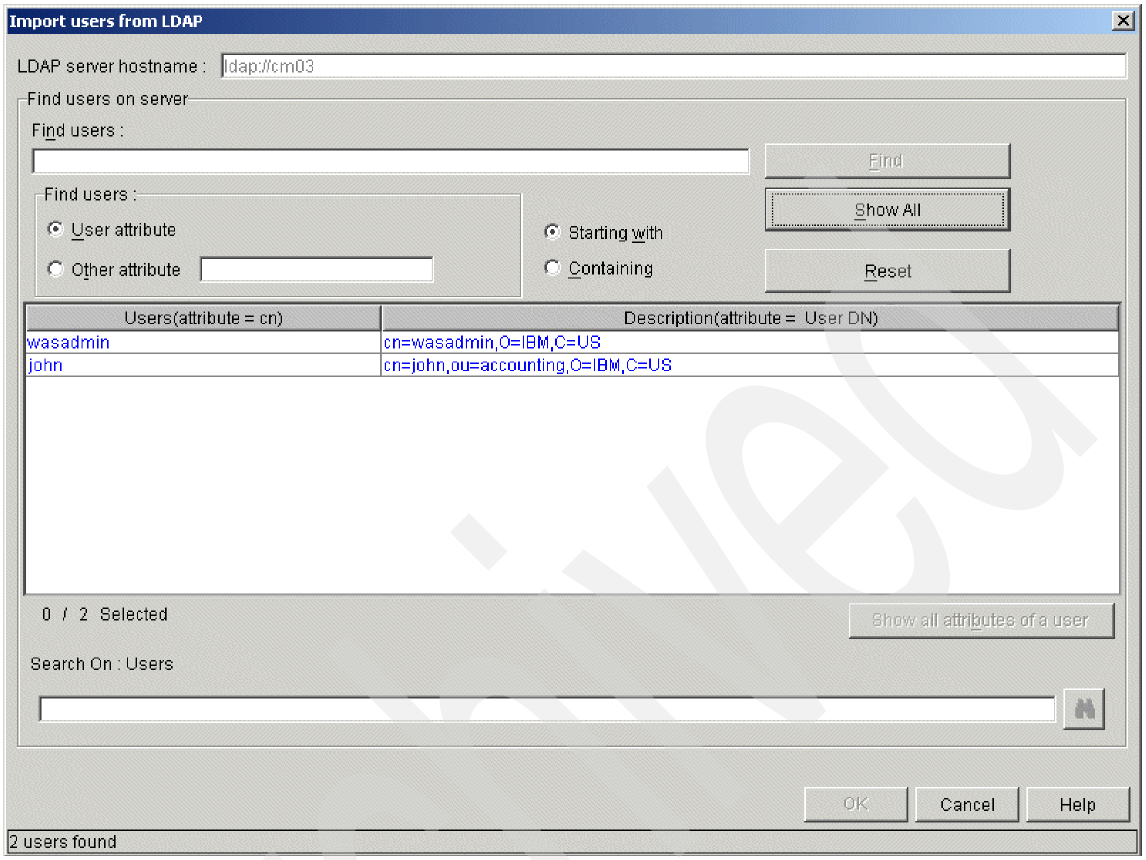
   The privilege group ClientTaskMinimum is selected for demonstration purposes. When you configure your production system, you need to select privilege groups based on your business requirements.

8. Check the **Select all** box at the top of Privileges field. This makes all entries in the privilege group appear in the Selected privileges field.
9. Click **OK** to save the privilege set.

17.5 Importing LDAP users into Content Manager V8.2

This section provides the steps to import LDAP users into the Content Manager server V8.2 through Content Manager System Administration Client on the Windows platform. Before continuing the rest of this section, make sure that IBM Directory Server V5.1 service is running in the Windows Control Panel.

1. Launch a Content Manager System Administration Client by selecting **Start -> Programs -> IBM Content Manager for Multiplatforms V8.2 -> System Administration Client**.
2. In the Navigation pane on the left, select **Authentication -> Users**.
3. Right-click **Users** and select **New**. The New User window is displayed.
4. Click **LDAP** in the Define Users tab to import a user.
5. Click **Show All** on the Import users from LDAP window. If you have defined many users in LDAP server, you should enter search criteria and click **Find**, instead.
6. A warning message is displayed: "`Max results retrieved limited to 5000. Continue retrieving?`" Click **Yes**.
7. You should have at least two users (wasadmin and john*)* on the user list. See Figure 17-13 on page 419.



LDAP server hostname : ldap://cm03

Find users :

Find users : [✓ User attribute] Other attribute

[✓ Starting with] Containing

Users(attribute = cn)	Description(attribute = User DN)
wasadmin	cn=wasadmin,O=IBM,C=US
john	cn=john,ou=accounting,O=IBM,C=US

0 / 2 Selected

Search On : Users

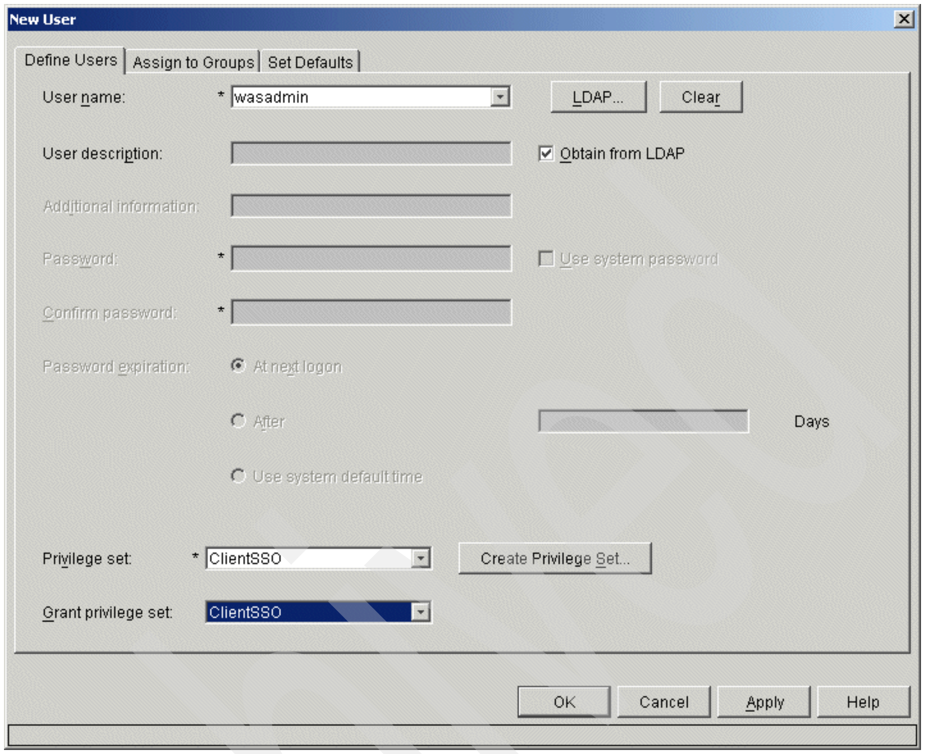
2 users found

Figure 17-13 Import LDAP users

8. Highlight user wasadmin on the user list.
9. Click **OK** to import the user. Notice that the user information and password options have been grayed out on the New User window. The information for the LDAP server is used.
10. Set the values for the fields in Table 17-12. See Figure 17-14 on page 420.

Table 17-12 Creating new Content Manager user

Field	Value
Privilege set	ClientSSO
Grant privilege set	ClientSSO



User name: * wasadmin

User description:

[✓] Obtain from LDAP

Additional information:

Password: *

[ ] Use system password

Confirm password: *

Password expiration: [● At next logon] ○ After ____ Days ○ Use system default time

Privilege set: * ClientSSO

Grant privilege set: ClientSSO

Figure 17-14 Creating new Content Manager user

11. Click **OK** to save the new user definition wasadmin.
12. Repeat the above steps to import an LDAP user named john and other LDAP users. LDAP user john must be imported because it will be used in a later section.
13. This should be the end of the single sign-on configuration in the Content Manager. However, due to a problem in the Version 8.2 product at the time of writing, you have to do one more step:
    a. Open a DB2 command window by selecting **Start -> Programs -> IBM DB2 -> Command Line Tools -> Command Window**.
    b. Enter the following DB2 SQL statement to connect to the Library Server:

    ```
    db2 connect to icmnlsdb user icmadmin using password
    ```

    c. Enter the following DB2 SQL statement

    ```
    db2 update ICMSTUSERS set PASSWORD='XYZ' where PASSWORD is NULL
    ```

    where `'XYZ'` is arbitrary. This statement populates the PASSWORD column in the table ICMSTUSERS in the Library Server database.

17.6 Configuring WebSphere Application Server V5.0

This section discusses the Security administration, LDAP User Registry configuration, and Lightweight Third Party Authentication (LTPA) in WebSphere Application Server.

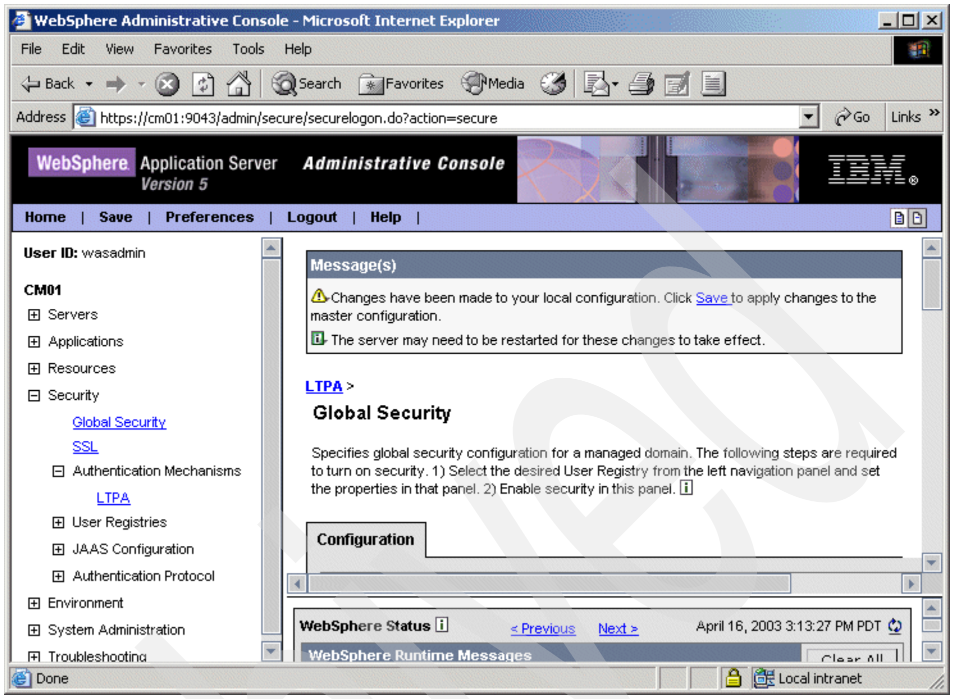
Before continuing the rest of this section, make sure that:

- IBM Directory Server V5.1 service is running in the Windows Control Panel.
- IBM WebSphere Application Server V5 - server1 service is running in the Windows Control Panel.

17.6.1 Configuring LTPA

Lightweight Third Party Authentication (LTPA) is intended for distributed, multiple application server and machine environments. It supports forwardable credentials, and therefore supports Single Sign-On. LTPA requires a configured User Registry to be a central shared repository. It could be LDAP user registry, a Windows Domain type user registry, or a custom user registry. LDAP server is used for the discussion in this chapter.

1. Launch WebSphere Administrative Console by selecting **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Administrative Console**.
2. Enter any user ID to log on to WebSphere Administrative Console since you have not enabled WebSphere security. The sole purpose to enter a user name on the logon window is for logging.
3. In the Navigation pane on the left, click **Security -> Authentication Mechanisms -> LTPA**.
4. Enter the password twice. This password is the password to protect LTPA keys. You will need this password in order to import the keys into any other SSO-enabled server.
5. Click **OK** to make the changes effective.
6. Click **Save** in the Messages box at the top of the window. See Figure 17-15 on page 422.


WebSphere Administrative Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address https://cm01:9043/admin/secure/securelogon.do?action=secure
Go Links
WebSphere Application Server Version 5 Administrative Console
IBM
Home | Save | Preferences | Logout | Help |
User ID: wasadmin
CM01
Servers
Applications
Resources
Security
Global Security
SSL
Authentication Mechanisms
LTPA
User Registries
JAAS Configuration
Authentication Protocol
Environment
System Administration
Troubleshooting
Configuration
WebSphere Status
< Previous Next >
April 16, 2003 3:13:27 PM PDT
WebSphere Runtime Messages
Clear All
Done
Local intranet

Message(s)

Changes have been made to your local configuration. Click Save to apply changes to the master configuration.

The server may need to be restarted for these changes to take effect.

LTPA >

Global Security

Specifies global security configuration for a managed domain. The following steps are required to turn on security. 1) Select the desired User Registry from the left navigation panel and set the properties in that panel. 2) Enable security in this panel.

Figure 17-15 Save step #1 in WebSphere

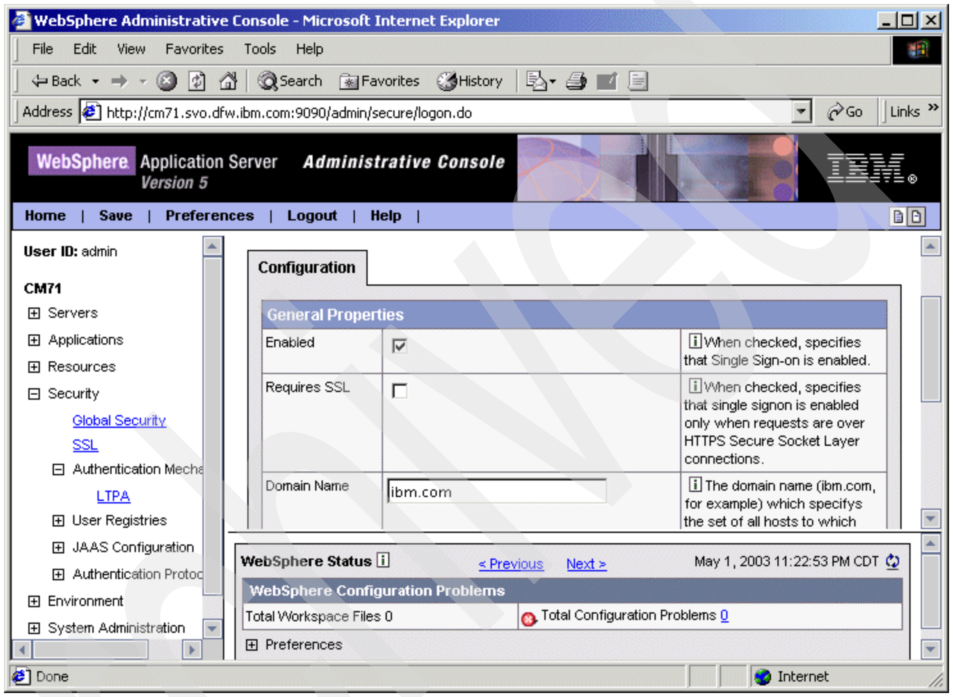
7. Click **Save** on the Save to Master Configuration window. Your modification made in WebSphere Administrative Console has not been saved to the configuration repository until now.
8. In the Navigation pane on the left, click **Security -> Authentication Mechanisms -> LTPA**.
9. Click **Single Signon (SSO)** at the bottom of the LTPA pane on the right. The Single sign-on window appears.
10. Set the values for the fields in Table 17-13. See Figure 17-16 on page 423.

Table 17-13 Configuring WebSphere single sign-on

Field	Value
Enabled	True
Requires SSL	False
Domain Name	ibm.com

The domain name (`ibm.com`, for example) specifies the set of all hosts to which single sign-on applies. If this field is not defined, the Web browser defaults the domain name to the host name where the Web application is running. This means single sign-on is restricted to that application server host name and does not work with other application server host names in the domain.

When SSL field is checked, it specifies that single sign-on is enabled only when requests are over HTTPS Secure Socket Layer connections.


WebSphere Administrative Console - Microsoft Internet Explorer

Address: http://cm71.svo.dfw.ibm.com:9090/admin/secure/logon.do

User ID: admin

CM71

Configuration

General Properties

Enabled	✓	When checked, specifies that Single Sign-on is enabled.
Requires SSL	☐	When checked, specifies that single signon is enabled only when requests are over HTTPS Secure Socket Layer connections.
Domain Name	ibm.com	The domain name (ibm.com, for example) which specifys the set of all hosts to which

WebSphere Status — May 1, 2003 11:22:53 PM CDT

WebSphere Configuration Problems

Total Workspace Files 0	Total Configuration Problems 0

Figure 17-16 Configuring WebSphere single sign-on

11.Click **OK** to make the change effective.

12.Click **Save** twice to save the change to WebSphere configuration repository.

17.6.2 Generating LTPA keys

Complete the following steps to generate LTPA keys:

1. In the Navigation pane on the left, click **Security -> Authentication Mechanisms -> LTPA**.
2. Click **Generate Keys**. This launches the key generation process in the background. You will be prompted to save the configuration after the process is completed.

3. Click **OK** to make the change effective.
4. Click **Save** twice to save the changes. The generated keys are stored in the security.xml file.
5. In the Navigation pane on the left, click **Security -> Authentication Mechanisms -> LTPA**.
6. In the Key File Name field, specify the name of the file where LTPA keys will be stored when you export them. You need to export the keys in order to enable single sign-on on another server. Specify the full path name for the key file, in our example c:\WebSphere\Appserver\etc\SSO_ltpakeys.
7. Click **Export Keys**. The key has been exported to the specified file. Example 17-2 shows a sample key file.

Example 17-2 Sample exported key file

```
#IBM WebSphere Application Server key file
#Thu May 01 23:40:40 CDT 2003
com.ibm.websphere.CreationDate=Thu May 01 23\:40\:40 CDT 2003
com.ibm.websphere.ltpa.version=1.0
com.ibm.websphere.ltpa.3DESKey=vQOPjmOXWk3YQFZe1lcgM+ON2gGrPWLbp7ji+BJPSDM\=
com.ibm.websphere.CreationHost=CM71
com.ibm.websphere.ltpa.PrivateKey=7JmY+QzBUThxt2FIJ7F+PKu7RLJcSEMjDkIs2jRp7KQIp
kEFNCCKq44mJ9GYFim/3yYKU8HP+j7EKFxsIXKWJGWcOpMIBMtrriQyKKgZHl8YyZQVR2zuqJO2C1Pr
uc5HeNcWNSKZ6oOovOwQXEGECMJCdPaY2IVkWfH1/3HqODYQGjr1hiUP5BgWO2c/UNva1XmxUJkg4Zz
zQEqSRfcg/zWPtH6NeUWPLZQ9REtKwam8hCd2xbm2+b4gfutJ9rcGiQg/uoQ8UfZyoIiR75nPclGsmY
DIrQHrO8Dt6F4u2VEFtyFF9ebiwZGtd5ZJvyz32K/3jpYJFKtuEWEBuYjSmhbB2JrVkE29Z8ObfObvN
oQ\=
com.ibm.websphere.ltpa.Realm=
com.ibm.websphere.ltpa.PublicKey=AKEPdnxl4dWUTgaWQhdDzucH2squnD52GDcvTtf7bZP4DX
JTkuvvoD63NGx3szOPJgEoOJLoUPIsw/UP7z7gABbW2fKmnvluiXytn7jCSOZMy4v2HvVycjUrUycG1
+wIbZ1zaSVWgZvx4vHWKmdmkbyJ55vg7uvIX3Yu5oSUPdvhAQAB
```

8. Click **Save** twice to save the change to the configuration repository.

17.6.3 Configuring LDAP user registries

To define WebSphere's LDAP user registries configuration, complete the following steps:

1. In the Navigation pane on the left, click **Security -> User Registries -> LDAP**. The LDAP user registries window is displayed on the right.
2. Set the values for the fields in Table 17-14 on page 425. See Figure 17-17 on page 426.

Table 17-14 Configuring LDAP user registries

Field	Value
Server User ID	cn=wasadmin,o=ibm,c=us
Server User Password	password
Type	IBM_Directory_Server
Host	cm03.ibm.com
Port	389
Base Distinguished Name (DN)	o=ibm,c=us
Bind Distinguished Name (DN)	cn=root
Bind Password	password
Search Timeout	take the default
Reuse Connection	True
Ignore Case	True (for IBM Directory Server 5.0)

The Server User ID field is the WebSphere administrator ID in LDAP server. We entered `cn=wasadmin,o=ibm,c=us` for the scenario.

For IBM Directory Server V5.1, specify IBM_Directory_Server as the Type. For the previous version of IBM Directory Server, use SecureWay® as the Type.

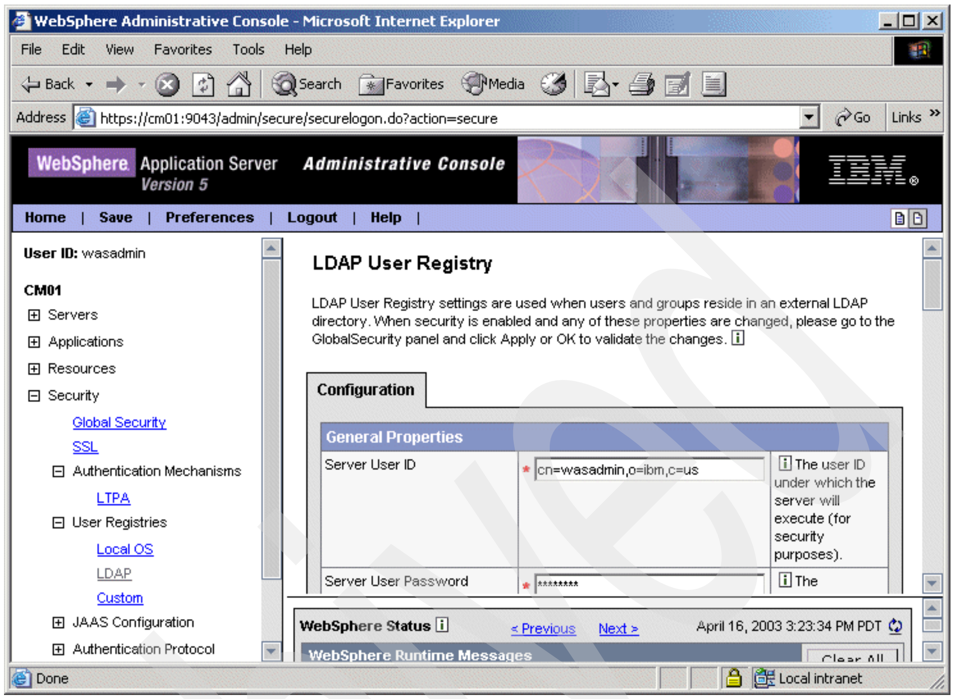
In the Host field, specify the host name of the LDAP server host name.

The default port number of the LDAP server is 389.

Specify the base DN of your LDAP configuration in the Base DN field. We used `o=ibm,c=us` for the scenario.

Bind DN field specifies the Distinguished Name for the application server to use to bind the LDAP server.

The Reuse Connection field should generally be selected. In rare situations, when you use a router to spray the requests to multiple LDAP servers and this router does not support affinity, we disable this check box.


WebSphere Administrative Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address https://cm01:9043/admin/secure/securelogon.do?action=secure
Go Links
WebSphere Application Server Version 5 Administrative Console
IBM
Home | Save | Preferences | Logout | Help |
User ID: wasadmin
CM01
Servers
Applications
Resources
Security
Global Security
SSL
Authentication Mechanisms
LTPA
User Registries
Local OS
LDAP
Custom
JAAS Configuration
Authentication Protocol
Configuration
WebSphere Status
< Previous Next >
April 16, 2003 3:23:34 PM PDT
WebSphere Runtime Messages
Clear All
Done
Local intranet

LDAP User Registry

LDAP User Registry settings are used when users and groups reside in an external LDAP directory. When security is enabled and any of these properties are changed, please go to the GlobalSecurity panel and click Apply or OK to validate the changes.

General Properties

Server User ID	* cn=wasadmin,o=ibm,c=us	The user ID under which the server will execute (for security purposes).
Server User Password	* ********	The

Figure 17-17 Define LDAP user registries

3. Click **OK** to make the change effective.
4. Click **Save** twice to save the change to the WebSphere configuration repository.

17.6.4 Enabling LTPA authentication

The following steps show you how to enable WebSphere Application Server security and use the LTPA authentication method for WebSphere Application Server.

1. In the Navigation pane on the left, click **Security -> Global Security**. The Global security window appears on the right.
2. Set the values for the fields in Table 17-15.

Table 17-15 Enabling security for WebSphere Application Server

Field	Value
Enabled	True

Field	Value
Enforce Java 2 Security	False
Active Protocol	CSI and SAS
Active Authentication Mechanism	LTPA
Active User Registry	LDAP

3. Click **OK** to make the change effective. WebSphere Application Server verifies the information defined in 17.6.3, “Configuring LDAP user registries” on page 424. If all information is valid, you are prompted to save the changes.
4. Click **Save** twice to save the change to the WebSphere configuration repository.
5. Log off the WebSphere Administrative Console.

17.6.5 Verifying WebSphere security configuration

You have enabled the security feature for WebSphere Application Server. Complete the following steps to stop and restart all running WebSphere Application Servers:

1. Open a command window on the WebSphere Application Server machine.
2. Run the command **`cd C:\WebSphere\AppServer\bin`**.
3. Run the command **`serverstatus -all`**. This command lists all available WebSphere servers and their running status. Example 17-3 shows an output of the command.

Example 17-3 serverStatus -all command

```
C:\WebSphere\AppServer\bin>serverStatus -all
ADMU0116I: Tool information is being logged in file
           C:\WebSphere\AppServer\logs\serverStatus.log
ADMU0500I: Retrieving server status for all servers
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: eClient_Server
ADMU0506I: Server name: icmrm
ADMU0506I: Server name: server1
ADMU0509I: The Application Server "eClient_Server" cannot be reached. It
           appears to be stopped.
ADMU0509I: The Application Server "icmrm" cannot be reached. It appears to be
           stopped.
ADMU0508I: The Application Server "server1" is STARTED
```

4. Stop all running servers. In our case, stop server1 by running the command **`stopServer server1`**.

If the command fails because of lack of permission (see Example 17-4), use the following command instead:

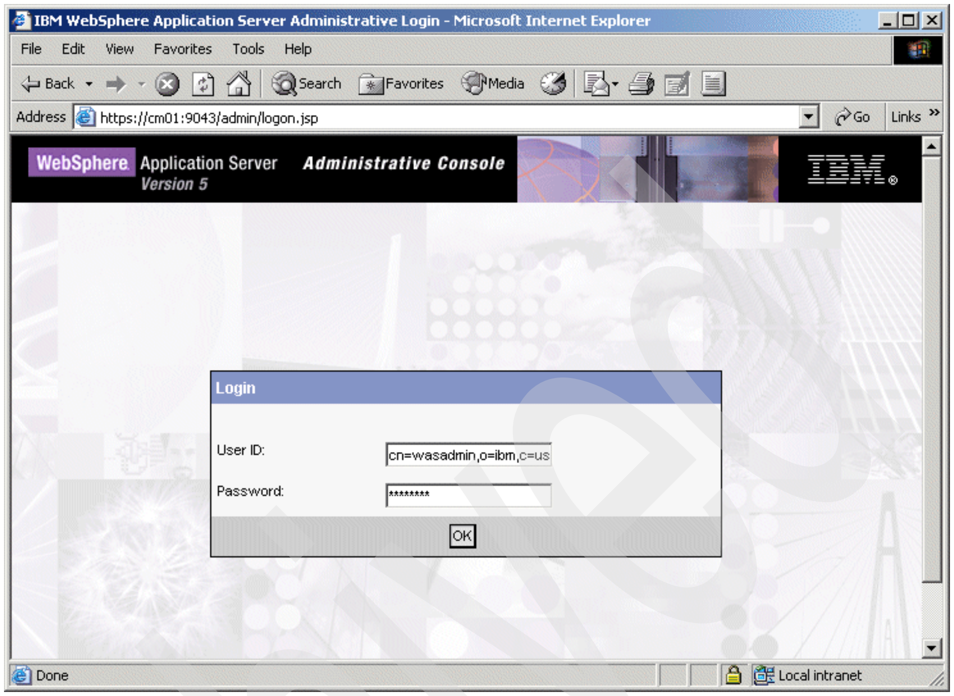
```
stopServer server1 -username cn=wasadmin,o=ibm,c=us -password password
```

Example 17-4 Failure of stopServer command

```
C:\WebSphere\AppServer\bin>stopServer server1
ADMU0116I: Tool information is being logged in file
           C:\WebSphere\AppServer\logs\server1\stopServer.log
ADMU3100I: Reading configuration for server: server1
ADMU0111E: Program exiting with error: javax.management.JMRuntimeException:
           ADMN0022E: Access denied for the stop operation on Server MBean due
           to insufficient or empty credentials.
ADMU0211I: Error details may be seen in the file:
           C:\WebSphere\AppServer\logs\server1\stopServer.log
ADMU1211I: To obtain a full trace of the failure, use the -trace option.
```

Important: You must submit the user name and password parameters to the **`stopServer`** command after WebSphere security is enabled.

5. Restart the desired servers. For now, you want to restart server server1 by running the command **`startServer server1`**.
6. Launch the WebSphere Administrative Console by selecting **Start -> Programs -> IBM WebSphere -> Application server v5.0 -> Administrative Console**.
7. The Administrative Console logon window is opened. It requires both user name and password. For the scenario, enter `cn=wasadmin,o=ibm,c=us` and `password` to log on. See Figure 17-18 on page 429.


IBM WebSphere Application Server Administrative Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address https://cm01:9043/admin/logon.jsp
Go Links
WebSphere Application Server Version 5 Administrative Console
IBM
Login
OK
Done
Local intranet

User ID: cn=wasadmin,o=ibm,c=us

Password: ********

Figure 17-18 Secured WebSphere administrative console logon window

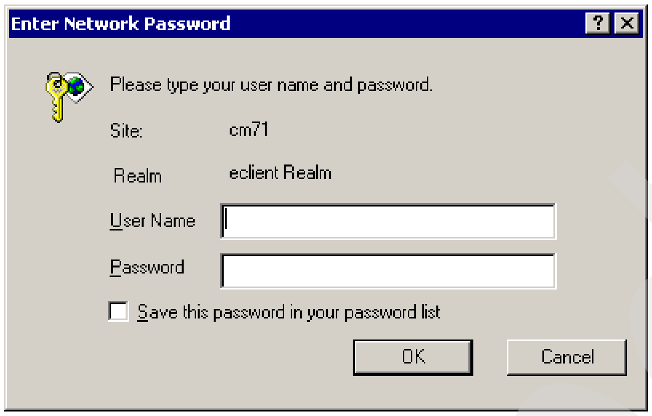
17.7 Verification

You have completed the configuration in IBM Directory Server Version 5.1, IBM WebSphere Application Server V5.0 and IBM Content Manager Server V8.2. Now, log on to eClient and see how it works differently.

1. Open a command window on the eClient server machine.
2. Run the command **`cd C:\WebSphere\AppServer\bin`**.
3. Run the command **`serverstatus -all`**. This command lists all available WebSphere servers and their running status.
4. If WebSphere server eClient_Server has not been started, enter the command **`startServer eClient_Server`** to start it.
5. Open a browser and enter the following URL to launch eClient:

   ```
   http://<eClientHostname>/eClient82/IDMInit
   ```

6. The following Network logon window is opened. See Figure 17-19 on page 430.


Enter Network Password

Please type your user name and password.

Site:	cm71
Realm	eclient Realm
User Name	
Password	

☐ Save this password in your password list

Figure 17-19 Network logon window

7. Enter the user name `cn=john,ou=accounting,o=ibm,c=us` and password `password`.
8. Click **OK** to log on to WebSphere Application Server. You have to enter the user name and password only once to log on to WebSphere Application Server if you have multiple applications with single sign-on enabled.
9. The eClient logon window is displayed. The user name has been taken from the user credentials in WebSphere Application Server and inserted in the User ID field. Note that the user ID and password fields are disabled. You are not allowed to enter data in these two fields.
10. Click **Logon** to log on to eClient.

If you also installed the WebSphere Portal Server, the logon attempt to the WebSphere Portal Server with the same user name at this time would not require entering a password again.

Part 5

Troubleshooting, debugging, and performance

In this part, we cover troubleshooting and debugging techniques. We include a list of typical problems users may face and their resolutions. In addition, we give a brief description of performance tuning for eClient.

18

Troubleshooting and debugging

In this chapter, we provide troubleshooting and debugging tips for IBM DB2 Information Integrator for Content Version 8 and IBM DB2 Content Manager Version 8 eClient.

This chapter covers the following specific topics:

- Isolating problems
- Tracing eClient
- Tracing EIP Java API
- Additional trace information
- Debugging your application
- Typical problems
- Support channels

18.1 Isolating problems

Content Manager eClient is a complex application. You have to install and configure many products before you are able to launch the logon window. For our scenario, we installed and configured the following products:

- WebSphere Application Server
- IBM HTTP Server
- EIP components
- eClient

If you want to successfully import and retrieve documents from a back-end server, such as the Content Manager server in our scenario, you have to install and configure these additional products:

- DB2 Universal database
- DB2 text Information Extender (if you want text search feature)
- WebSphere Application Server
- Content Manager (Library Server and Resource Manager)

In such a complex environment, it is extremely important to isolate the problem while you are doing troubleshooting. For example, if you have problems in the Web server but you are troubleshooting in the WebSphere Application Server, the problem will never be resolved. Before you can fix a problem, you have to identify which product or products cause the problem.

18.2 Tracing eClient

To troubleshoot eClient, we can configure the location, trace level and size for trace files by setting parameters in the IDM.properties file.

18.2.1 Configuring IDM.properties file

The WorkingDir parameter determines the location of the trace files. Set this parameter to the full path for the directory that you want to contain the trace files. For example, in a Windows environment, set the following parameter:

```
WorkingDir=C:\\CMeClient\\logs
```

The TraceLevel parameter controls the trace level. You can set the trace level to one of the following values in Table 18-1 on page 435.

Table 18-1 The valid TraceLevel value

TraceLevel	Description
0	Tracing off
1	Exceptions, and errors
2	Level 1 with the addition of general information, method entry, and method exit points
3	Level 2 with the addition of API calls
4	Level 3 with the addition of EIP non-visual bean tracing
5	Performance tracing

The trace file name is in the format eClientTrace_yyyy.mm.dd.log. For example, the trace file eClientTrace_2003.05.01.log is an eClient activity trace on May 1, 2003. For each day when there are activities on the eClient server, you have at least one log file. The log files are located in the directory defined in the WorkingDir parameter. If you set the trace level to 4, you should have additional EIP non-visual bean trace information in the eipBeanTrace.log file.

After you make changes in the IDM.properties file, you must stop and restart the eClient application server to make the changes effective.

18.2.2 Sample log file

In this section, we show you an eClient trace file. We set TraceLevel=1 in the IDM.properties file while the sample trace file is created.

To create the sample log file, we go through a simple login process and deliberately supply the wrong password information:

1. Stop the eClient_Server application server.
2. Delete every file in the C:\CMeClient\logs directory.
3. Start the eClient_Server application server.
4. Open a Web browser and enter the following URL to display the eClient logon window:

   ```
   http://localhost/eClient82/IDMInit
   ```

5. Enter `icmadmin` and `passwd` to log on. Note that the password is incorrect.
6. A message window pops up and indicates that an incorrect user ID, password or server name has been entered.

7. Open the eClient log file, which in our case is C:\CMeClient\logs\eClientTrace_2003.05.01.log. You should see the content of the sample log file shown in Example 18-1.

Example 18-1 Sample eClient log file

```
2003.05.12 14:43:32.016 com.ibm.idm.servlets.IDMConnection connect
ETT3DMQ1BXB1BYDOTOJDH5Y Servlet.Engine.Transports : 0
  com.ibm.mm.beans.CMBConnectFailedException: DGL0394A: Error in :
DKDatastoreICM.connect; [SERVER = ICMNLSDB, USERID = icmadmin]; ICM7172: The
password provided is invalid for this user ID, or it is NULL. (STATE) : [LS RC
= 7172]
```

In the sample log file, the Content Manager return code. is ICM7172. The message in the eClient log file helps you pinpoint the problem.

Note that dklog.log file is also available in the C:\CMeClient\logs directory. If you open the file, you see almost the identical information as shown in 18.3.2, “Sample log file” on page 439. The dklog.log file is discussed in the next section.

18.3 Tracing EIP Java API

eClient is built on top of the EIP Java APIs. The logging and tracing mechanism in EIP also provides valuable information while you troubleshoot problems in eClient.

When you run an application that is written in EIP Java API - in this case, it is eClient - the EIP log file dklog.log resides in the current working directory. For eClient, the working directory is defined by the WorkingDir parameter in the IDM.properties file, as discussed in 18.2, “Tracing eClient” on page 434. By default, it is C:\CMeClient\logs.

18.3.1 Configuring log manager

EIP Java API has two log managers: the default and LOG4J. You can configure and use only one of them at a time. The same configuration file, cmblogconfig.properties, is used to control which log manager type is used and how to configure them. By default, the file resides in the C:\Program Files\IBM\cmgmt directory. In this section, we discuss how to configure the default log manager.

There are four sections in the cmblogconfig.properties file:

- Section 0: Global Setting
- Section 1: Log Manager Factory Setting

- Section 2: Default Log Manager Setup
- Section 3: Log4J Log Manager Setup

Since we use the default log manager for the discussion, we focus on the first three sections.

Section 0 (Global Setting) specifies the key(s) that control the internal behavior of the logging facility, regardless of which log manager is to be used. The parameter MAX_EXCEPTION_COUNT is defined in the section. By default, it is set to ON. Its valid values are shown in Table 18-2.

Table 18-2 Values for parameter MAX_EXCEPTION_COUNT

Value	Description
ON	Limit the maximum number of exceptions per message ID sent to the logger. The maximum number is 5.
OFF	Unlimit the maximum number of exceptions per message ID sent to the logger.

Whether the logger logs an exception or not is determined by the logging priority setting.

Section 1 (Log Manager Factory Setting) specifies which log manager facility is to be used. The parameter DKLogManagerFactory is defined in the section. By default, the default log manager is used. See Table 18-3 for a list of the valid values.

Table 18-3 Values for parameter DKLogManagerFactory

Value	Description
com.ibm.mm.sdk.logtool.DKLogManager Factory_default	Use the default log manager.
com.ibm.mm.sdk.logtool.DKLogManager Factory_Log4J	Use the Log4J log manager.

Section 2 (Default Log Manager Setup) is to be recognized only when you choose to use the default log manager in section 1. Four parameters are defined in the section.

The parameter DKLogPriorit specifies the Log Priority. Its values are defined in Table 18-4 on page 438. By default, it is set to ERROR.

Table 18-4 Values for parameter DKLogPriorit

Value	Description
DISABLE	Disable logging
FATAL	Log FATAL messages
ERROR	Log FATAL + ERROR messages
PERF	Log FATAL + ERROR + PERFORMANCE messages
INFO	Log FATAL + ERROR + PERFORMANCE + INFO messages
TRACE_NATIVE_API	Log FATAL + ERROR + PERFORMANCE + INFO + TRACE_NATIVE_API messages
TRACE_ENTRY_EXIT	Log FATAL + ERROR + PERFORMANCE + INFO + TRACE_NATIVE_API + TRACE_ENTRY_EXIT messages
TRACE	Log FATAL + ERROR + PERFORMANCE + INFO + TRACE_NATIVE_API + TRACE_ENTRY_EXIT + TRACE messages
DEBUG	Log FATAL + ERROR + PERFORMANCE + INFO + TRACE_NATIVE_API + TRACE_ENTRY_EXIT + TRACE + DEBUG messages

The parameter DKLogOutputSetting specifies log output destinations. Its values are defined in Table 18-5. By default, it is set to 1.

Table 18-5 Values for parameter DKLogOutputSetting

Value	Description
1	log to a file
2	log to Standard Error
3	log to Standard Console

If you have set DKLogOutputSetting = 1, you need to set two additional parameters: DKLogOutputFileName and DKLogOutputFileSize. The parameter DKLogOutputFileName specifies the log file name to be used. The default name is dklog.log. The parameter DKLogOutputFileSize specifies the log file size limit before the log file is rolled over to a backup copy. The log file size is in MB. The default log file size is 5 MB. If the log file size is set to 0, the log file will keep being appended and never get to be rolled over. The rolled-over backup file name is constructed by appending the suffix .sav to the log output file name.

18.3.2 Sample log file

On the EIP server, do the following steps:

1. Open an EIP Development Window by selecting **Start -> Programs -> Enterprise Information Portal for Multiplatforms 8.2 -> Development Window**.
2. Run the command **`cd C:\CMBROOT\SAMPLES\java\icm`**.
3. Run the command **`javac SConnectDisconnectICM.java`** to compile the sample code.
4. Run the command **`java SConnectDisconnectICM icmnlsdb icmadmin passwd`** to run the sample code. Note because the password is incorrect for the user, the command fails to complete.
5. Open the C:\CMBROOT\SAMPLES\java\icm\dklog.log file and you should see the sample log file in Example 18-2. Note that we have the parameter DKLogPriorit=ERROR in the cmblogconfig.properties file while running the sample code.

Example 18-2 Sample dklog.log file

```
[EXC]: 05/12/2003 at 15:52:42.598 CDT @ CM71 (9.30.41.127);
com.ibm.mm.sdk.common.DKLogonFailure #
com.ibm.mm.sdk.logtool.DKLogManagerFactory_default
[USR]: admin (C:\Documents and Settings\admin) @ C:\CMBROOT\SAMPLES\java\icm
[THD]: main ( 677fb987 )
[THG]: main = { main }
[LOC]: com.ibm.mm.sdk.server.DKDatastoreICM:logon
[MSG]: DGL0394A: Error in : DKDatastoreICM.connect; [SERVER = icmnlsdb, USERID
= icmadmin]; ICM7172: The password provided is invalid for this user ID, or it
is NULL. (STATE) : [LS RC = 7172]
   at com.ibm.mm.sdk.server.DKDatastoreICM.logon(DKDatastoreICM.java:3084)
   at com.ibm.mm.sdk.server.DKDatastoreICM.connect(DKDatastoreICM.java:2763)
   at SConnectDisconnectICM.main(SConnectDisconnectICM.java:213)
```

`[EXC]`	Provides the exception type and log manager name. In our example, they are `com.ibm.mm.sdk.common.DKLogonFailure` and `com.ibm.mm.sdk.logtool.DKLogManagerFactory_default`.
`[USR]`	Provides the login user home directory and the current work directory. In our example, they are `C:\Documents and Settings\admin` and `C:\CMBROOT\SAMPLES\java\icm`.
`[THD]`	Provides the name (hashcode) of the thread that reports the error.
`[THG]`	Provides the thread group (threads and sub-thread groups within this thread group). Both of them are `main` in our example.

[LOC]	Provides the class name and the method name within which the error occurs. In our example, it is `com.ibm.mm.sdk.server.DKDatastoreICM:logon`.
[MSG]	Provides the Content Manager return code and the exception type whenever applicable. It is `ICM7172` in our example.

18.4 Additional trace information

Since eClient is a WebSphere application server, you can obtain additional information through the WebSphere logging mechanism. By default, the WebSphere log files for eClient are in the C:\WebSphere\AppServer\logs\eClient_Server directory.

If you have problems displaying the eClient logon window, it is possible that the Web server has problems or has not even started. In this case, the Web server log files and Web server plug-in log files may present tips for you to identify the problem. By default, these log files reside in both the C:\IBMHttpServer\logs and C:\WebSphere\AppServer\logs directories. In the C:\WebSphere\AppServer\logs directory, pay special attention to the activity.log and http_plugin.log files.

18.5 Debugging your application

In addition to tracing, you can debug your applications using programming and external third-party tools.

18.5.1 Debugging Java code using stdout

One way of debugging your Java source code, whether it is in a custom servlet, a custom Java file, or Java code in a JSP, is by writing messages to the stdout file using the System.out.println() method. For example:

```
String myVariable = “test”;
System.out.println(“The value of test is ‘” + myVariable + “‘”);
```

The output from this code will go to the stdout file.

If you use WebSphere Studio Application Developer, the stdout output goes to the console that can be viewed in the WebSphere Studio Application Developer Web perspective.

If you are running in the standard eClient, the location of the stdout file is controlled by WebSphere. The default location is C:\Program Files\IBM\CMeClient\logs\eClient_Server_stdout.log.

To check or change the location of stdout, run the WebSphere Administrative Console. Be sure to stop your WebSphere Studio Application Developer test environment and start the WebSphere server1 service before trying to run the standard WebSphere Administrative Console. They cannot be run at the same time. When the Administrative Console is up, select **Troubleshooting -> Logs and Trace -> eClient_Server**. You can modify the location of stdout from that window.

18.5.2 Debugging JavaScript code

Because JavaScript runs in a Web browser, you cannot use System.out.println() to debug as you would for Java code in your JSP. The following are two methods of debugging JavaScript:

1. Use the alert() statement

   In most cases, adding alert() calls in the JavaScript code embedded in your JSP files is adequate for debugging. The alert() statement displays a message box containing the text that you pass into the function. You can place these messages in the JavaScript contained in the JSP to trace the execution path and display the values of variables.

2. Use JavaScript debugging facilities of your browser.

   Microsoft Internet Explorer has a facility for debugging JavaScript code built into it, as does Netscape Navigator. See the following URL for more information or search for “Debugging client-side scripts” or “JavaScript debugger”:

   `http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vsdebug/html/vxtskDebuggingClient-SideScript.asp`

   For Netscape Navigator, see the following URL for information on debugging JavaScript:

   `http://developer.netscape.com/software/jsdebug.html`

18.6 Typical problems

In this section, we present some typical eClient problems with the problem explanation and possible solutions on a Windows platform. The problems discussed in this section occur in the following core system environment:

- DB2 V8.1
- WebSphere Application Server V5.0
- Content Manager V8.2
- EIP V8.2
- eClient V8.2

Similar problems may also occur in other environments.

We cover the following typical problems, their explanation, and solutions in the following sections:

- “Problem 1: eClient V8.2 cannot access Content Manager V8.1” on page 442
- “Problem 2: Cannot launch eClient logon window” on page 442
- “Problem 3: Cannot launch document” on page 444
- “Problem 4: Persistent blank window during eClient deployment” on page 445
- “Problem 5: Cannot view Chinese character sets in Netscape” on page 445
- “Problem 6: Cannot e-mail document” on page 445
- “Problem 7: eClient is not deployed automatically” on page 446
- “Problem 8: IBM_eClient_82 has unknown status in WebSphere Application Server V5.0” on page 446
- “Problem 9: Only the Search option is available” on page 447
- “Problem 10: Cannot display IST home page properly” on page 447

Problem 1: eClient V8.2 cannot access Content Manager V8.1

eClient V8.2 is installed on a machine with the DB2 V8.1 environment. It cannot access any content servers that are installed in a DB2 V7.1 environment.

Explanation and solution

If eClient is installed on a system with DB2 Version 8, it can only access content servers that are installed on DB2 Version 8.

If you use eClient to access a DB2 Version 8 federated server, it can only access federated entities that are mapped to content servers that are installed on DB2 Version 8. You can only use federated search templates that are defined on those federated entities for search.

Problem 2: Cannot launch eClient logon window

While entering URL `http://<hostname>/eClient82/IDMInit` in a Web browser, the eClient logon window is not displayed.

Explanation and solution

There could be many potential causes for the problem.

1. If you receive the error message “`The page cannot be displayed`**”** in the browser, most likely your HTTP server is not started. To check if the HTTP Server is running, do the following:

   a. Open a Web browser on the machine where the HTTP server is installed.

   b. Enter `http://localhost`.

   c. If the HTTP server home page is displayed, it means that the HTTP server is running. If you still receive the error message “`The page cannot be displayed`” in the browser, it means that the HTTP server is not started.

2. If you have verified that the HTTP server is running and yet you receive the errror message “`Internal Server Error`” while you access the eClient URL, the first thing to check is if the eClient application server is running in WebSphere Application Server.

   a. Open a command window.

   b. Run the command **`cd C:\WebSphere\AppServer\bin`**.

   c. Run the command **`serverStatus -all`**. You should see something similar to Example 18-3.

Example 18-3 Sample output of serverStatus -all command

```
C:\WebSphere\AppServer\bin>serverstatus  -all
ADMU0116I: Tool information is being logged in file
           C:\WebSphere\AppServer\logs\serverStatus.log
ADMU0500I: Retrieving server status for all servers
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: eClient_Server
ADMU0506I: Server name: icmrm
ADMU0506I: Server name: server1
ADMU0509I: The Application Server "eClient_Server" cannot be reached. It
           appears to be stopped.
ADMU0509I: The Application Server "icmrm" cannot be reached. It appears to be
           stopped.
ADMU0508I: The Application Server "server1" is STARTED
```

   d. In Example 18-3, the eClient application server is not running in WebSphere Application Server. In this case, you must enter the following command to start it:

```
startServer eClient_Server
```

3. If you have verified that the HTTP server and the eClient application server are all running, yet you still receive the error message “`Internal Server Error`” when you access the eClient URL, the problem is probably the Web server plug-in. The Web server plug-in is responsible for directing eClient requests from the Web server to the eClient application server.

   Complete the following steps to regenerate the Web server plug-in:

   a. Open a command window.

   b. Run the command **`cd C:\WebSphere\AppServer\bin`**.

   c. Run the command **`serverStatus -all`**. Make sure that server1 is running.

   d. If server1 is not running, run the command `startServer server1` to start it.
   e. Launch WebSphere Application Server Administrative Console by selecting **Start -> Programs -> IBM WebSphere -> Application Server v5.0 -> Administrative Console**.
   f. In the Navigation pane on the left, select **Environment -> Update WebServer Plugin**.
   g. Click **OK** to regenerate the Web server plug-in.
   h. If your HTTP server and WebSphere application server are installed on separate machines, you must copy the new generated Web server plug-in from the WebSphere Application Server machine to the HTTP server machine. For detailed information, see 3.8, "Configuring HTTP Web server" on page 73.

4. If the Logon window is not displayed and you receive the error message "`HTTP 404 - File not found`" or "`The page cannot be found`", make sure that you enter the correct URL. Verify that the default name of the eClient Web application, eclient82, is not changed in the IDM.properties file.

Problem 3: Cannot launch document

eClient cannot launch a document. Microsoft Internet Explorer may show a blank page. Netscape Navigator launches the plug-in, but does not show the document. For example, Adobe Acrobat provides the message "`The file is damaged and could not be repaired`."

Explanation and solution

IBM HTTP Server for Windows cannot launch in the plug-in application documents over 64 KB that are handled by a Web browser.

At this time, there is no permanent fix for this problem in the IBM HTTP Server for Windows. A workaround is to disable the HTTP cache-Afpa and the Afpcache keywords. To accomplish this without using the HTTP GUI administrator:

1. Open the http.conf file located in the \IBM HTTP SERVER\conf\ directory. In this file, you see two keywords:

   ```
   AfpaEnable AfpaCache on
   ```

2. Comment out these keywords by using the pound sign:

   ```
   #AfpaEnable #AfpaCache on
   ```

3. Restart the HTTP server to set these changes.

Problem 4: Persistent blank window during eClient deployment

While deploying eClient into a WebSphere application server during installation, a blank window hangs over five minutes.

Explanation and solution

It is possible that the eClient deployment phase has stopped due to a running WebSphere Java process. Terminate the eClient installation program and then terminate the WebSphere Java processes. On Windows, examine Task Manager to ensure that you terminated all Java processes. On AIX and Solaris, use the command **`ps -ef | grep java`** to verify that the WebSphere application Java processes are not running.

After terminating all running WebSphere Java processes, run the eClient installation program again.

Problem 5: Cannot view Chinese character sets in Netscape

Documents in Simplified Chinese cannot be viewed in the Netscape Navigator 4.7 browser.

Explanation and solution

The English version of Netscape Navigator 4.7 browser does not support viewing of the double-byte character set. If you want to use a Netscape Navigator 4.7 browser to view documents in Simplified Chinese (or Traditional Chinese, Japanese, Korean) in the eClient, you need the Simplified Chinese version of Netscape. Also, Netscape 4.7 does not support rollover text for double-byte character set. To view rollover text in these languages, use Microsoft Internet Explorer.

Alternately, you may use later versions of the Netscape Navigator browser, or Microsoft Internet Explorer.

Problem 6: Cannot e-mail document

When e-mailing documents stored in the Content Manager server, you receive the error message "`java.lang.NullPointerException`" on the browser.

Explanation and solution

The Resource Manager on which the document is stored may not be running. You may retrieve and display the same document in the eClient for a quick verification. If you also failed while retrieving the same document, it is very likely that the Resource Manager is not running. To start the Resource Manager:

1. On the Resource Manager machine, open a command window.

2. Run the command **`cd C:\WebSphere\AppServer\bin`**.
3. Run the command **`startServer icmrm`**.

Problem 7: eClient is not deployed automatically

The eClient installation was completed successfully; however, the eClient application server is not deployed in the WebSphere Application Server.

Explanation and solution

The eClient installation procedure will automatically deploy the eClient application server in WebSphere Application Server at your command. In order to ensure a successful deployment, WebSphere Application Server must have a certain running status at the time of installation, depending on the version of WebSphere Application Server that you are using.

- If you are using WebSphere Application Server AES V4.05, stop any WebSphere Application Server server that is running.
- If you are using WebSphere Application Server AE V4.05, the WebSphere Application Server Administration Server (AE) service in the Windows Control Panel must be running.
- If you are using WebSphere Application Server V5.0, the WebSphere Application Server Administration Server (server1) must be started.

If the WebSphere Application Server was not in the proper status when you installed the eClient, the auto deployment may fail. To manually deploy the eClient application server:

1. Open a command window.
2. Run the command **`cd c:\CmeClient\Save`**.
3. Run the command **`idmwas.bat [userid] [password]`** to deploy it again. The userid and password are used to log on to the WebSphere Application Server.

Problem 8: IBM_eClient_82 has unknown status in WebSphere Application Server V5.0

After you have successfully installed the eClient, everything is working. You are able to log on to eClient, do searches, and retrieve documents. In the WebSphere Application Server Administrative Console, however, the eClient enterprise application IBM_eClient_82 has unknown status.

Explanation and solution

This is a known problem in the WebSphere Application Server. There is no fix at the time of this writing.

Problem 9: Only the Search option is available

After logging on to eClient, only the Search option is displayed on the home page. Other options, such as Import, are not available.

Explanation and solution

Parameters in the IDM.properties file control whether a feature is available to the eClient user. For example, to have the following four options:

- Search
- Import
- Worklists
- Create Folder

on the eClient home page, you must set the parameters in the IDM.properties file as in Example 18-4.

Example 18-4 Sample parameters in IDM.properties file

```
workFlowEnabled=true
importSupported=true
CreateFolderEnabled=true
```

Problem 10: Cannot display IST home page properly

While launching the Information Structuring Tool, the IST home page is not properly displayed.

Explanation and solution

If you receive the error message "`Your current security settings prohibit running ActiveX controls on this page. As a result, the page may not display correctly`." while launching the Information Structuring Tool, most likely you do not have the Java plug-in installed on your browser.

1. Go to the following URL and download the Java plug-in, such as Version 1.4.1_02:

   `http://java.sun.com/products/plugin`

2. Close all browsers.
3. Install the Java plug-in.
4. Open a Web browser and launch the Information Structuring Tool.

If you receive the error message "`Loading Java Applet Failed`" in the browser status bar while launching the Information Structuring Tool, make sure that you are not using a SOCKS proxy server. To turn off the SOCKS proxy server in Internet Explorer:

1. Select **Tools -> Internet options** from the menu bar.

2. Select the **Connections** tab.
3. Click the **LAN Settings** button.
4. Uncheck **Use a proxy server**.
5. Close all browsers.
6. Open a browser and launch the Information Structuring Tool.

18.7 Support channels

For current product information and known problems, go to the official eClient support Web site. You may also post your eClient questions in the EIP user group.

18.7.1 Official support Web site

The official IBM eClient support Web site is:

`http://www-3.ibm.com/software/data/cm/support.html`

On this Web site, you can find:

- Frequently asked questions (FAQs)
- Hints and tips
- Technotes
- Product information
- White papers

18.7.2 Forums

The EIP user group is available at:

`http://w3.ibm.com/forums/forums.htm`

The eClient topics are covered in the forums.software.eip user group. You may also check the group forums.software.contentmanager for Content Manager related topics.

You may post your question to seek advice from others in the eClient community. You can search through other postings for clues that may resolve your problem.

Performance tuning

In this chapter, we discuss performance tuning for eClient.

This chapter covers the following topics:

- Tuning recommendations
- Maintenance and monitoring for performance

19.1 Introduction

The eClient mid-tier server may be viewed as a series of three server components (see Figure 19-1):

- Web server: Handles incoming HTTP requests from clients using browsers.
- Servlet engine: Executes the eClient servlets (which in turn call the EIP Beans).
- Data source: The WebSphere Connection Pool to the back-end Content Manager server.

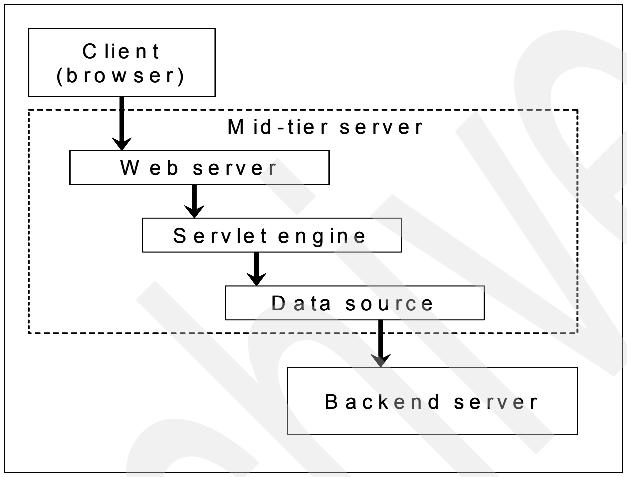


Figure 19-1 eClient mid-tier server

Each component can process a user-defined number of requests simultaneously. If the server component is already processing the maximum number of requests, the new requests will wait in a queue.

The goal of eClient performance tuning is to maximize the number of requests that can be processed efficiently with the limited system resources, in this case, CPU and memory.

Tuning eClient involves setting the number of maximum simultaneous requests and queue length of each server component. Tuning also involves allocating memory resources to JVM. Optimizing performance is not easy, especially for high-volume mid-tier eClient servers. The following are some suggestions from the Content Manager Performance team from the Silicon Valley Lab:

- Use automated test tools to drive a multi-user test load based on your projected workload.

- Performance tuning depends on a coordinated and interdependent tuning of all subsystems. Change only a small number of tuning parameters and evaluate the effect before making additional changes. Iterate running of the test workload.
- Plan for an initial tuning period to maximize confidence and reduce risk before going into production.
- The WebSphere Application Performance Monitoring and Tuning Workshop class is recommended for developers and administrators of custom mid-tier applications.
- The eClient server can be considered as a series of three components. For each component, tune with two goals in mind:
  - Maximize the number of concurrent requests.
  - Avoid overloading the queue of the next component.
- Experiment with the JVM heap size, the connection pool, garbage collection, the number of threads, cloning and clustering.

19.2 Tuning recommendations

The Content Manager Performance team has worked with many customers, conducting a lot of research and testing on how to better tune an eClient system for performance. We found the *Content Manager Performance Tuning Guide*, produced by the team, to be very useful. In this section, we extract the eClient-related tuning recommendations from the guide. We highly recommend reading them before you do any eClient performance tuning.

Remember when tuning a mid-tier server, back up the configuration files before making changes so the original values can be restored if the changes are not effective.

The following is a list of tuning recommendations:

- If users view large documents, view many documents per login session, or frequently manipulate images, you can improve the mid-tier server's scalability with the viewer applet. The applet can bypass the mid-tier server and directly retrieve from the Resource Manager.

  To enable the viewer applet, see "Enable viewer applet" on page 49.

  For information on how to calculate memory usage when using the viewer applet or mid-tier server, see Appendix A, "Calculating memory needed for an image" on page 459.

  To increase the maximum or the minimum heap size for your Java plug-in:

  a. Select **Start -> Settings -> Control Panel -> your Java plug-in**.

   b. Locate the Java Runtime Parameters input field that is either in the Advanced tab or the Basic tab.

   c. Enter the desired heap size values in the Java Runtime Parameters input field, for example:

   ```
   -mx 25m -ms 256m
   ```

   d. Click **Apply**.

- Turn on the WebSphere Connection Pool to maximize the number of real users that the mid-tier server can support. As a starting point, estimate the largest number of users that could log in concurrently, and set the maximum pool size to one-tenth of that number.

  To configure eClient to use IBM WebSphere 4 or 5 Connection pooling, see Chapter 4 in *IBM Content Manager for Multiplatforms / IBM Information Integrator for Content: Installing, configuring, and Managing eClient*, SC27-1350.

  Another good reference for configuring eClient to use WebSphere's connection pooling is as follows:

  http://www-3.ibm.com/software/data/cm/pubs/cm81/eclientpooling/WASpool.htm

- There are several eClient Java Virtual Machine (JVM) configuration parameters that influence performance and scalability. Note that these changes will affect all applications for WebSphere AEs (single server), while for WebSphere AE, they will affect only the eClient application.

  In the WebSphere Advanced Administrative Console Version 4, expand **WebSphere Administrative Domain -> Nodes -> hostname -> Application Servers**, click your eClient server, and do the following settings:

   a. Click the **JVM Settings** tab. As a starting point, set the eClient application's Java heap sizes (both initial and maximum) to about one-fourth of your mid-tier server's physical memory. For example, in a server with 2 GB RAM, set the initial and maximum heap sizes to 512 MB. On Windows or AIX, do not use JVM heap sizes larger than 1700 MB. On Sun Solaris, always set the minimum and maximum JVM heap size parameters to the same value, but not larger than 512 MB.

   b. For Windows and AIX, on the JVM Settings page, click **Advanced JVM Settings**. The Advanced JVM Settings window opens. In the Command line arguments field, type:

   ```
   -Xgcpolicy:optavgpause
   ```

   Click **OK** to save your changes and exit. This parameter allows *concurrent marking* for garbage to be collected before the full garbage collection does its work. This can help to reduce the duration of the garbage collection pauses, which for high-volume systems helps smooth out response times. The benefit of this option can be verified by using `–verbosegc` and examining the "completed in XXX ms" garbage collection times. In one case on performance test servers under a heavy multi-user test load, this option reduced 1000 millisecond garbage collection pauses to about 300 milliseconds. Note: If the "completed in XXX ms" time exceeds 15% of the "YYY ms since last AF" time, the system is spending too much time for garbage collection.

Important: Do not use this setting on Sun Solaris.

c. For Sun Solaris, special tuning is required for optimal garbage collection behavior. In the Advanced JVM Settings, set the following command-line arguments, which control the Sun JVM's "generation" garbage collection:

```
-XX:NewSize=128M
–XX:MaxNewSize=128M
-XX:SurvivorRatio=2
-XX:TargetSurvivorRatio=90
```

Click **OK** to save your changes and exit. The NewSize and MaxNewSize parameters set the size for Sun JVM's more efficient garbage collection. If a heap smaller than 512 MB is used, these two parameters should be set to one-fourth the heap size. The SurvivorRatio parameter sets the sizes of the three subregions in the new generation. The TargetSurvivorRatio parameter controls when objects are moved from the new generation into the old generation. When 90% of the new generation is full, some of the objects will be moved to the older generation to make room in the new generation for new objects.

For more information, see `http://java.sun.com/docs/hotspot/gc`.

d. Click the **Services** tab. Click **Web Container Service** -> **Edit Properties**. The Web Container Service window opens on the General tab. As a starting point, set the minimum number of threads to 20 (50 on a server with four or more CPUs), and set the maximum to 30 (75 on a 4+ CPU server). If under load response times are slow but the CPUs are not saturated, experiment with larger values for Maximum. If the CPUs are saturated and/or response times have wild fluctuations, experiment with values for maximum. Always make sure that **Allow thread allocation beyond maximum** is *not* checked. This setting prevents an overloaded server from thrashing; once the server reaches its limit, additional requests are queued and eventually rejected, so that the requests currently executing have a chance to complete. Also on the General tab, consider increasing the thread inactive timeout (up to, for example, 100), if the WebSphere Resource Analyzer shows the number of concurrent threads swinging wildly.

e. Still in the Web Container Service window, click the **Transport** tab. Consider tuning the maximum keep alives, maximum requests per keep alive, keep alive timeout, and I/O timeout. As a starting point, set the maximum keep alives to about 80% of the CM Connection Pool Maximum Threads value (see below). Then select **Port -> Edit**, and consider tuning the Connection backlog, with a starting value `>= ThreadsPerChild – MaxThreadPoolSize`. Larger values may be less efficient, but can help avoid “`500 Server Internal Error`” messages.

f. Still in the Web Container Service window, click the **Servlet Caching** tab. Disable servlet caching. Click **OK** to save changes and exit.

g. Important: To save changes without erasing the environment variables that you already have, open the **General** tab and click **Environment -> OK -> Apply**.

- On UNIX systems, if a mid-tier server’s federated database is on the same server as the Library Server or the Resource Manager, install the database into a separate instance. Separating the instances makes it possible to tune various database independently and also simplifies the tuning.
- For the HTTP Server, consider tuning these parameters in the httpd.conf file: Loglevel, Timeout, Keepalive, MaxKeepAliveRequests, KeepAliveTimeout, MaxRequestPerChild, and ThreadsPerChild. For ThreadsPerChild, set it to about the expected number of concurrent users.
- The Connection Pool settings can be accessed from the **WebSphere Admin Console -> WebSphere Admin Domain -> Resources -> eClient -> Data Sources -> Connection pooling** tab. As a starting point, consider setting the maximum pool size to about one connection for every 10 expected concurrent users. Also consider tuning the Connection Timeout, Idle Timeout, Orphan Timeout, and Statement Cache Size parameters.
- For the eClient application JVM, monitor the heap size with the WebSphere Resource Analyzer. If “used memory” is constantly close to the “total memory”, then the heap size may need to be increased. Also estimate how often garbage collection occurs, using the WebSphere Resource Analyzer’s JVM Setting. Count the number of data samples for each rise-fall cycle and multiply by the sampling period (default 10 seconds). It may be necessary to reduce the sampling period to five or even one second to see the rise-fall behavior. If garbage collection is occurring more frequently than every 10 seconds, consider increasing the heap size.
- If the heap size is already at the recommended maximum for your platform (1700 MB for AIX or Windows, 512 MB for Sun Solaris), consider the use of cloning the eClient. Each clone has its own JVM and JVM heap, with WebSphere balancing the load from the HTTP Server evenly across the clones. On performance test servers with sufficient memory and processing power, cloning significantly increased scalability.

19.3 Maintenance and monitoring for performance

Routine maintenance tuning and monitoring for performance helps to keep system performance within the expected boundaries. If not maintained on a regular basis, the system performance may degrade over time.

In order to maintain system performance, we recommend the following tasks to be done on a regular basis:

- Routine DB2 tuning (runstat/rebind and reorg)
- Performance monitoring and profile maintenance
- Tuning system parameters and configurations

For details, refer to the following publications:

- *Performance Tuning for Content Manager*, SG24-6949

  This redbook provides detailed information on how to maintain and improve system performance for Content Manager in general.

- *IBM Content Manager V8.2 Performance Tuning Guide*, by the Content Manager Performance team, found in:

  `http://www.ibm.com/software/data/cm/cmgr/mp/support.html`

  Search for "performance tuning guide" under Whitepapers.

  This white paper contains the recommendations from the Content Manager Performance team for Content Manager including eClient.

There are two commands we want to emphasis here to keep your system up and running smoothly, so that your eClient performance is within your expectation: **`runstat/rebind`** and **`reorg`**.

Runstat/rebind

Routinely keep the Library Server and the Resource Manager database statistics and execution plans up to date using **`runstats`** and **`rebind`** to maintain good performance. For **`runstat`** to be effective, running **`rebind`** on the application is necessary after executing **`runstat`**.

The following is a set of commands you can run from a DB2 command line window that will create the script you need to run.

```
db2 connect to db user userid using password
echo db2 connect to db user userid using password >fname.bat
db2 –x "select 'db2 runstats on table 'concat tabschema concat
'.' concat tabname concat 'with distribution and detailed
indexes all ' from syscat.tables where tabschema='schema ' and type='T '">>
fname.bat
echo db2 connect reset >>fname.bat
```

```
db2 connect reset
echo db2rbind db –l logfile all – u userid –p password >>fname.bat
```

Change `db` to the name of your database and change `userid` and `password` for your system values. Do not forget to use capital letters for the schema name.

Run the generated file fname.bat script daily for the first few weeks and during initial database loading. After that, run it weekly or at least monthly as routine performance maintenance, and whenever your database has changed significantly in size.

Note that this should be part of your database administration. Library Server relies heavily on DB2 stored procedures and precompiled access modules to perform its functions. This is why **`runstats`** is so important for maintaining the performance of a Content Manager.

Reorg

Over a period of time, after many insertions, deletions, and updates to the database table data, logically sequential data may be on non-sequential physical data pages. This causes the database manager to perform additional read operations to access data. Additional read operations are also required if a significant number of rows have been deleted. In such a case, you need to reorganize the table to match the index and to reclaim space. You can reorganize the system catalog tables as well as database tables.

When you reorganize tables, you remove empty spaces and arrange table data for efficient access. Reorganizing tables take a lot more time than simply checking (**`reorgchk`**) what tables may require reorganization. It is always a good idea to perform **`reorgchk`** first. If you have already identified the pattern as what tables most likely need reorganization, you may schedule the reorg task on these tables without first running **`reorgchk`**. Do not reorganize tables when there is a lot of server activities since reorg impacts performance. DB2 locks any data in a table that is currently being reorganized.

Use the following command from a DB2 command-line window to check:

```
db2 reorgchk update statistics on table all >out.txt
```

Where `out.txt` contains the result of the command.

To reorganize a specific table, use the following command from a DB2 command line window:

```
db2 reorg table <table name>
```

where `<table name>` is the specific table you want to reorganize.

Part 6

Appendixes

A

Calculating memory needed for an image

This appendix describes how to calculate the approximate amount of memory needed for displaying an image.

Introduction

Memory is consumed differently when using an applet viewer or when using mid-tier conversion.

If users view large image documents, viewing many documents per login session, or frequently manipulate images, you can improve the mid-tier server's scalability with the applet viewer. In this configuration, the applet can bypass the mid-tier server and directly retrieve from the Resource Manager.

But how is memory used by the applet viewer or mid-tier conversion? In this section, we describe how to calculate the approximate amount of memory needed for displaying an image using applet viewer or using mid-tier conversion.

Note that the calculation is approximate and drawn from the experiences of the Content Manager support team. Use them as guidelines only.

For information on how to enable viewer applet, see "Enable viewer applet" on page 49.

Calculate memory usage for applet viewer

Table 19-1 documents the approximate viewer memory usage. Note that the numbers mentioned are approximate and observed.

Table 19-1 Approximate viewer memory usage for applet viewer

State	Peak used memory	Stable memory used	Comment
JRE loaded	15 - 17 MB	3 MB	
Applet viewer loaded	15 - 20 MB	4.5 MB	Needed for the toolbars, tool icons, etc.
Document data loaded	File size * 2	File size	First, the data is copied into small arrays. Once all data is loaded, it is copied into one big array. Hence for the fraction of time, file size * 2 memory is needed.

State	Peak used memory	Stable memory used	Comment
First page conversion	Page width * dpi * height * dpi * pixel depth	Page width * dpi * height * dpi * pixel depth * scale	First, the uncompressed page image data is allocated in memory during decompression. During and after conversion, the image data in memory can be less than the full uncompressed due to a smaller view scale. The scale (= screen resolution / dpi) is usually less than 1, since the screen resolution (typically 96) is less than the dpi resolution (typically 200)
Thumbnail conversion	Page width * dpi * height * dpi * pixel depth	Page width * dpi * height * dpi * pixel depth * scale	For each thumbnail, first the complete uncompressed image data is allocated in memory. During and after conversion, the image data in memory will be much less, since the thumbnail scale is much smaller than the main page scale.

Until the document is dropped (from the end user point of view, closed), all the converted pages will stay in memory to speed up performance.

For example, we are viewing a typical 500 KB TIFF file that is 8.5 x 11 inches in page dimension and 200 dpi resolution with a 1-bit pixel depth. Assuming that we have unlimited Java heap size available, and hence garbage collection never happens, the maximum memory usage for this file will be calculated as follows:

Initially, 30 MB (out of which, almost 23 MB can be recovered if needed before loading the document) +
1 MB during loading of the document (out of which only 500 KB can be recovered if needed) +
14 Mb, derived from 8.5 * 200 * 11 * 200 * 4 during decompression (in enhanced mode 1-bit image's pixel data is stored in one integer, hence 4 bytes) +
4 MB, derived from 8.5 * 96 *11 * 96 * 4 during conversion +
1 MB other overhead
= approximately 50 MB.

For the same file, concurrent peak usage will not go beyond 30 MB, since the initial setup and the conversion process are nonconcurrent activities.

Once the first page is displayed, the data that is held back is as follows:

7 - 8 MB from initial +
4 MB from the page image +
1 - 2 MB overhead
= about 13-14 MB.

When the second document is loaded, it will have the total Java heap size minus 13 - 14 MB memory available, but it will need less concurrent peak memory since the initial toolbar and tools requirement is one time only. If the second document also has similar attributes, the peak memory needed for displaying it will be about 19 - 20 MB. Once it is displayed, another 4 - 5 MB of memory will be held back. This calculation can be similarly extended further.

To put the above in a single formula, we get:

java heap size needed = peakUsage + stableUsage where

peakUsage = greaterOf (InitialNeed, (fileSize*2 + width * dpi * height * dpi * pixelDepth)) and

stableUsage = numberOfDocuments (average fileSize + (average width * average dpi * average height * average dpi * average pixel depth * scale))

Note, all the above calculations and numbers are empirical and approximate, based solely on experience and observations. Factors such as fragmentation of the Java heap and operating system memory allocation techniques make it impossible to make exact predictions. These values and formulae should only be used as explanatory guidelines.

Calculate memory usage for mid-tier conversion

Table 19-2 documents the approximate amount of memory needed for each image viewed by end users.

Table 19-2 Approximate viewer memory usage for mid-tier conversion

State	Peak used memory	Stable memory used	Comment
Image loaded	File size * 2	File size	First, the data is copied into small arrays. Once all data is loaded, it is copied into one big array. Hence for the fraction of time, file size * 2 memory is needed.
Conversion for each of the image page	Page width * dpi * height * dpi * pixel depth	Page width * dpi * height * dpi * pixel depth * scale	First, the uncompressed page image data is allocated in memory during decompression. During and after conversion, the image data in memory can be less than the full uncompressed due to a smaller view scale. The scale (= screen resolution / dpi) is usually less than 1, since the screen resolution (typically 96) is less than the dpi resolution (typically 200).

For example, for a 55 KB TIFF image, when viewing the first page, the amount of peak memory (before any garbage collection kicks in) would be approximately:

55000 * 2 + 8.5 * 200 * 11 * 200 * 1 = 110000 + 3740000 = 3.85 MB

This memory amount may differ as well, depending on what the dpi is for the image.

Note that:

- This memory consumption will be for every image being viewed by end user.
- If the document is kept open by the end user, the memory associated with all the pages that are already converted will not be released. This is to make sure when a user goes to a different page, it will be available quickly.

B

Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

`ftp://www.redbooks.ibm.com/redbooks/SG246964`

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG246964.

Using the Web material

The additional Web material that accompanies this redbook includes the following files:

File name	*Description*
6964samples.zip	Zipped code samples

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	5 MB minimum
Operating System:	Windows 2000/NT
Processor:	800 MHz or higher
Memory:	512 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

Glossary

A

abstract class. An object-oriented programming *class* that represents a concept; classes derived from it represent implementations of the concept. You cannot construct an object of an abstract class; that is, it cannot be instantiated.

access control list. A list consisting of one or more user IDs or user groups and their associated *privileges.* You use access control lists to control user access to *items* and *objects* in the Content Manager system. You use access control lists to control user access to *search templates* in the Enterprise Information Portal system.

access control. The process of ensuring that certain functions and stored *objects* can be accessed only by authorized users in authorized ways.

action list. An approved list of the actions, defined by a system administrator or some other *workflow coordinator,* that a user can perform in a *workflow* or document routing process.

address. The unique code assigned to each device or workstation connected to a network. See also *IP address.*

admission control. The process used by the server to ensure that its bandwidth needs are not compromised by new asset requests.

ADSM. See *Tivoli Storage Manager.*

aggregate bandwidth. Total throughput, in megabits per second, that moves through a server or server subsystem.

alias. In the *Internet,* a name assigned to a server that makes the server independent of the name of its host machine. The alias must be defined in the *domain name server.*

American National Standard Code for Information Interchange (ASCII). The standard code, using a coded character set consisting of 7-bit coded characters (8 bits including parity check), that is used for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters.

analog video. Video in which the information that represents images is in a continuous-scale electrical signal for amplitude and time.

API. See *application programming interface.*

application programming interface (API). A software interface that enables applications to communicate with each other. An API is the set of programming language constructs or statements that can be coded in an application program to obtain the specific functions and services provided by the underlying licensed program.

application server. Software that handles communication with the client requesting an asset and queries of the Content Manager.

archive. Persistent storage used for long-term information retention, typically very inexpensive for each stored unit and slow to access, and often in a different geographic location to protect against equipment failures and natural disasters.

ASCII. See *American National Standard Code for Information Interchange.*

asset group. An organizational grouping within the multimedia file system with similar characteristics. You can use an asset group to allocate resources of a *data pump*. For example, you could establish two asset groups representing distinct departments whose assets should be kept separate for security or billing purposes.

asset. A digital multimedia resource that is stored for later retrieval as requested by an application. An example of such a resource is a digitized video or audio file. An asset is stored as a file in a multimedia file system supported by the *data pump*.

asymmetric video compression. In multimedia applications, the use of a powerful computer to compress a video so that a less powerful system can decompress it.

asynchronous transfer mode (ATM). A transfer mode in which the information is organized into cells; it is asynchronous in the sense that the recurrence of cells containing information from an individual user is not necessarily periodic. ATM is specified in international standards such as ATM Forum UNI 3.1.

attribute group. Convenience grouping of one or more *attributes.* For example, Address might include the attributes Street, City, State, and Zip.

attribute. A unit of data that describes a certain characteristic or property (for example, name, address, age, and so forth) of an item, and which can be used to locate that item. An attribute has a type, which indicates the range of information stored by that attribute, and a value, which is within that range. For example, information about a file in a multimedia file system, such as title, running time, or encoding type (MPEG1, H.263, and so forth). For Enterprise Information Portal, see also *federated attribute* and *native attribute.*

audio. The sound portion of a video signal.

Audio/Video Interleaved (AVI). A RIFF (*Resource Interchange File Format*) file specification that permits audio and video data to be interleaved in a file. The separate tracks can be accessed in alternate chunks for playback or recording while maintaining sequential access on the file device.

AVI. See *Audio/Video Interleaved.*

B

background. The conditions under which low priority, non-interactive programs are run.

bandwidth. The difference, expressed in *Hertz*, between the highest and the lowest frequencies of a range of frequencies. In *asynchronous transfer mode* (ATM), the capacity of a virtual channel, expressed in terms of peak cell ate (PCR), sustainable cell rate (SCR), and maximum burst size (MBS). A measure of the capacity of a communication transport medium (such as a TV cable) to convey data.

base attributes. A set of indexes that is assigned to each *object*. All Content Manager objects have base *attributes*.

batch. An accumulation of data to be processed. A group of records or data processing jobs brought together for processing or transmission.

binary large object (BLOB). A sequence of bytes with a size ranging from 0 bytes to 2 gigabytes. This string does not have an associated code page and character set. Image, audio, and video objects are stored in BLOBs.

bitmap. A representation of an image by an array of bits. A pix map with a depth of one bit plane.

BLOB. See *binary large object.*

block. A string of data elements recorded or transmitted as a unit. The elements can be characters, words, or physical records. Disk device drivers currently use a block size of 32 KB or 256 KB to write to the disk.

bus. A facility for transferring data between several devices located between two end points, only one device being able to transmit at a given moment.

C

cache. A special-purpose buffer, smaller and faster than main storage, used to hold a copy of data that can be accessed frequently. Use of a cache reduces access time, but might increase memory requirements. See also *Resource Manager cache* and *LAN cache.*

caching proxy server. A proxy server that can store the documents it retrieves from other servers in a local *cache*. The catching proxy server can then respond to subsequent requests for these documents without retrieving them from other servers, a process that can improve response time.

cardinality. The number of rows in a database table.

category. See *item type.*

CGI script. A computer program that runs on a Web server and uses the *Common Gateway Interface (CGI)* to perform tasks that are not usually done by a Web server (for example, database access and form processing). A CGI script is a CGI program that is written in a scripting language such as Perl.

CGI. See *Common Gateway Interface.*

child component. Optional second or lower level of a hierarchical *item type.* Each child component is directly associated with the level above it.

CIF. See *common interchange file.*

CIU. See *common interchange unit.*

class. In object-oriented design or programming, a model or template that can be instantiated to create objects with a common definition and therefore, common properties, operations, and behavior. An object is an instance of a class.

Client Application for Windows. A complete object management system provided with Content Manager and written with Content Manager APIs. It supports document and folder creation, storage, and presentation, processing, and access control. You can customize it with user exit routines and partially invoke it with APIs.

client application. An application written with the Content Manager APIs to customize a user interface. An application written with the object-oriented or Internet APIs to access *content servers* from Enterprise Information Portal.

client. A computer system or process that requests a service of another computer system or process that is typically referred to as a server. Multiple clients can share access to a common server.

client/server. In communications, the model of interaction in distributed data processing in which a program at one site sends a request to a program at another site and awaits a response. The requesting program is called a client; the answering program is called a server.

codec. A processor that can code analog audio or video information in digital form for transmission, and decode digital data back to analog form.

collection. A group of objects with a similar set of management rules.

combined search. A query that combines one or more of the following types of searches: *parametric,* text, or image.

Common Gateway Interface (CGI). A standard for the exchange of information between a Web server and programs that are external to it. The external programs can be written in any programming language that is supported by the operating system on which the Web server is running. See *CGI script.*

common interchange file (CIF). A file that contains one ImagePlus Interchange Architecture (IPIA) data stream.

common interchange unit (CIU). The independent unit of transfer for a common interchange file (CIF). It is the part of the CIF that identifies the relationship to the receiving database. A CIF can contain multiple CIUs.

component. Generic term for a *root component* or a *child component.*

compressed audio. A method of digitally encoding and decoding several seconds of voice quality audio per single videodisc frame. This increases the storage capability to several hours of audio per videodisc. Sometimes referred to as still frame audio or sound over still.

compressed video. A video resulting from the process of digitally encoding and decoding a video image or segment using a variety of computer techniques to reduce the amount of data required to represent the content accurately.

compression. The process of eliminating gaps, empty fields, redundancies, and unnecessary data to shorten the length of records or blocks.

connection manager. A Content Manager component that helps maintain connections to the Library Server, rather than starting a new connection for each query. The connection manager has an application programming interface.

connector class. Object-oriented programming *class* that provides standard access to APIs that are native to specific *content servers.*

constructor. In programming languages, a method that has the same name as a class and is used to create and initialize objects of that class.

container. An element of the user interface that holds objects. In the *folder manager,* an *object* that can contain other folders or documents.

content class. See *MIME type.*

content server. A software system that stores multimedia and business data and the related metadata required for users to work with that data. Content Manager and Content Manager ImagePlus for OS/390 are examples of content servers.

controller. The functional component responsible for resource management (load balancing and admission control). The controller communicates with one or more *data pumps* to initiate and terminate connections to clients.

cursor. A named control structure used by an application program to point to a specific row within some ordered set of rows. The cursor is used to retrieve rows from the set.

D

data format. See *MIME type.*

data pump. The combination of the disks that hold the data and the networking hardware and software required to deliver assets to clients.

data rate. The rate at which data is transmitted or received from a device. Interactive applications tend to require a high data rate, while batch applications can usually tolerate lower data rates.

data striping. Storage process in which information is split into blocks (a fixed amount of data) and the blocks are written to (or read from) a series of disks in parallel.

data transfer rate. The average number of bits, characters, or blocks per unit time passing between corresponding equipment in a data transmission system. Notes: The rate is expressed in bits, characters, or blocks per second, minute, or hour. Corresponding equipment should be indicated; for example, modems, intermediate equipment, or source and sink.

datastore. Generic term for a place (such as a database system, file, or directory) where data is stored. In an application program, a virtual representation of a *content server.*

DCA. See *document content architecture.*

DCE. See *Distributed Computing Environment*.

DDO. See *dynamic data object.*

decode. To convert data by reversing the effect of some previous encoding.

decompression. Process of restoring compressed data to its original state, so that it can be used again.

destager. A function of the Content Manager *Resource Manager* that moves objects from the *staging area* to the first step in the object's *migration policy.* This function has been eliminated in Content Manager Version 8.

device driver. Software used to manage a specific device. Other software uses the device driver as the interface to the device for reading, writing, and control functions.

device manager. In a Content Manager system, the interface between the *Resource Manager* and one or more physical devices.

digital audio. Audio tones represented by machine-readable binary numbers rather than by analog recording techniques.

digital video. Video in which the information (usually including audio) is encoded as a sequence of binary digits. The information is usually compressed. It can be stored and transported just as any other digital information. Viewing digital video involves decompressing the video data, converting it to an analog form, displaying the video on a monitor, and playing the sound through an amplifier and speakers.

digital. Pertaining to data in the form of digits.

digitize. To convert analog video and audio signals into digital format.

digitized image. An image derived from a scanning device or a digitizing card with a camera.

Distributed Computing Environment (DCE). The Open Software Foundation (OSF) specification (or a product derived from this specification) that assists in networking. DCE provides such functions as authentication, directory service (DS), and remote procedure call (RPC).

document content architecture (DCA). An architecture that guarantees information integrity for a document being interchanged in an office system network. DCA provides the rule for specifying form and meaning of a document. It defines revisable form text (changeable) and final form text (unchangeable).

document root directory. The primary directory where a Web server stores accessible documents. When the server receives requests that do not point to a specific directory, it tries to serve the request from this directory.

document routing process. In Content Manager a sequence of *work steps,* and the rules governing those steps, through which a *document* or *folder* travels while it is being processed.

document type definition (DTD). The rules that specify the structure for a particular class of XML documents. The DTD defines the structure with elements, attributes, and notations, and it establishes constraints for how each element, attribute, and notation can be used within the particular class of documents. A DTD is analogous to a database schema in that the DTD completely describes the structure for a particular markup language.

document. An *item* that can be stored, retrieved, and exchanged among Content Manager systems and users as a separate unit. An item given the “document” semantic type is expected to contain information that forms a document, but does not rigidly mean an implementation of a specific document model. An item created from a “document” (also known as “document model”) classified item type means that the item will contain document parts, a specific implementation of a document model provided by Content Manager. Document classified item types may create items given either the document or folder semantic type. The document parts can include varied types of content, including for example, text, images, and spreadsheets.

domain name server. In the *Internet* suite of *protocols*, a server that responds to queries from clients for name-to-address and address-to-name mappings as well as for other information.

domain name. In the *Internet* suite of *protocols*, a name of a host system. A domain name consists of a sequence of subnames separated by a delimiter character.

domain. That part of a computer network in which the data processing resources are under common control.

DTD. See *document type definition.*

duplex. Pertains to communications data that can be sent and received at the same time. Synonymous with full duplex and FDX. Contrast with half duplex.

dynamic data object (DDO). In an application program, a generic representation of a stored object that is used to move that object in to, and out of, storage.

E

element. An *object* that the *list manager* allocates for an application.

encode. To convert data by using a code in such a manner that reconversion to the original form is possible.

Ethernet. A 10-Mbps baseband local area network that allows multiple stations to access the transmission medium at will without prior coordination, avoids contention by using carrier sense and deference, and resolves contention by using collision detection and transmission.

extended data object (XDO). In an application program, a generic representation of a stored complex multimedia *object* that is used to move that object in to, and out of, storage. XDOs are most often contained within *DDOs.*

Extensible Markup Language (XML). A standard metalanguage for defining markup languages that was derived from, and is a subset of, SGML. XML omits the more complex and less-used parts of SGML and makes it much easier to write applications to handle document types, author and manage structured information, and transmit and share structured information across diverse computing systems. The use of XML does not require the robust applications and processing that is necessary for SGML. XML is being developed under the auspices of the World Wide Web Consortium (W3C).

External Data Representation (XDR). A standard, developed by Sun Microsystems, Incorporated, for representing data in machine-independent format.

F

FDDI. See *Fiber Distributed Data Interface.*

feature. The visual content information that is stored in the image search server. Also, the visual traits that image search applications use to determine matches. The four *QBIC®* features are average color, histogram color, positional color, and texture.

federated attribute. An Enterprise Information Portal metadata category that is mapped to *native attributes* in one or more *content servers*. For example, the federated attribute, policy number, can be mapped to an *attribute,* policy num, in Content Manager and to an attribute, policy ID, in Content Manager ImagePlus for OS/390.

federated collection. A grouping of objects that results from a *federated search.*

federated datastore. Virtual representation of any number of specific *content servers,* such as Content Manager.

federated entity. An Enterprise Information Portal metadata object that is comprised of *federated attributes* and optionally associated with one or more *federated text indexes.*

federated search. A query issued from Enterprise Information Portal that simultaneously searches for data in one or more *content servers,* which can be heterogeneous.

federated text index. An Enterprise Information Portal metadata object that is mapped to one or more *native text indexes* in one or more *content servers.*

Fiber Distributed Data Interface. An American National Standards Institute (ANSI) standard for a 100-Mbps LAN using optical fiber cables.

file name extension. An addition to a file name that identifies the file type (for example, text file or program file).

file system manager. The component that manages the multimedia file system.

file system. In AIX, the method of partitioning a hard drive for storage. See also *multimedia file system.*

File Transfer Protocol (FTP). In the *Internet* suite of *protocols*, an application layer protocol that uses *Transmission Control Protocol (TCP)* and Telnet services to transfer bulk-data files between machines or hosts.

firewall. (1) In communication, a functional unit that protects and controls the connection of one network to other networks. The firewall (a) prevents unwanted or unauthorized communication traffic from entering the protected network and (b) allows only selected communication traffic to leave the protected network. (2) In equipment, a partition used to control the spread of fire.

folder manager. The Content Manager model for managing data as online documents and folders. You can use the folder manager APIs as the primary interface between your applications and the Content Manager content servers.

folder. A *item* of any item type, regardless of classification, with the "folder" semantic type. Any item with the folder semantic type will contain specific folder functionality provided by Content Manager, in addition to all non-resource item capabilities and any additional functionality available from an item type classification such as document model or resource. Folders may contain any number of items of any type, including documents and sub-folders. A folder is indexed by *attributes*.

fps. Frames per second. The number of frames displayed per second.

fragment. The smallest unit of file system disk space allocation. A fragment can be 512, 1024, 2048, or 4096 bytes in size. The fragment size is defined when a file system is created.

FTP. See *File Transfer Protocol.*

full-motion video. Video reproduction at 30 frames per second (*fps*) for *NTSC* signals or 25 fps for *PAL®* signals.

full duplex (FDX). See *duplex*

G

gateway. A functional unit that interconnects two computer networks with different network architectures. A gateway connects networks or systems of different architectures. A bridge interconnects networks or systems with the same or similar architectures.

GB. See *gigabyte.*

gigabyte (GB). For processor storage, real and virtual storage, and channel volume, 2^{30}, or 1 073 741 824 bytes. For disk storage capacity and communications volume, 1 000 000 000 bytes.

H

half duplex (HD or HDX). Pertains to communications in which data can be sent in only one direction at a time. Contrast with *duplex*.

handle. A character string that represents an object, and is used to retrieve the object.

Hertz (Hz). A unit of frequency equal to one cycle per second. In the United States, line frequency is 60 Hz or a change in voltage polarity 120 times per second; in Europe, line frequency is 50 Hz or a change in voltage polarity 100 times per second.

history log. A file that keeps a record of activities for a *workflow.*

home page. The initial Web page that is returned by a Web site when you enter the address for the Web site in a Web browser. For example, if a user specifies the address for the IBM Web site, which is http://www.ibm.com, the Web page that is returned is the IBM home page. Essentially, the home page is the entry point for accessing the contents of the Web site.

host name. In the *Internet* suite of *protocols*, the name given to a computer. Sometimes, host name refers to the fully qualified domain name; other times, it is used to mean the most specific subname of a fully qualified domain name. For example, if mycomputer.city.company.com is the fully qualified domain name, either of the following might be considered the host name: mycomputer.city.company.com mycomputer

host. A computer, connected to a network, which provides an access point to that network. A host can be a client, a server, or a client and a server simultaneously.

HTML. See *Hypertext Markup Language.*

HTTP (Hypertext Transfer Protocol). In the *Internet* suite of *protocols*, the protocol that is used to transfer and display hypertext documents

HTTP daemon. A multi-threaded Web server that receives incoming *Hypertext Transfer Protocol (HTTP)* requests.

HTTP method. An action used by the *Hypertext Transfer Protocol (HTTP)*. HTTP methods include GET, POST, and PUT.

HTTPd. See *HTTP daemon.*

Hypertext Markup Language (HTML). A markup language that conforms to the SGML standard and was designed primarily to support the online display of textual and graphical information that includes hypertext links.

Hz. See *Hertz.*

I

I frame (information frame). In video compression a frame that has been compressed independently of any other frames. Also referred to as a reference frame, intra frame, or still frame.

Image Object Content Architecture (IOCA). A collection of constructs used to interchange and present images.

index class subset. In earlier Content Manager, a view of an *index class* that an application uses to store, retrieve, and display folders and objects.

index class view. In earlier Content Manager, the term used in the APIs for *index class subset.*

index class. See *item type.*

index. To add or edit the attribute values that identify a specific *item* or *object* so that it can be retrieved later.

information mining. The automated process of extracting key information from text (summarization), finding predominant themes in a collection of documents (categorization), and searching for relevant documents using powerful and flexible queries.

inline. In Content Manager, an object that is online and in a drive, but has no active *mounts.* Contrast with *mounted*.

i-node. In the AIX operating system, the internal structure that describes the individual files in the operating system; there is one i-node for each file. An i-node contains the node, type, owner, and location of a file. A table of i-nodes is stored near the beginning of a *file system*.

interactive video. Combining video and computer technology so the user's actions determine the sequence and direction the application takes.

interchange. The capability to import or export an image with its index from one Content Manager ImagePlus for OS/390 system to another ImagePlus system using a *common interchange file* or *common interchange unit.*

Internet Protocol (IP). In the *Internet* suite of *protocols*, a connectionless protocol that routes data through a network or interconnected networks and acts as an intermediary between the higher protocol layers and the physical network.

Internet. The worldwide collection of interconnected networks that use the Internet suite of *protocols* and permit public access.

intranet. A private network that integrates *Internet* standards and applications (such as Web browsers) with an organization's existing computer networking infrastructure.

IOCA. See *Image Object Content Architecture.*

IP address. The unique 32-bit address that specifies the actual location of each device or workstation on the *Internet*. The address field contains two parts: the first part is the network address; the second part is the host number. For example, 9.67.97.103 is an IP address.

IP multicast. Transmission of an *Internet Protocol (IP)* datagram to a set of systems that form a single multicast group. See *multicast.*

IP. See *Internet Protocol.*

ISO-9660. Format used for files on CD-ROM. Used with DOS.

isochronous. A communications capability that delivers a signal at a specified, bounded rate, which is desirable for continuous data such as voice and full-motion video.

item type classification. A categorization within an *item type* that further identifies the *items* of that item type. All items of the same item type have the same item type classification. Content Manager supplies the following item type classifications: *folder, document,* object, video, image, and text; users can also define their own item type classifications.

item type. A template for defining and later locating like *items,* consisting of a *root component,* zero or more *child components,* and a classification.

item. In Content Manager, generic term for an instance of an *item type.* For example, an item might be a *folder, document,* video, or image. Generic term for the smallest unit of information that Enterprise Information Portal administers. Each item has an identifier. For example, an item might be a *folder* or a *document.*

iterator. A class or construct that you use to step through a collection of objects one at a time.

J

JavaBeans. A platform-independent, software component technology for building reusable Java components called "beans." After they are built, these beans can be made available for use by other software engineers or can be used in Java applications. Using JavaBeans, software engineers can manipulate and assemble beans in a graphical drag-and-drop development environment.

Joint Photographic Experts Group (JPEG). A group that worked to establish the standard for the compression of digitized continuous-tone images. The standard for still pictures developed by this group.

JPEG. See *Joint Photographic Experts Group.*

K

Kb. See *Kilobit.*

KB. See *Kilobyte.*

kbps. *Kilobits* per second.

key field. See *attribute.*

kilobit (Kb). For processor storage, real and virtual storage, and channel volume, 210 or 1024 bits. For disk storage capacity and communications volume, 1000 bits.

kilobyte (KB). For processor storage, real and virtual storage, and channel volume, 210 or 1024 bytes. For disk storage capacity and communications volume, 1000 bytes.

L

LAN cache. An area of temporary storage on a local Resource Manager that contains a copy of objects stored on a remote Resource Manager.

LAN. See *local area network.*

latency. The time interval between the instant at which an instruction control unit initiates a call for data and the instant at which the actual transfer of the data starts.

LBR. See *low bit rate.*

library client. The component of a Content Manager system that provides a low-level programming interface for the library system. The library client includes APIs that are part of the software developer's kit.

library object. See *item.*

Library Server. The component of a Content Manager system that stores, manages, and handles queries on *items*.

link. A directional relationship between two *items:* the source and the target. You can use a set of links to model one-to-many associations. Contrast with *reference.*

local area network (LAN). A network in which a set of devices are connected to one another for communication and that can be connected to a larger network.

low bit rate (LBR). A generic term for an interleaved H.263/G.723 stream. Low bit rate streams range from 6.4 kbps up to 384 kbps.

M

machine-generated data structure (MGDS). An IBM structured data format protocol for passing character data among the various Content Manager ImagePlus for OS/390 programs. Data extracted from an image and put into general data stream (GDS) format.

management class. The term used in the APIs for *migration policy.*

Management Information Base (MIB). A collection of objects that can be accessed by means of a network management *protocol*.

maximum transmission unit (MTU). In *LANs,* the largest possible unit of data that can be sent on a given physical medium in a single frame. For example, the MTU for *Ethernet* is 1500 bytes.

Mb. See *megabit.*

MB. See *megabyte.*

Mbps. *Megabits* per second.

MCA. See *Micro Channel® architecture.*

media archiver. A physical device that is used for storing audio and video stream data. The VideoCharger is a type of media archiver.

media server. An AIX-based component of the Content Manager system that is used for storing and accessing video files.

megabit (Mb). (1) For processor storage, real and virtual storage, and channel volume, 220 or 1 048 576 bits. (2) For disk storage capacity and communications volume, 1 000 000 bits.

megabyte (MB). (1) For processor storage, real and virtual storage, and channel volume, 220 or 1 048 576 bytes. (2) For disk storage capacity and communications volume, 1 000 000 bytes.

method. In Java design or programming, the software that implements the behavior specified by an operation. Synonymous with member function in C++.

MGDS. See *machine-generated data structure.*

MIB variable. A managed object that is defined in the *Management Information Base (MIB).* The managed object is defined by a textual name and a corresponding object identifier, a syntax, an access mode, a status, and a description of the semantics of the managed object. The MIB Variable contains pertinent management information that is accessible as defined by the access mode.

MIB. See *Management Information Base.*

Micro Channel Architecture (MCA). The rules that define how subsystems and adapters use the Micro Channel *bus* in a computer. The architecture defines the services that each subsystem can or must provide.

MIDI. See *Musical Instrument Digital Interface.*

migration policy. A user-defined schedule for moving *objects* from one *storage class* to the next. It describes the retention and class transition characteristics for a group of objects in a storage hierarchy.

migration. The process of moving data and source from one computer system to another computer system without converting the data, such as when moving to a new operating environment. Installation of a new version or release of a program to replace an earlier version or release.

migrator. A function of the *Resource Manager* that checks *migration policies* and moves objects to the next *storage class* when they are scheduled to move.

MIME type. An Internet standard for identifying the type of object being transferred across the Internet. MIME types include several variants of audio, image, and video. Each object has a MIME type.

mount. To place a data medium in a position to operate.

mounted. In Content Manager, an object that is online and in a drive, with active *mounts*. Contrast with *inline*.

Moving Pictures Expert Group (MPEG). A group that is working to establish a standard for compressing and storing motion video and animation in digital form. The standard under development by this group.

MPEG. See *Moving Pictures Expert Group.*

MTU. See *maximum transmission unit.*

multicast. Transmission of the same data to a selected group of destinations.

multimedia file system. A *file system* that is optimized for the storage and delivery of video and audio.

multimedia. Combining different media elements (text, graphics, audio, still image, video, animation) for display and control from a computer.

Multipurpose Internet Mail Extensions (MIME). See *MIME type.*

Musical Instrument Digital Interface (MIDI). A *protocol* that allows a synthesizer to send signals to another synthesizer or to a computer, or a computer to a musical instrument, or a computer to another computer.

N

name server. See *domain name server.*

native attribute. A characteristic of an object that is managed on a specific *content server* and that is specific to that content server. For example, the *key field* policy num might be a native attribute in a Content Manager content server, whereas the field policy ID might be a native attribute in an Content Manager OnDemand content server.

native entity. An *object* that is managed on a specific *content server* and that is comprised of *native attributes.* For example, Content Manager *index classes* are native entities comprised of Content Manager *key fields.*

native text index. An index of the text *items* that are managed on a specific *content server.* For example, a single text search index on a Content Manager content server.

network table file. A text file that contains the system-specific configuration information for each node in a Content Manager system. Each node in the system must have a network table file that identifies the node and lists the nodes that it needs to connect to. The name of a network table is FRNOLINT.TBL.

O

Object Linking and Embedding (OLE). A Microsoft specification for both linking and embedding applications so that they can be activated from within other applications.

Object Server cache. See *Resource Manager cache.*

Object Server. See *Resource Manager.*

object. Any digital content that a user can store, retrieve and manipulate as a single unit, for example, *JPEG* images, MP3 audio, *AVI* video, and a text block from a book.

OLE. See *Object Linking and Embedding.*

P

package. A collection of related *classes* and interfaces that provides access protection and namespace management.

page pool. The area in the shared memory segment from which buffers are allocated for data that is read from or written to disk. Page pool size is one of the file manager startup configuration parameters.

PAL. See *Phase Alternation Line.*

parametric search. A query for *objects* that is based on the *properties* of the objects.

part. See *object.*

patron. The term used in the Content Manager APIs for *user.*

pattern-matching character. See *wildcard character.*

PCI. See *Peripheral Component Interconnect.*

peak rate. The maximum rate encountered over a given period of time.

performance group. A group of file systems sharing system resources that can affect file system performance.

Peripheral Component Interconnect (PCI). A type of *bus* architecture.

persistent identifier (PID). An identifier that uniquely identifies an *object,* regardless of where it is stored. The PID consists of both an item ID and a location.

Phase Alternation Line (PAL). The television broadcast standard for European video outside of France and the countries of the former Soviet Union.

PID. See *persistent identifier.*

pin. Keeping the program from being paged out after it is loaded into memory.

port group. A logical name used to group one or more ports (network devices or interfaces) of the same network type that can be used to reach a given end-user destination. For example, if multiple *ATM* adapters in the VideoCharger Server complex are connected to the same ATM networks, these adapters can be configured under the same port group. The controller selects ports as necessary to balance the load.

port. A system or network access point for data entry or exit. In the *Internet* suite of *protocols*, a specific logical connector between the *Transmission Control Protocol (TCP)* or the *User Datagram Protocol (UDP)* and a higher-level protocol or application.

presentation formatter. A *CGI* program that defines the forms used to select and present assets to clients.

privilege set. A collection of *privileges* for working with system components and functions. The administrator assigns privilege sets to users (user IDs) and *user groups*.

privilege. The right to access a specific *object* in a specific way. Privileges includes rights such as creating, deleting, and selecting objects stored in the system. Privileges are assigned by the administrator.

property. A characteristic of an *object* that describes the object. A property can be changed or modified. Type style is an example of a property.

protocol gateway. A type of *firewall* that protects computers in a business network from access by users outside that network.

protocol. The meanings of, and the sequencing rules for, requests and responses used for managing a network, transferring data, and synchronizing the states of network components.

proxy server. A server that receives requests intended for another server and that acts on the client's behalf (as the client's proxy) to obtain the requested service. A proxy server is often used when the client and the server are incompatible for direct connection (for example, when the client is unable to meet the security authentication requirements of the server but should be permitted some services).

purger. A function of the *Resource Manager* that removes *objects* from the system.

Q

QBIC. See *query by image content.*

quality of service (Do's). For *an asynchronous transfer mode (ATM)* virtual channel or a Networking BroadBand Services (NBBS) network connection, a set of communication characteristics such as end-to-end delay, jitter, and packet loss ratio.

query by image content (QBIC). A query technology that enables searches based on visual content, called features, rather than plain text. Using QBIC, you can search for objects based on their visual characteristics, such as color and texture.

query string. A character string that specifies the properties and property values for a query. You can create the query string in an application and pass it to the query.

R

RAID. See *Redundant Array of Independent Disks*.

rank. An integer value that signifies the relevance of a given part to the results of a query. A higher rank signifies a closer match.

Readme file. A file that should be viewed before the program associated with it is installed or run. A Readme file typically contains last-minute product information, installation information, or tips for using the product.

real time. The processing of information that returns a result so rapidly that the interaction appears to be instantaneous.

Real-Time Transport Protocol (RTP). A *protocol* that provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over *multicast* or unicast network services.

rebalance. Restriping and redistributing data across the available hard disks after a disk or disks have been removed from a *file system.*

Redundant Array of Independent Disks (RAID). A collection of two or more disk drives that present the image of a single disk drive to the system. In the event of a single device failure, the data can be read or regenerated from the other disk drives in the array.

reference. Single direction, one-to-one association between a root or *child component* and another *root component.* Contrast with *link.*

release. To remove suspend criteria from an *item.* A suspended item is released when the criteria have been met, or when a user with proper authority overrides the criteria and manually releases it.

Remote Method Invocation (RMI). A set of APIs that enables distributed programming. An object in one Java Virtual Machine (JVM) can invoke methods on objects in other JVMs.

remote procedure call (RPC). A facility that a *client* uses to request the execution of a procedure call from a server. This facility includes a library of procedures and an external data representation. A client request to a service provider located in another node.

render. To take data that is not typically image-oriented and depict or display it as an image. In Content Manager, word-processing documents can be rendered as images for display purposes.

request. The part of a Web address that follows the *protocol* and server *host name*. For example, in the *address* http://www.ibm.com/dir/file.html, the request is /dir/file.html.

ReSerVation Protocol (RSVP). A resource reservation setup *protocol* designed for an integrated services *Internet*. The protocol provides receiver-initiated setup of resource reservations for *multicast* and unicast data flows.

Resource Interchange File Format (RIFF). Used for storing sound or graphics for playback on different types of computer equipment.

Resource Manager cache. The working storage area for the *Resource Manager*. Also called the *staging area.*

Resource Manager. The component of a Content Manager system that manages *objects.* These objects are referred to by *items* stored on the *Library Server.*

Response time. The elapsed time between when a request is submitted and when the response from that request is returned**.**

restriping. Redistributing and rebalancing data across all available and defined disks in a *multimedia file system*. This is typically done when a disk is removed from a file system for repair or when a new disk is added to a *file system*.

RIFF. See *Resource Interchange File Format.*

RLE. See *Run-Length Encoding.*

RMI server. A server that implements the Java *Remote Method Invocation (RMI)* distributed object model.

root component. The first or only level of a hierarchical *item type,* consisting of related system- and user-defined *attributes.*

RPC. See *remote procedure call.*

RSVP. See *ReSerVation Protocol.*

RTP. See *Real-Time Transport Protocol.*

Run-Length Encoding (RLE). A type of *compression* that is based on strings of repeated, adjacent characters or symbols, which are called "runs."

S

SCSI. See *small computer system interface.*

search criteria. In Content Manager, *attribute* values that are used to retrieve a stored *item*. In Enterprise Information Portal, specific fields that an administrator defines for a *search template* that limit or further define choices available to the *users*.

search template. A form, consisting of *search criteria* designed by an administrator, for a specific type of federated search. The administrator also identifies the *users* and *user groups* who can access each search template.

semantic type. The usage or rules for an *item.* Base, annotation, and note are semantic types supplied by Content Manager; users can also define their own semantic types.

server definition. The characteristics of a specific *content server* that uniquely identify it to Enterprise Information Portal.

server inventory. The comprehensive list of *native entities* and *native attributes* from specified *content servers.*

server type definition. The list of characteristics, as identified by the administrator, required to uniquely identify a custom server of a certain type to Enterprise Information Portal.

server. A functional unit that provides services to one or more clients over a network. Examples include a file server, a print server, and a mail server.

Simple Network Management Protocol (SNMP). In the *Internet* suite of *protocols*, a network management protocol that is used to monitor routers and attached networks. SNMP is an application layer protocol. Information on devices managed is defined and stored in the application's *Management Information Base (MIB)*.

small computer system interface (SCSI). A standard hardware interface that enables a variety of peripheral devices to communicate with one another.

SMIT. See *System Management Interface Tool.*

SMS. See *system-managed storage.*

SNMP. See *Simple Network Management Protocol.*

SQL communication area (SQLCA). A set of variables that provides an application program with information about the execution of its SQL statements or its requests from the database manager.

SQL descriptor area (SQLDA). (1) A set of variables that is used in the processing of certain SQL statements. The SQLDA is intended for dynamic SQL programs.
(2) A structure that describes input variables, output variables, or the columns of a result table.
Term3 definition.

SQL. See Structured Query Language.

SQLCA. See SQL communication area.

SQLDA. See SQL descriptor area.

SQLVAR. Collective name for a sequence of variables that has arbitrary number of occurrences within an SQLDA. It is followed by four variables in SQLDA, and has two types: base SQLVARs, and secondary SQLVARs. Base SQLVARs are always present. They contain the base information about the column, parameter marker, or host variable such as data type code, length attribute, column name, host variable address, and indicator variable address. Secondary SQLVARs are optional. For user-defined types, they contain the user-defined type name. For reference types, they contain the target type of the reference. For LOBs, they contain the length attribute of the host variable and a pointer to the buffer that contains the actual length.

staging area. The working storage area for the Resource Manager. Also referred to as *Resource Manager cache.*

staging. The process of moving a stored *object* from an offline or low-priority device back to an online or higher priority device, usually on demand of the system or on request of a user. When a user requests an object stored in permanent storage, a working copy is written to the *staging area*.

stand-alone system. A pre-configured Content Manager system that installs all of the components of a Content Manager system on a single personal computer.

sticky pool. The part of the *page pool* that is made available to cache the first block of frequently used interactive files. Sticky pool size is one of the file manager startup configuration parameters.

storage class. Identifies the type of media that an object is stored on. It is not directly associated with a physical location; however, it is directly associated with the *device manager*. Types of storage classes include: DASD, Fixed Disk, Optical, Stream, Tape, TSM

storage group. Associates a storage system to a storage class.

storage system. A generic term for storage in the Content Manager system. See *TSM volume*, *media archiver*, and *volume.*

streamed data. Any data sent over a network connection at a specified rate. A stream can be one data type or a combination of types. Data rates, which are expressed in bits per second, vary for different types of streams and networks.

stripe group. A collection of disks that are grouped together for serving media streams. The *multimedia file system* uses stripe groups to optimize delivery of multimedia *assets*.

stripe width. The size of the block that data is split into for *striping*.

striping. Splitting data to be written into equal blocks and writing blocks simultaneously to separate disk drives. Striping maximizes performance to the disks. Reading the data back is also scheduled in parallel, with a block being read concurrently from each disk then reassembled at the host.

Structured Query Language (SQL). A standardized language for defining and manipulating data in a relational database.

subclass. A *class* that is derived from another class. One or more classes might be between the class and subclass.

superclass. A *class* from which a class is derived. One or more classes might be between the class and superclass.

suspend. To remove an *object* from its *workflow* and define the suspension criteria needed to activate it. Later activating the object enables it to continue processing.

System Management Interface Tool (SMIT). An interface tool of the AIX operating system for installing, maintaining, configuring, and diagnosing tasks.

system-managed storage (SMS). The Content Manager approach to storage management. The system determines object placement, and automatically manages object backup, movement, space, and security.

T

table of contents (TOC). The list of *documents* and *folders* that are contained in a folder or *workbasket.* Search results are displayed as a folder table of contents.

Tagged Image File Format (TIFF). A file format for storing high-quality graphics.

TCP. See *Transmission Control Protocol.*

TCP/IP. See *Transmission Control Protocol/Internet Protocol.*

thin client. A client that has little or no installed software but has access to software that is managed and delivered by network servers that are attached to it. A thin client is an alternative to a full-function client such as a workstation.

throughput. (1) A measure of the amount of information transmitted over a network in a given period of time. For example, a network's data transfer rate is usually measured in bits per second. Throughput is a measure of performance. It is also measured in *kbps* or *Mbps*. (2) A measure of the amount of work over a period of time. In other words, it is the number of workload operations that can be accomplished per unit of time.

TIFF. See *Tagged Image File Format.*

Tivoli Storage Manager (TSM). A *client/server* product that provides storage management and data access services in a heterogeneous environment. It supports various communication methods, provides administrative facilities to manage the backup and storage of files, and provides facilities for scheduling backup operations.

TOC. See *table of contents.*

token ring. According to IEEE 802.5, network technology that controls media access by passing a token (special packet or frame) between media-attached stations.

token-ring network. A network that uses a ring topology, in which tokens are passed in a circuit from node to node. A node that is ready to send can capture the token and insert data for transmission.

topology. In communications, the physical or logical arrangement of nodes in a network, especially the relationships among nodes and the links between them.

Transmission Control Protocol (TCP). A communications *protocol* used in the *Internet* and in any network that follows the Internet Engineering Task Force (IETF) standards for internetwork protocol. TCP provides a reliable host-to-host protocol between hosts in packet-switched communications networks and in interconnected systems of such networks. It uses the *Internet Protocol (IP)* as the underlying protocol.

Transmission Control Protocol/Internet Protocol (TCP/IP). The suite of transport and application *protocols* that run over the Internet Protocol.

TSM volume. A logical area of storage that is managed by *Tivoli Storage Manager.*

TSM. See *Tivoli Storage Manager.*

U

UDP. See *User Datagram Protocol.*

uniform resource locator (URL). A sequence of characters that represent information resources on a computer or in a network such as the Internet. This sequence of characters includes the abbreviated name of the protocol used to access the information resource and the information used by the protocol to locate the information resource. For example, in the context of the Internet, these are abbreviated names of some protocols used to access various information resources: HTTP, FTP, gopher, telnet, and news.

User Datagram Protocol (UDP). In the *Internet* suite of *protocols*, a protocol that provides unreliable, connectionless datagram service. It enables an application program on one machine or process to send a datagram to an application program on another machine or process. UDP uses the *Internet Protocol (IP)* to deliver datagrams.

user exit routine. A user-written routine that receives control at predefined *user exits*.

user exit. A point in an IBM-supplied program at which a user exit routine can be given control.

user group. A group consisting of one or more defined individual *users,* identified by a single group name.

user mapping. Associating Enterprise Information Portal user IDs and passwords to corresponding user IDs and passwords in one or more content servers. User mapping enables single logon to Enterprise Information Portal and multiple *content servers*.

user. A person who requires the services of Content Manager. This term generally refers to users of client applications, rather than the developers of applications, who use the Content Manager APIs. In Enterprise Information Portal, anyone who is identified in the Enterprise Information Portal administration program.

utility server. A Content Manager component that is used by the database utilities for scheduling purposes. You configure a utility server when you configure a *Resource Manager* or *Library Server*. There is one utility server for each Resource Manager and each Library Server.

V

video mixing. The process of dynamically inserting or combining multiple *video objects* into a single object for distribution. An example would be the mixing of commercials and broadcast programs for satellite distribution.

video object. The data file containing a program recorded for playback on a computer or television set.

video stream. The path data follows when read from the VideoCharger Server system to the display unit.

video-on-demand (VOD). A service for providing consumers with movies and other programming almost immediately, per request.

VOD. See *Video-on-demand.*

volume. A representation of an actual physical storage device or unit on which the objects in your system are stored.

W

WAIS. See *Wide Area Information Service.*

WAV. A format to store digitally recorded sound.

Web server. A server that is connected to the *Internet* and is dedicated to serving Web pages.

Wide Area Information Service (WAIS). A network information system that enables clients to search documents on the World Wide Web.

wildcard character. A special character such as an asterisk (*) or a question mark (?) that can be used to represent one or more characters. Any character or set of characters can replace a wildcard character.

work item. In earlier Content Manager workflow and Enterprise Information Portal advanced workflow, any work activity that is active within a *workflow.*

work packet. In Enterprise Information Portal Version 7.1, a collection of *documents* that is routed from one location to another. Users access and work with work packets through *worklists.*

work state. The status of an individual *work item*, *document*, or *folder.*

work step. A discrete point in a *workflow* or *document routing process* through which an individual *work item, document,* or *folder* must pass.

workbasket. A collection of *documents* or *folders* that are either in process or waiting to be processed. A workbasket definition includes the rules that govern the presentation, status, and security of its contents.

workflow coordinator. In earlier Content Manager workflow, a user who receives notification that a *work item* in the *workflow* has not been processed in some specified time. The user is selected for a specific *user group* or upon creation of the workflow.

workflow state. The status of an entire *workflow.*

workflow. In earlier Content Manager, a sequence of *workbaskets* through which a *document* or *folder* travels while it is being processed. In Enterprise Information Portal, a sequence of *work steps,* and the rules governing those steps, through which a *work packet, document,* or *folder* travels while it is being processed. For example, claims approval would describe the process that an individual insurance claim must follow for approval.

worklist. A collection of *work items, documents,* or *folders* that are assigned to a user.

World Wide Web (WWW). A network of servers that contain programs and files. Many of the files contain hypertext links to other documents available through the network.

WWW. See *World Wide Web.*

X

XDO. See *extended data object.*

XML. See *Extensible Markup Language.*

Abbreviations and acronyms

API	Application programming interface
B2B	Business-to-business
CM	Content Manager
CSS	Cascading Style Sheet
DB2 UDB	DB2 Universal Database
DBA	Database Administrator
DDL	Data Definition Language
DDO	Dynamic Data Object
DLL	Dynamically Linked Libraries
EJFS	Extended Journaled File System
FCM	Fast Communication Manager
IBM	International Business Machines Corporation
IIS	Internet Information Service
ITSO	International Technical Support Organization
JFS	Journaled File System
JFS2	Journaled File System 2
Kb	Kilobits
KB	Kilobyte
kbps	Kilobits per second
LAN	Local Area Network
LS	Library Server
Mb	Megabit
MB	Megabyte
Mbps	Megabits per second
MTU	Maximum Tran
OCR	Optical character recognition
PID	Persistent ID
RDBMS	Relational Database Management System
RM	Resource Manager
RMI	Remote Method Invocation
SQL	Structured Query Language
SQLCA	SQL communication area
SQLDA	SQL descriptor area
TSM	Tivoli Storage Management
UDF	User-defined functions
WAN	Wide Area Network
XQPE	XQuery Path Expressions

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see "How to get IBM Redbooks" on page 490. Note that some of the documents referenced here may be available in softcopy only.

- *Performance Tuning for Content Manager*, SG24-6949
- *WebSphere Studio Application Developer Version 5 Programming Guide*, SG24-6957

Other publications

These publications are also relevant as further information sources:

- *IBM Content Manager for Multiplatforms / IBM Information Integrator for Content: Installing, Configuring, and Managing eClient*, SC27-1350
- *IBM Content Manager for Multiplatforms: Planning and Installing Information Integrator for Content*, GC27-1345
- *IBM Content Manager for Multiplatforms: Managing Information Integrator for Content*, SC27-1346
- *IBM Content Manager for Multiplatforms / IBM Information Integrator for Content: Workstation Application Programming Guide*, SC27-1347
- *IBM Content Manager for Multiplatforms: Planning and Installing Your Content Manager System*, GC27-1332
- *IBM Content Manager for Multiplatforms: System Administration Guide*, SC27-1335
- *IBM Content Manager V8.2 Performance Tuning Guide*, by Content Manager Performance Team, found in:

  `http://www.ibm.com/software/data/cm/cmgr/mp/support.html`

  Search for "performance tuning guide" under Whitepapers.

- *IBM Directory Server Version 5.1 Administration Guide* and *BM Directory Server V5.1 Installation and Configuration Guide*, found at:

  http://publib.boulder.ibm.com/tividd/td/IBMDirectoryServer5.1.html
- *IBM Content Manager for Multiplatforms Version 8.1: System Administration Certification Study Guide*

Online resources

These Web sites and URLs are also relevant as further information sources:

- The official IBM eClient support Web site:

  http://www-3.ibm.com/software/data/cm/support.html
- Reference for configuring eClient to use WebSphere's connection pooling:

  http://www-3.ibm.com/software/data/cm/pubs/cm81/eclientpooling/WASpool.htm
- Tuning garbage collection with the 1.3.1 Java Virtual Machine (JVM):

  http://java.sun.com/docs/hotspot/gc
- The EIP user group is available at:

  http://w3.ibm.com/forums/forums.htm

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Index

IBM

Redbooks

eClient 101 Customization and Integration

(1.0” spine)
0.875”<->1.498”
460 <-> 788 pages

eClient 101
Customization and Integration

Redbooks

Basic introduction to installing and using eClient

eClient customization and integration with sample codes

Special topics on information mining, Siebel and single sign-on integration

This IBM Redbook provides a basic introduction to IBM DB2 Content Manager Version 8 eClient. By providing helpful, easy-to-understand sample codes and step-by-step instructions, this redbook will help you in your next eClient integration and customization project.

We provide detailed step-by-step instructions on installing eClient, installing eClient in a WebSphere Network Deployment environment, and using eClient.

To prepare for eClient customization and integration, we introduce J2EE, servlets, and JSPs. We cover the eClient architecture and inspect a basic eClient control flow. We also provide the essential information required to start creating applications with EIP. We include sample codes to demonstrate the usages of APIs. In addition, we provide step-by-step instructions on setting up an eClient development environment and discuss design and implementation considerations.

We provide sample codes for changing the look and feel, customizing the edit attributes window, adding customized functions in the search results window, and using EIP privileges for access control. By demonstrating some of the customization with the sample codes, you should have a better understanding of what and how you can customize eClient for your business needs.

We cover e-mail integration and special topics on Information Mining Service, Siebel integration, and single sign-on integration. The installation, setup, configuration, and integration are presented with detailed step-by-step instructions. We provide detailed sample codes for enabling metadata-based data retrieval.

Finally, we provide tips and recommendations on how to troubleshoot problems, with a list of typical problems and their resolutions. In addition, we give a brief introduction on performance tuning for eClient.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6964-00 ISBN 0738499196