WebSphere. software

IBM

Administering and Implementing WebSphere Business Integration Server V4.3

Implement a business integration infrastructure

Develop and deploy solution components

Manage WebSphere BI infrastructure

Alicia Harvey
Travis Jeanneret
Thiam Cheng Lee
Rangarajan S Manavalan
Marty Trice

Redbooks

ibm.com/redbooks

International Technical Support Organization

Administering and Implementing WebSphere Business Integration Server V4.3

April 2006

SG24-6647-00

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

First Edition (April 2006)

This edition applies to Version 4, Release 3, Modification 0 of IBM WebSphere Business Integration Server.

Contents

Notices

Trademarks

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

eserver®
eserver®
Redbooks (logo) ™
^®
e-business on demand™
iSeries™
xSeries®
z/OS®
AIX®
ClearCase®
CICS®
DB2 Connect™
DB2 Universal Database™
DB2®
IBM®
IMS™
MQSeries®
OS/390®
Rational®
Redbooks™
RDN™
S/360™
SupportPac™
Tivoli®
WebSphere®

The following terms are trademarks of other companies:

Java, JavaScript, JDBC, JDK, JVM, Solaris, Sun, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows server, Windows NT, Windows, Win32, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Pentium, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

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

Linux is a trademark of Linus Torvalds 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 describes three major phases in a WebSphere® Business Integration (BI) project.

- We discuss the planning and system design for a WebSphere BI infrastructure designed to support several business integration projects. We extend the real-life scenario written for another IBM Redbook. Following planning and design, we discuss the implementation of the run-time engines available in IBM WebSphere Business Integration Server V4.3.
- The next phase is developing and testing a business integration solution within our infrastructure. The integration solution combines three run-time engines of WebSphere Business Integration Server V4.3. These engines provide for human interaction, straight-through processing, and message brokering and aggregation.
- The final phase of our WebSphere BI project involves deploying the solution into the production environment, and how to manage this solution. We address issues such as how to coordinate stopping and starting components, and troubleshooting run-time problems. We end by discussing performance tuning in WebSphere Business Integration Server V4.3.

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 (ITSO), Raleigh Center.

From left to right: Rangarajan, Marty, Thiam Cheng, Travis and Alicia

Alicia Harvey is an IT Specialist with the IBM International Technical Support Organization, Raleigh Center. Alicia has been a Systems Integrator with IBM since 1993. For the last six years she has specialized in EAI solutions using the WebSphere MQ family of products. Alicia writes extensively and teaches IBM classes worldwide on all areas of the WebSphere MQ family and Business Integration.

Travis Jeanneret is a Certified Senior Pre-Sales IT Specialist with IBM Software. Travis has eight years of experience in EAI and Business Integration solutions. He holds a degree in Computer Engineering from Kansas State University. Travis's areas of expertise include business process modeling, process Integration, and application integration.

Thiam Cheng Lee is a Senior IT Specialist with IBM Business Consulting Services in Malaysia. Thiam Cheng has eight years of experience in application integration specializing mainly in the banking and finance industry. He holds a degree in Computer Science from University Science of Malaysia. Thiam Cheng's other areas of interest include host application integration and cryptography related applications.

Rangarajan S Manavalan is a Staff Software Engineer with IBM at Burlingame, California. Rangarajan has been developing software at IBM Since 2001. For the last three years he has specialized in WebSphere Business Integration Server development. He holds a Masters Degree in Electronics and Communications Engineering from Anna University. Rangarajan's interests include application integration and distributed applications.

Marty Trice is an Enterprise Application Integration Administrator with Sara Lee Corporation, Winston Salem, NC USA. Marty has three years of experience in engineering and integrating EAI solutions on multiple platforms. He holds undergraduate and graduate degrees in Education (BS), Math Education (BS), Mathematics (MS), and Computer Science (MS) from North Carolina State and North Carolina A&T State Universities, respectively. Marty's areas of experience include WebSphere MQ, WebSphere MQ Integrator Broker, WebSphere MQ Workflow, WebSphere InterChange Server, and WebSphere Application Server. In addition, Marty is a Mathematics Instructor with the North Carolina Community College System.

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

Murali Behara
Sara Lee Corporation EAI Development

John Ganci
IBM International Technical Support Organization, Raleigh Center

Geert Van de Putte
International Technical Support Organization, Raleigh Center

Gang Chen
IBM China Development Lab

Steve Rice
IBM Americas Business Integration PoC Speed Team

IBM WebSphere Business Integration Performance Team, Austin, Texas
IBM Software Group

Chris Richardson
IBM WebSphere Business Integration Performance Team, Austin

Mike Collins
IBM WebSphere Business Integration Performance Team, Austin

Andreas Eggenschwiler
IBM Sales & Distribution, Software Sales, Switzerland

Manuel Carlos Rodriguez Alvarez-Querol
IT Specialist, Spain

Satyavani Suryadevara
Miracle Software Systems Inc., Detroit

Francois van der Merwe
IBM Sales & Distribution, Software Sales, South Africa

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 email to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HZ8 Building 662
  P.O. Box 12195
  Research Triangle Park, NC 27709-2195

Part 1

Implementing a BI solution framework

Business integration (BI) technologies have made significant advances in recent years, reaching the stage of maturity where BI solutions are implemented more rigorously. Today, BI infrastructures are designed to support integration solutions for many groups within an organization. In the past, business integration solutions were often implemented to service one single part of an organization. As a result of this, IT departments found themselves faced with islands of integration.

Now, BI infrastructures are created to host business integration solutions and to provide the foundation layer for the next layers of integration, business process management and business-to-business (B2B) integration.

1

The state of business integration technology

For many reasons, business integration (BI) is an important focus area for IT organizations. Not only does it promise great cost and efficiency benefits by automating and controlling the interactions of disparate systems whether they are existing or new, home-grown or packages, it is also seen as an important supporting technology for Business Process Management (BPM) and an enabler for business-to-business (B2B) technology.

In this chapter, we introduce the design, implementation, and administration concepts that are central to the business integration solution described later in this book.

1.1 IBM WebSphere BI Overview

In the world of business integration, there is no single solution for all situations. This is why the IBM WebSphere software platform contains several complementary technology offerings that provide WebSphere Business Integration (BI) functionality. In this book, we describe using WebSphere Business Integration Server. In addition to WebSphere MQ itself, which forms the foundation, there are four additional components that are discussed in this book:

- WebSphere MQ Workflow
- WebSphere InterChange Server
- WebSphere Business Integration Message Broker
- WebSphere Business Integration Adapters

These components are packaged together as *WebSphere Business Integration Server* in order to give customers the opportunity to choose the components that best fit each of their integration projects.

This IBM Redbook is about implementing and managing an integration solution built upon these four components.

1.2 The evolution of business integration technology

It seems inconceivable today that before the mid-1960s every new generation of computers changed the programming model so much that effectively all applications had to be rewritten or were condemned to run in emulation mode, where available.

The IBM S/360™ computing architecture promised customers for the first time that future architectures would be backward-compatible so that investments in software would be protected.

Several diverse architectures exist still, but they have evolved over a period of time and the issues surrounding the porting of code between them are well understood, if not completely solved.

Historically, applications were written to solve specific, well-delineated problems. Architects and developers gave little or no thought to a holistic application landscape that would cover the whole range of business requirements, so no need for an integrated architecture was seen. As a result, solutions evolved on a great variety of platforms. If and where integration was needed, it was usually achieved by hosting the applications on the same system and sharing files. This was no great restriction, because most applications at that time were batch-oriented and large central computers, mainframes, were the accepted technology standard. With the evolution of database management systems, the models for sharing information soon moved to this technology.

Whenever a business identified a need for information to be shared across their computing platforms, they had to use the networking capabilities of the day, which were anything but user-friendly. Protocols on all levels were proprietary, often complex, and usually not well-understood, especially for cross-platform implementations. Files remained the favorite entities to share, for several reasons:

- They worked well between applications on the same system.
- Support was available for cross-platform file transfers and file-sharing on network servers.
- Above all, most applications were still batch-oriented.

Where online processing was introduced, businesses found it more acceptable from risk and system-capacity perspectives to simply collect data during the online day (in files) and do the actual processing during nightly batch runs. This mode of operation is still prevalent in businesses.

Occasionally, though, a need for real-time, or near real-time, communication between two applications on disparate platforms would emerge. This brought several problems with it:

- Technology choices had to be made on all levels of network protocols, on both platforms. These early solutions were strictly point-to-point)
- The communications functions had to be added to the applications. This involved highly specialized programming skills, often a different skill set for each platform.
- The APIs were complex at best and, lots of exception handling and recovery logic was required in the code if there was to be any degree of success.

In a nutshell, such applications were exceedingly expensive to build. They also took so long to implement that business benefits were often realized too late to justify the development effort.

This is where messaging middleware came in, notably WebSphere MQ. By providing a simple, easy-to-use API that conceals network complexities from application programmers, WebSphere MQ made writing communication functions into application programs much easier and faster. And, because MQ was made available on so many different platforms, cross-platform communications became much easier as well.

Asynchronous communication with queues allowed for loose coupling of systems and provided reliability (if queues were kept safe on disks) without compromising the ability to communicate in real time. Transactional capabilities enabled users to build robust messaging applications that would ensure data integrity across systems. In addition, message queuing allowed for easy transition between the batch-oriented processing model of the past and event-driven, real-time processing, which is very much a requirement in today's e-business on demand™ culture. This is because message queues can serve as buffers between the new online systems and the old batch systems that will be replaced by online back-end systems sooner or later. Well-defined messaging interfaces also serve to facilitate this transition.

The use of messaging almost automatically leads to a need to architect interfaces between applications. This is not really new, because databases (and flat files) also required agreement between applications on what data was placed where and, with what meaning. However, a messaging interface adds a new dimension, because it provides for a great deal of interoperability between communicating applications. It is quite easy, for example, to replace an old application on an existing platform with a new implementation on a different platform, as long as the specific message interface continues to be observed.

Originally, any applications that wanted use WebSphere MQ had to do so using the provided API, the MQI. In many instances, however, this was not possible (as in the case of off-the-shelf packages) or it was difficult (as with some existing applications). To help in these situations, a series of components was developed that are referred to as *bridges*, *adapters*, or *connectors*. They all provide a WebSphere MQ interface for specific applications or groups of applications that are noninvasive. The bridges for CICS® and IMS™ are examples of this. The bridges enable a WebSphere MQ-enabled application to start existing transactions in these systems and receive results back.

Adapters use several techniques to interface with packages, such as database or flat file interfaces, terminal emulation, or package-specific APIs. They typically provide and expect messages in specific formats based on the application's capabilities and requirements, although some implementations provide a degree of message transformation.

At this point in evolution, inter-application programming was still needed, if only to route messages between packaged applications and to transform them according to the applications' requirements.

The promise of business integration is that ultimately any application will be able to communicate with any other application, as long as there is some meaning in such a communication. It should then also be possible to reconfigure such networks of applications to form innovative solutions without changing the applications themselves.

Enter message brokering technology. After you make all applications speak the same language on a transport level, whether by using adapters or by adding API calls to the middleware, you can look at clever ways of managing data traffic. *Message brokers* are systems that can be located centrally between applications, such as in a hub-and-spoke topology. Consider the following example.

Two applications (A and B) communicate with WebSphere MQ using a particular message interface agreed between their developers. A new package (C) is introduced. It is determined that the messages that are exchanged by A and B could be used as a data feed into C, but the message formats are not quite correct. Because C is a package and it cannot be modified, we now face modifying both A and B. If either of them cannot be changed (the package itself, an existing application with no maintenance skills available, or the owners think a change would be too risky), the project stops right there. Even if there are no inhibitors, changes such as this can be very costly and time-consuming. Our example features only three applications, but imagine a scenario in which something as central as a general ledger must be replaced. You could be looking at hundreds of interfaces.

So it would be advantageous to have a central hub where messages can be reformatted and rerouted according to easy-to-set-up rules that can be adjusted quickly to changing requirements. That is exactly the role of a message broker, in our case WebSphere Business Integration Message Broker.

WebSphere Business Integration Message Broker uses a visual construction approach for you to build message flows by wiring together a sequence of processing functions that are provided as *processing nodes.* Processing nodes can be customized for the exact function parameters at any specific point in the flow, such as the name of a queue or database to interact with, transactional behavior, or field-by-field transformations.

A comprehensive set of adapters to help integrate all applications of the enterprise and a message broker makes it easy to model the information flow between applications, Enterprise Application Integration is achieved, almost.

From the perspective of the IT infrastructure, all of the tools for enabling the flow of information between the applications are in place. But the business wants something more. The flow of data through an enterprise, or between separate business entities as in B2B, must be managed according to certain business process rules.

A classic example is a customer service situation. A customer service request can arrive through several different channels, such as a telephone call, voice message, fax, e-mail, or even a traditional letter.

Any such triggering event must be captured, categorized, and sent on its way through the organization along a predetermined path. At the same time, this event is monitored by a management system that ensures that a resolution (and customer satisfaction) is achieved within a certain time frame and that any slippage is detected, escalated, and remedied according to specific service-level parameters.

This method of inquiry resolution is called *Business Process Management* (BPM), and IBM offers WebSphere MQ Workflow for this function. It enables the analyst to model a step-by-step business process involving both applications that perform their functions unattended in the background and those that operate interactively in a dialog with a human operator. WebSphere MQ Workflow is equipped to drive and manage the process flow between steps, to interface with the applications and to assign work to the staff members according to an organizational profile. WebSphere MQ Workflow is built on WebSphere MQ technology, including WebSphere Business Integration Message Broker wherever appropriate, providing for easy and robust application integration. It also uses DB2® databases to maintain persistent state information for all process instances, so that business processes can be implemented in a fully transactional fashion even though they might well extend over many weeks.

One common denominator of a message-oriented processing environment, and a differentiator against the older, batch-oriented systems, is that it supports event-driven processing in real or near real time when appropriate. There are many business integration scenarios in which that capability is a prerequisite, such as straight-through processing, transaction synchronization spanning multiple systems, same-day value, online banking, or ordering over the Internet.

The WebSphere InterChange Server was designed to detect real-time events and use those events to drive sequences of related business activities. Thus, it choreographs the interaction between desperate systems and technologies by means of automated business processes called collaborations. The InterChange Server was built upon the premise of a canonical data model allowing for the abstraction of the business process and rules from the underlying integration technology. The collaborations act upon a common data structure (Business Objects) and when communicating with a source or target application, the InterChange server will map (translate) that common data structure to or from the application specific form of that data.

The next two steps in the evolution of integration technology, already under way, are:

- Inclusion of external systems, such as those of your trading partners, suppliers, or customers, into your process automation. This is usually referred to as Business-to-Business Integration (B2B).

- Development of a comprehensive set of standards for Web-based service infrastructures, known as Web services. Standardizing the way that services can be exposed and invoked makes such services easily accessible from anywhere on the Internet, making a huge contribution toward doing business on the Internet.

Both of these developments are beyond the scope of this book and are only mentioned here to complete our overview of Enterprise Application Integration.

1.3 Integration capabilities

WebSphere is positioned as the IBM platform for integration. There are many and varied products in the platform, but the common denominator is that they all support integration in one way or another. To help categorize this large number of products on the basis of their functionality, different phases of building an integration solution were identified. The integration capabilities of a product of the Business Integration family of products can be categorized in six activities.

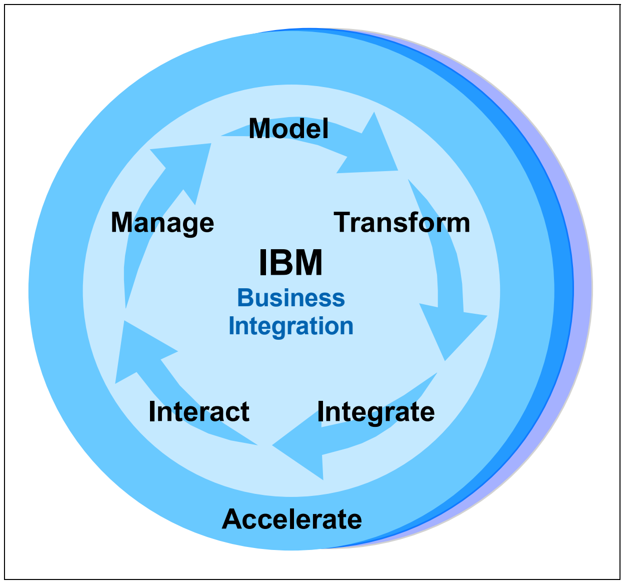


Figure 1-1 Integration capabilities

The following overview helps to position the functional offerings in the Business Integration product set and enables us to delineate their capabilities against each other in an effort to employ the correct technology for each solution.

These six types of integration capabilities have been identified:

- *Model* business functions and business processes

  During this stage, business analysts design or redesign processes quickly and graphically across people, partners, and applications. The modeled process can be simulated using what-if scenarios and optimized to obtain projected business benefits. The modeling phase also results in deployable components, without much code writing.

  The business analysts, not the IT developers, use a powerful graphical tool to define and describe a new business process. Using a standard library of processes, interfaces, and business definitions, the business analyst draws up a new process or makes changes to an existing process. The business analyst documents the business requirements in a straightforward manner, which are then be shared with colleagues in the organization. If necessary, the analyst designs forms and applies standard business objects to the forms, or creates new forms that could be used throughout the process.

  The analyst applies standard business cost for the selected elements or applies unique costs to build up a profile of the process. Using a standard test selection, the analyst then runs a set of transactions through the process to see what business results arise.

  IBM WebSphere Business Integration Modeler enables line-of-business analysts to describe a new process using graphical tools and to put into that process the necessary steps to integrate applications, people, and processes across the business or businesses. The analysts can designate resources and activities that are within the department, within the enterprise, or in any part of an extended process.

- *Transform* applications processes and data.

  Transforming applications and data is the phase where you apply tools to help you reuse existing applications in an e-business on demand environment. This is all about transforming applications, processes, and data, whether creating new business value from existing IT systems or maximizing predictability, efficiency, and quality development overall. The IBM Enterprise Transformation portfolio provides tools that enable you to:

  - Transform the user experience.

    Convert green screens to a user-friendly Web interface, improve the workflow and navigation of host applications, and provide host access through a browser or portal.

  - Transform the connectivity.

    Create Web services from CICS, IMS, or iSeries™ applications and transform existing processes into reusable, shareable business components. Use Java™ connectors to integrate existing applications with WebSphere Application Server.

  - Transform the application structure.

    Discover the unrealized business value in your existing assets and develop new applications that reuse existing code for greater efficiency and flexibility.

- *Integrate* islands of applications, processes, and information.

  Integration enables you to bring together islands of siloed information across and beyond the enterprise including:

  - Application and data sources across and beyond the enterprise
  - Structured and unstructured information from relational data stores, existing data sources, content repositories, spreadsheets, e-mail, and more

  Requirements for this type of cross-enterprise integration include:

  - The ability to define and manage the way that standalone applications work together in an end-to-end business process. These business processes also must extend to applications and data sources that reside with partners.
  - Integration also requires middleware that is capable of near real-time communications between the different applications and data sources that are involved in the business process.
  - Integration might also involve providing a common view of data and content from sources no matter what form the data is stored in, no matter where it resides.

- *Interact* with resources any time, anywhere, and with any device.

  This capability delivers personalized information that is drawn from multiple sources through diverse channels. It creates a single user experience across applications on a variety of devices. This style is implemented by using portal, host integration, and mobile device technologies, including such functions as transcoding, translation, and personalization. Federated database searches are a part of this style, as well as a consolidated view of applications that provide related information but are physically disparate and not integrated at all. This style also meets the user requirement for a unified and consolidated view of his IT resources, including such features as single sign-on.

- *Manage* performance against business measures.

  This capability allows you to take a considered look at the performance of a process over a period of time and see if that process meets the expectation of the simulation. This goes beyond traditional system management and monitoring tools because it ties operational results back to business measures. Also, you can generate reports based on real-time and historical data. The operational data and results can be used again in the model phase to improve the design and to assess initial assumptions versus actual values.

- *Accelerate* the implementation of intelligent processes.

  At all stages of the integration project, you can apply tools and templates to accelerate the implementation. With pre-built processes, pre-built connectivity, and domain or industry expertise, capabilities that are reflected in Accelerate help reduce the costs, and adapt or grow based on dynamic market conditions.

  This is achieved with the capabilities that are contained in the Accelerate category:

  – WebSphere Commerce: pre-built processes for selling, buying, and channel-management solutions

  – Process templates: also known within IBM as *Collaborations*, provided across many industries in the areas of Procurement, CRM, Order Management, Financial and Human Resources, Collaboration Foundation, Insurance, Telecommunications, Retail, and Healthcare.

  – Adapters: provided for Applications, Technology and Data Handlers, Mainframe, e-Business

Figure 1-2 on page 13 maps these capabilities on actual product offerings in the WebSphere portfolio.

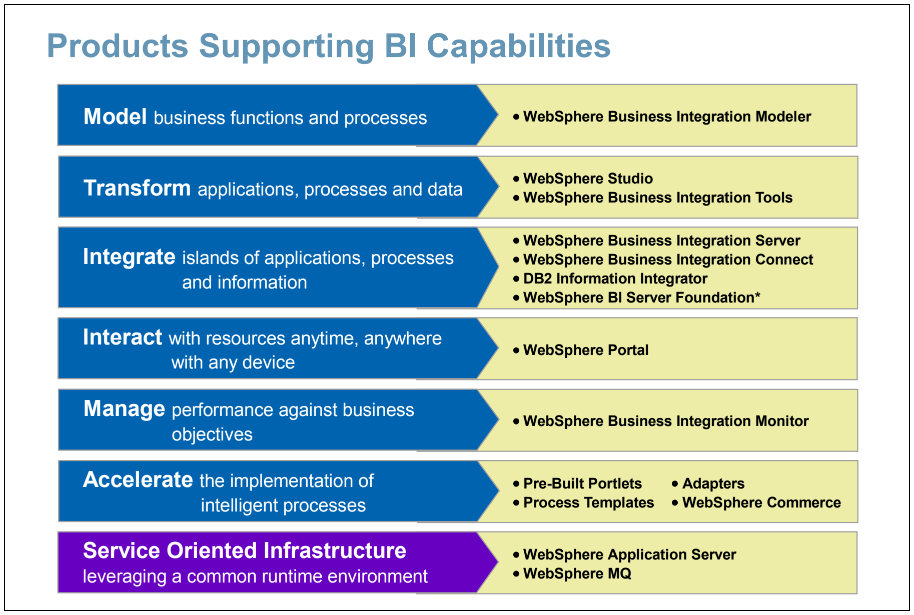

Products Supporting BI Capabilities
Model business functions and processes
• WebSphere Business Integration Modeler
Transform applications, processes and data
• WebSphere Studio
• WebSphere Business Integration Tools
Integrate islands of applications, processes and information
• WebSphere Business Integration Server
• WebSphere Business Integration Connect
• DB2 Information Integrator
• WebSphere BI Server Foundation*
Interact with resources anytime, anywhere with any device
• WebSphere Portal
Manage performance against business objectives
• WebSphere Business Integration Monitor
Accelerate the implementation of intelligent processes
• Pre-Built Portlets
• Process Templates
• Adapters
• WebSphere Commerce
Service Oriented Infrastructure leveraging a common runtime environment
• WebSphere Application Server
• WebSphere MQ

Figure 1-2 Products supporting business integration capabilities

2

Building an implementation plan

Rapid change and constant cost pressure are the drivers of most Business Integration initiatives. WebSphere Business Integration Server responds to these drivers by providing:

- Out-of-box, ready-to-deploy components
- Pervasive reuse of components, infrastructure, and skills
- Delegation of concerns that are not business-related to the infrastructure

The integration infrastructure, when it is set up, is ready to cope with one project's need and handle future requirements. The infrastructure provides availability, performance, scalability, integrity, and security for mission-critical processes.

A WebSphere Business Integration Server infrastructure has:

- Low incremental costs for implementing additional business requirements
- Low risk for new projects that depend on the infrastructure
- Short time-to-value

2.1 WebSphere Business Integration Server overview

In today's business environment, companies face constant cost pressure and shorter windows of opportunities than ever. Companies respond to these pressures by improving the efficiency and agility of their operations. This is achieved through the integration of people, processes, and applications. The IBM WebSphere Business Integration Server helps companies to achieve more effective and agile processes by enabling:

- Modeling of processes at a business level
- Transformation of applications and data
- Integration of people, processes, and applications
- Interaction with resources any time
- Managing, reviewing, and improving processes and performance
- Acceleration of implementation

Addressing these six capabilities is the heart of business integration. Some of these capabilities are provided by WebSphere Business Integration Server while it provides the required interfaces to support the other capabilities. The output of the WebSphere Business Integration Modeler can be deployed in WebSphere Business Integration Server, which generates monitoring data that is used by WebSphere Business Integration Monitor. The WebSphere MQ Workflow Web Client features can be integrated easily into WebSphere Portal Server, providing the main user interface to a WebSphere Business Integration Server solution.

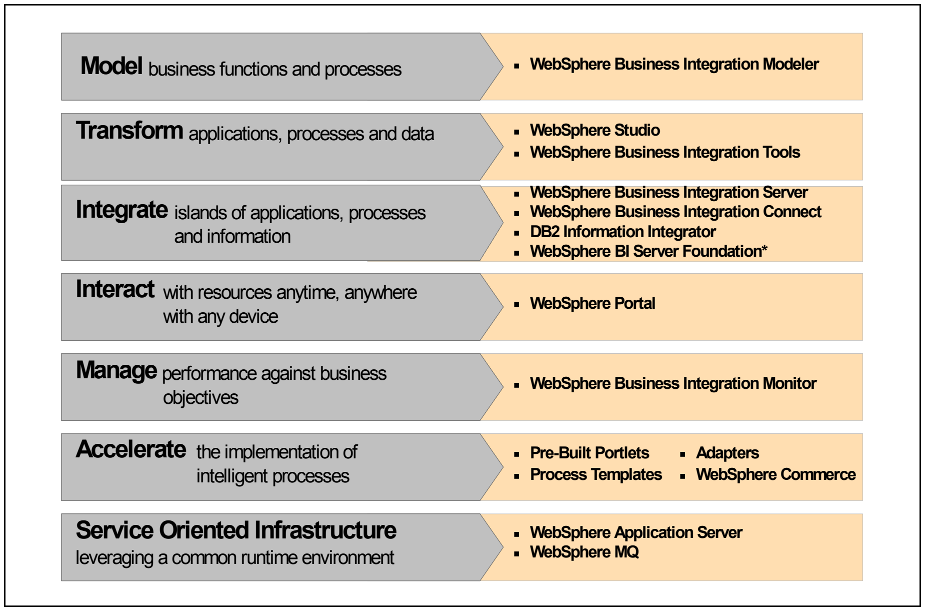

Model business functions and processes
WebSphere Business Integration Modeler
Transform applications, processes and data
WebSphere Studio
WebSphere Business Integration Tools
Integrate islands of applications, processes and information
WebSphere Business Integration Server
WebSphere Business Integration Connect
DB2 Information Integrator
WebSphere BI Server Foundation*
Interact with resources anytime, anywhere with any device
WebSphere Portal
Manage performance against business objectives
WebSphere Business Integration Monitor
Accelerate the implementation of intelligent processes
Pre-Built Portlets
Process Templates
Adapters
WebSphere Commerce
Service Oriented Infrastructure leveraging a common runtime environment
WebSphere Application Server
WebSphere MQ

Figure 2-1 The integration capabilities and WebSphere Business Integration Server

This book is about the runtime integration and connection infrastructure. The runtime components of a Business Integration system that is ready for enterprise-wide business processes need these features:

- Coordination of long-running activities that span multiple computer systems, enterprise organizations, and enterprises
- The ability to provide a simple generic model to describe and implement the interactions between potentially complex business systems.
- A high-speed business services bus that connects the process actors and mediates between data formats.
- Universal connectivity using a single adapter framework to connect:
  - Business applications such as SAP R/3 and PeopleSoft
  - Transaction systems such as CICS and IMS
  - Internal systems using standard interfaces such as RMI/IIOP, JCA, and JDBC™
  - External systems and trading hubs using Web services, EDI, and industry-specific standards.

Each of these requirements is satisfied by one component of WebSphere Business Integration Server.

2.1.1 WebSphere MQ Workflow: long-running processes

The WebSphere MQ Workflow engine enables companies to deploy long-running business processes that span multiple computer systems, organizations, and enterprises. WebSphere MQ Workflow contains the necessary software to define, build, and manage business processes.

End users interact with WebSphere MQ Workflow using Web applications, WebSphere Portal Server portlets, or Microsoft® Windows® standalone clients. Using these user interfaces, people who participate in business processes can pick up work items and add their own contributions to the business processes. Ongoing business processes can be monitored and the status of ongoing processes can be checked. Work items can be rescheduled and reassigned.

In order to integrate internal and external business systems with WebSphere MQ Workflow, processes can access straight-through business processes and business objects that are deployed to the WebSphere InterChange Server. This provides a reusable and easy method for accessing potentially complex business systems from WebSphere MQ Workflow. The complexity of integrating internal and external systems is handled by the InterChange Server. Business analysts and end users can concentrate on the business logic of the long-running process.

WebSphere MQ Workflow generates trace data about completed and in-progress processes. Detailed reports enable business analysts to assess the costs, throughput, and cycle time and to improve and redesign the business processes. This trace data is used by WebSphere Business Integration Monitor.

Development tools

A graphical representation of a WebSphere MQ Workflow process can be generated by using WebSphere MQ Workflow Build Time. Alternatively, processes that were designed by tools that support Flow Definition Language (FDL) can be imported into WebSphere MQ Workflow. WebSphere Business Integration Modeler is a process modeling and simulation application that can export process definitions to WebSphere MQ Workflow.

Runtime environment

The WebSphere MQ Workflow Runtime is the container in which the deployed processes run. WebSphere MQ Workflow Runtime uses a database management system in order to keep track of the processes' internal state. WebSphere MQ Workflow uses WebSphere MQ queues to communicate with external actors. WebSphere Application Server is used for providing a graphical user interface to end users who participate in the workflow.

Monitoring and management

The WebSphere MQ Workflow administrator uses the Administration Utility to:

- Start and stop servers
- Monitor and analyze error logs

Additionally, WebSphere Business Integration Monitor can be used to analyze trace data and to generate reports and graphical representation about the business data that is generated by the processes. This statistical data that is obtained by WebSphere Business Integration Monitor can be used by Modeler to re-engineer business processes.

2.1.2 WebSphere InterChange Server: objects and their interactions

In most enterprises, business logic and business data is distributed over multiple business systems. These business systems use many different external interfaces and data models. However, computer systems and end users must be able to access one single and simple view of the enterprise. Many business transactions change the internal state of more than one business system. Distributed and long-running units of work that affect many systems must be coordinated in order to avoid inconsistent data.

WebSphere InterChange Server implements a generic business object model that allows the description of business entities without exposing their underlying complexity. Whenever a generic business object is accessed, InterChange Server will access the underlying business systems and map their internal data structure to the generic business object.

WebSphere InterChange Server uses straight-through business processes to implement the interactions between the generic business objects. These straight-through processes (multi-step processes run end-to-end without the need for manual intervention) can span many business systems. The integrity of the distributed enterprise data can be assured by InterChange Server even if the business systems do not allow coordination of external transactions.

Graphical component design tools make it easy to design straight-through processes, business objects, and data mappings. The object design tool enables access to enterprise applications in order to obtain business object metadata and to automatically construct business objects.

WebSphere Business Integration Server can run IBM ready-to-use, industry-specific business integration solutions. These are available for following industries:

- Automotive
- Banking
- Chemical and Petroleum
- Electronics
- Energy and Utilities
- Financial Markets
- Health care
- HIPAA
- Insurance
- Life Sciences and Pharmaceuticals
- Retail Distribution
- Telecommunications

Development tools

InterChange Server uses the System Manager, an integrated development environment based on WebSphere Studio Workbench, for creating, debugging, testing, and deploying integration components. Specific perspectives in WebSphere Studio are used for designing, testing, deployment, and version management. WebSphere Studio Workbench is based on the Eclipse framework.

The design of InterChange Server is highly modular. For each of the integration components that are required for a WebSphere Business Integration solution, there is a design tool that is started from WebSphere Studio Workbench.

- The Connector Designer is used to configure the connectors, which are the components that send and receive business objects to and from enterprise applications.
- The Business Object Designer is used to design business objects, the integration components that represent the state of a business entity within the integration system.
- The Map Designer is used to graphically design maps between fields of business objects. Maps are integration components that can map one business object to another.
- The Relationship Designer is used for designing relationships, which are integration components that describe associations between business objects. For example, in one business object you might use ISO country codes, and in another business object you might use the full name of a country. A relationship links these two together so that you can perform a look-up to map the country code to the name and vice-versa.
- The Collaboration Designer is used for designing collaborations, which are straight-through business processes. The logic of collaborations can be graphically described using the Collaboration Designer.

For almost every adapter, for example the WebSphere Business Integration Adapters for SAP R/3 and the WebSphere Business Integration Adapters for JDBC, there are object discovery agents that can connect to the target applications in order to retrieve metadata to create business objects.

Additional tools can help to test and debug integration components:

- The Visual Test Connector can simulate application connectors during development. This enables development and testing of maps, business objects, and collaborations without connecting to the enterprise application.
- The Integrated Test Environment is a WebSphere Studio perspective that can be used to automate the tasks that are required for unit testing of integration components.

Runtime environment

The core InterChange Server application is a Java application that is used as the container for the integration components. It uses WebSphere MQ, JMS, and IIOP for communicating with the adapters, the development and management tools, and the monitoring application.

Monitoring and management

InterChange Server includes specific applications for monitoring an integration system:

- The System Monitor can monitor and get information messages about changes in all InterChange Server runtime components. From System Monitor, components can be started, stopped, paused, and shut down. The System Monitor is available as a standalone GUI application or as a Web-based application.
- The Flow Manager can be used to locate, view, and handle failed events. Flow Manager can list unresolved flows and obtain details about them. Unresolved flows can then be discarded, submitted, or otherwise handled.
- The Log Viewer is used to display message and trace logs.

The deployment and control of integration solutions within the InterChange Server is performed from within the System Manager perspective in WebSphere Studio. This perspective provides an overall view of development and management of integration solutions for the WebSphere InterChange Server.

Since Version 4.2.2, the InterChange Server provides other ways to pass information to the WebSphere Business Integration Monitor. For collaborations that are initiated with the WebSphere MQ Workflow connector, you can enable flow monitoring, which results in data being sent to the same event database that is used by WebSphere MQ Workflow and WebSphere Business Integration Monitor.

2.1.3 WebSphere BI Message Broker: routing and reformatting

The enterprise applications and integration infrastructure within an enterprise generate a high volume of WebSphere MQ messages. These messages contain many different destination addresses and payloads in different formats. These hard-coded destination addresses and application-specific data formats make it hard to change an enterprise infrastructure. New requirements can demand new data formats and new destination addresses. In order to satisfy these requirements, the integration infrastructure must be able to do:

- Message routing based on both address and content
- Publish/subscribe-type routing in which the subscribers subscribe to a specific message content
- Reformatting and augmenting of the content of messages
- Publish/subscribe-type routing in which each subscriber requires a specific message format.

WebSphere Business Integration Message Broker implements a message bus. Business events are routed to different destinations according to both their header data and their content. Subscribers subscribe to messages with a specific content. WebSphere Business Integration Message Broker changes and removes content and accesses databases in order to add content or to write it to database tables. Subscribers can request a specific content format.

WebSphere Business Integration Message Broker can access message queues and databases as part of one logical unit of work. This ensures that no message is lost and that no database is updated without the corresponding messages being sent as requested.

WebSphere Business Integration Message Broker uses a graphical user interface to design message flows, which describe message routing and transformation. Each step in the flow is described by a node. Data is accessed and stored in data access nodes. Extended SQL is used to transform data in transformation nodes. Business logic is described by Compute nodes. These Compute nodes are programmed in ESQL.

Development tools

The latest version of the broker product, Version 5, introduces a development toolkit that is based on WebSphere Studio Workbench. Developers use the Broker Application Development perspective in WebSphere Business Integration Message Broker to design solution components. This toolkit provides editors and views to:

- Develop, modify, debug, and deploy message flows.
- Develop, modify, and deploy message sets. *Message sets* describe the representation of business data payload of WebSphere MQ messages.
- Define broker topologies.
- Export and import resource definitions.

Runtime environment

The runtime environment of WebSphere Business Integration Message Broker consists of two distinctive components:

- The configuration manager, which manages a broker domain. A *broker domain* is a collection of one or more brokers that run on one or more platforms.
- The broker, which provides one or more runtime containers within which message flows are executed.

The configuration manager coordinates and authorizes configuration changes of the components that are deployed within the broker domain. This includes adding and removing brokers, deploying message flows, and deploying message sets.

In previous versions of the product, the configuration manager played a crucial role in the development process as well, because resources were checked in and checked out of the configuration database. This team development concept has been dropped in the latest version. Now the product relies on team development tools that plug into the Eclipse development environment under WebSphere Studio. An example of such a version control product is Rational® ClearCase®.

The broker is the container of the runtime components of WebSphere Business Integration Message Broker. The formal name for a runtime container is a *data flow engine* or *execution group*. It participates in transactions and performs transformation and routing as described by the integration components that are deployed to that data flow engine. Transactions are coordinated by WebSphere MQ.

Monitoring and management

The main management tool is the Broker Administration perspective in WebSphere Studio. Within this perspective, the broker administrator can assemble and deploy integration projects. The administrator builds a broker archive that contains message flows and message sets. This broker archive is then added to an execution group within a broker.

The actual deployment operation is a process that runs asynchronously between the configuration manager and the broker. Success or failure is reported back to the administrator with the Event Log view within WebSphere Studio.

Runtime errors are normally reported to a system log. For Windows platforms, this is the Event Viewer, which is part of the operating system.

2.1.4 WebSphere BI Adapters: connectivity

An enterprise IT infrastructure consists of enterprise systems such as SAP and PeopleSoft, of transaction systems such as CICS and IMS, and of computer systems that use specific technologies such as SOAP, RMI/IIOP, or JDBC. Each of these protocols uses its own format to encapsulate the data that is sent.

WebSphere Business Integration Adapters provide one single framework for accessing each of these data formats and systems. The adapter consists of two components: the adapter agent maintaining the connection to the application and the adapter controller maintaining the connection to the WebSphere Business Integration system. Agent and controller communicate using WebSphere MQ, JMS, or IIOP. The adapter agent component ensures that no business event is lost, even if connectivity to the WebSphere Business Integration System is interrupted. Application-specific components extend the business application in order to detect business events within the application. If there is a WebSphere Business Integration system that subscribes to the specific business event, the event is sent to the integration system. Possible integration servers include:

- WebSphere InterChange Server
- WebSphere Business Integration Message Broker
- WebSphere Application Server Enterprise Edition

The adapter controller function is specific to the use of WebSphere InterChange Server as the integration server.

Development tools

WebSphere Business Integration Adapters configurations can be created using the Connector Configurator. Additionally, the Business Object Designer with the object discovery agent, if available, can be used to generate application-specific business objects. The XML schemata that describe the business objects can be imported into WebSphere Business Integration Message Broker as a new message set if the adapter connects to WebSphere Business Integration Message Broker.

Both the Connector Configurator and the Business Object Designer integrate with WebSphere Studio.

Runtime environment

The WebSphere Business Integration Adapters consist of an adapter agent component that connects to the application and, if InterChange Server is the integration broker, of a adapter controller component that is deployed in the InterChange Server. The agent component runs as a standalone Java application. It can either run on the integration server platform or on the application platform. On Windows platforms, you can configure the adapter to run as a Windows service.

Monitoring and management

WebSphere Business Integration Adapters can be monitored and managed using the System Monitor application, which is part of InterChange Server. The System Monitor component offers the options to start, stop, suspend, resume, and shut down the adapter agent and controller. It is integrated in the System Manager and it is also available as a Web application.

For adapters that use WebSphere Application Server as its integration server, these management and monitoring functions are integrated in the WebSphere Application Server administrative console.

For problem determination, the agent and broker trace and log files provide additional information about the state of the adapters.

2.1.5 Base components

WebSphere Business Integration Server exploits the capabilities that are provided by the WebSphere Platform and by the database management system:

- WebSphere MQ provides assured, once-and-only-once delivery of messages between the loosely coupled components of the WebSphere Business Integration Server. It is the JMS provider of WebSphere Application Server. WebSphere MQ can be set up in a high-availability configuration in order to satisfy specific availability requirements.
- WebSphere Application Server is a J2EE–compliant application server which is used to provide a Web user interface for end users and administrators of the WebSphere Business Integration system.
- WebSphere MQ Workflow, WebSphere InterChange Server and WebSphere Business Integration Message Broker use a database management system, for example DB2 or Oracle, for tracking in-flight business processes and storing deployed solution components.

2.1.6 Bringing it all together

Each of the components of WebSphere Business Integration Server is responsible for its own part of the total solution. The advantages of such a layered architecture are:

- Reuse of components: Change of business processes typically are more frequent than changes of the underlying business objects. All business processes that involve the same business object can share the components for implementing the business object. This makes adapting business processes to changed needs fast and easy.
- Separation of concerns: A business analyst who designs the workflow for processing a sales order does not have to know how such sales orders are implemented in the underlying business systems.
- Exploiting each component’s strength: Each component of WebSphere Business Integration Server is designed for solving one part of the Business Integration problem domain. A combination of WebSphere Business Integration Server components can exploit each component’s particular strength.

The total solution is shown in Figure 2-2 on page 26.

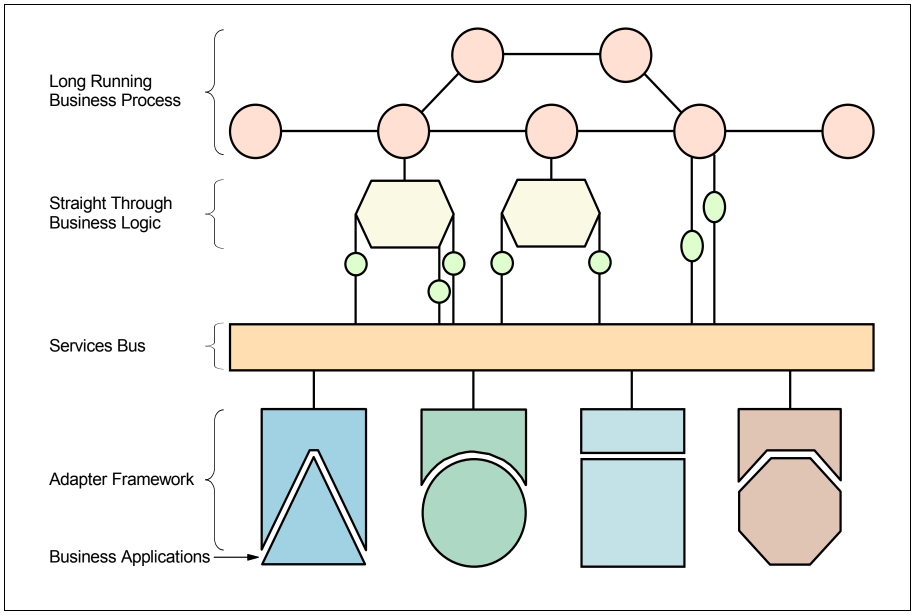


Figure 2-2 WebSphere Business Integration Server components working together

2.2 Business requirements

Obviously, an integration system must implement the use cases that are provided by an integration project's sponsor. However, the value of an integration infrastructure for an enterprise is much higher if the infrastructure is ready for the requirements of future projects. Such an infrastructure can minimize risk, incremental costs, and time-to-value for implementing additional requirements. An integration system's agility, its ability to integrate existing systems, and its ability to be monitored easily by the owners of the business processes each contribute to the integration system's value.

2.2.1 Implementation of use cases as required

Typically, use cases are provided by business analysts. They represent the functionality of an integration system from an end user's point of view. Often, use cases are provided as written descriptions of a user's or a business system's interactions with other users and other business systems. Describing and analyzing a use case is especially easy if WebSphere Business Integration Modeler is used to model the business process. The process can be graphically composed by a business analyst and simulated in WebSphere Business Integration Modeler in order to find errors, bottlenecks, and inconsistencies.

Costs of existing and new processes can be compared, and return on investment (ROI) of the integration project can be calculated. Finally, the representation of the business process can be deployed to the individual runtime components of WebSphere Business Integration Server. This means that the output that is generated by WebSphere Business Integration Modeler is imported in the development tools specific for a runtime component to allow for further development by IT specialists.

2.2.2 Agility

The solution must be easy to extend and adapt to future needs. In order to be easy to extend and adapt, it must provide:

- The ability to reuse existing integration components in new business scenarios
- The ability to deploy out-of-box components
- The scalability to cope with additional workloads
- The ability to change and add components to a running system without interrupting already deployed business integration systems

2.2.3 Ability to integrate existing services

Low costs, low risk, and short time-to-value of an integration system is achieved by pervasive reuse of components. Services that are provided by business applications such as SAP and PeopleSoft and by transaction system such as CICS and IMS can be accessed easily by the integration system. External services can be accessed using EDI and Web Services protocols. GUI tools that can access services metadata make the development of adapter components for existing business systems easy.

2.2.4 Business monitoring

After having deployed a business process to the WebSphere Business Integration Server, the business owners of the deployed processes must be able to monitor processes as they execute. WebSphere Business Integration Monitor enables business analysts to inquire on the state of work-in-progress items and to take corrective action by reassigning, reprioritizing, and rescheduling them. This enables them to:

- Check compliance with service level agreements.
- Access up-to-date process information to make operational decisions.
- Generate reports and statistical analysis from process data.

The insight into the running business processes that are obtained by business monitoring enables a business analyst to optimize business processes and to adapt them to changing needs.

2.3 Quality of service requirements

A business integration solution must satisfy the business requirements and provide the necessary qualities of service. Performance, security, and availability must be adequate for a given scenario. The integration solution's ability to be integrated into an operational IT environment strongly influences its total cost. A system that is easily integrated into an operational IT environment can be monitored by the IT operations staff. Its performance and availability correspond to its importance for the enterprise. The integration system must provide the security that is required for a system that accesses critical business applications. The system's ability to be managed directly affects its total costs.

2.3.1 Performance

The performance of the integration system affects the response time that end users see and the amount of computing resources that are needed in order to provide a specific service level. The end user response time influences the value of the integration system while the necessary computing resources influence the costs of the integration system. Value and cost together define the system's return on investment.

The response time is affected by:

- The response time of the enterprise applications involved
- The time that WebSphere MQ messages take to reach the integration system
- The performance of database access
- The time that a specific component of the integration system takes to digest the message

Flight time of WebSphere MQ transactions and database transactions strongly depend on the computer networks that are used.

The business logic of most integration systems is lightweight. In most cases, enterprise system response time is the most significant contributor to total response time.

2.3.2 Availability

The integration system must provide the availability that is required by the specific business process. Each component of the integration system including networking infrastructure can affect an integration system's availability. Redundant components can provide higher availability. However, they also affect the costs. The cost of downtime can be a significant contribution to an integration system's total cost. It directly affects the system's ROI.

Availability is especially critical if an external trading partner could be affected by downtime and if service level agreements are involved. In such cases, the costs of downtime can dominate the ROI.

2.4 System design for our scenario

This section describes the system design that resulted from our business requirements, the quality of service requirements, and the constraints. Our design principles were:

- Three-tier architecture

  Our system design uses the well-established separation of the total solution into Web-client tier, application-server tier, and database-server tier.

- Separation of the main components

  The advantage of locating the main components on separate computer systems is that each component could be installed, configured, and operated separately.

- Single central database management system

  We used one single, central database management system on one server. The advantage of this setup is that the database administrator has one single system to set up, manage, and tune.

Figure 2-3 on page 31 summarizes the interaction between the different components and systems. On the left, it shows users of WebSphere MQ Workflow Buildtime that require access to the common Buildtime database. Typically, you will have more than one installation of WebSphere MQ Workflow Buildtime within an organization. In the middle, we see runtime servers for WebSphere MQ Workflow and WebSphere Business Integration Message Broker. While only one server is shown for each runtime, adding a second workflow server is easy. Additional workflow runtime servers use the same shared workflow database. Clustering workflow servers in an MQ cluster is a relatively straightforward process, because the workflow configuration tools have built-in support for setting up workflow clusters and MQ clusters.

End users of a workflow server typically use a browser to interact with processes that are running in the workflow server. A WebSphere Application Server ties the browser to the workflow servers. A workflow system also requires a management interface, which relies on MQ client interactions to manage the workflow systems. Management tasks include deployment of new workflows, stopping and starting workflow runtime components, accessing logs, and runtime error information.

The WebSphere Business Integration Message Broker runtime machine performs a dual role. It is both the host of the configuration manager and the host of the single broker in our broker domain. The configuration manager also relies on a database to store information about solutions that are deployed to the different brokers in the domain. Note that it is not required to run the configuration manager on the same machine as a broker. A broker can be created on one or more servers and on one or more platforms. Only one configuration manager is required to manage the entire broker domain.

Adding additional WebSphere Business Integration Message Brokers requires a separate queue manager to support the broker. Multiple brokers can share the same database and tables, because every table has the broker UUID as a key column. Multiple brokers can be added to a single MQ cluster, if required.

Typically, a broker solution has no real end-user interface, only an interface for developers and administrators of the broker domain. Message flows run without user interaction (or, at the minimum, users are not aware that their system interactions include a message flow). The users who get involved with a broker solution are developers and administrators. Both types of users use the Broker Toolkit plug-ins of WebSphere Studio. This requires the use of MQ client to interact with the queue manager of the configuration manager. Setting up multiple instances of the Broker Toolkit for either administrative or development purposes is easy to do. However, the diagram in Figure 2-3 on page 31 does not address a possible shared repository for the developers, such as Rational ClearCase. Setting up such a team development environment is not within the scope of this redbook.

Finally, on the right side of Figure 2-3 on page 31, we see the InterChange Server and one or more users of the System Manager perspective and related tools. The InterChange Server relies again on a database for storing information about deployed solutions and information about active interactions and events. To manage an active system, an administrator can use a Web-based application deployed on WebSphere Application Server. Or he can use the Studio-based tools. Both rely on a connection to the ORB (Object Request Broker) that gets installed as part of the InterChange Server. Note that for Web-based management tasks, only WebSphere Application Server has to have such a connection to the ORB. The end user simply uses a browser to drive the Web application and does not require a connection to the ORB.

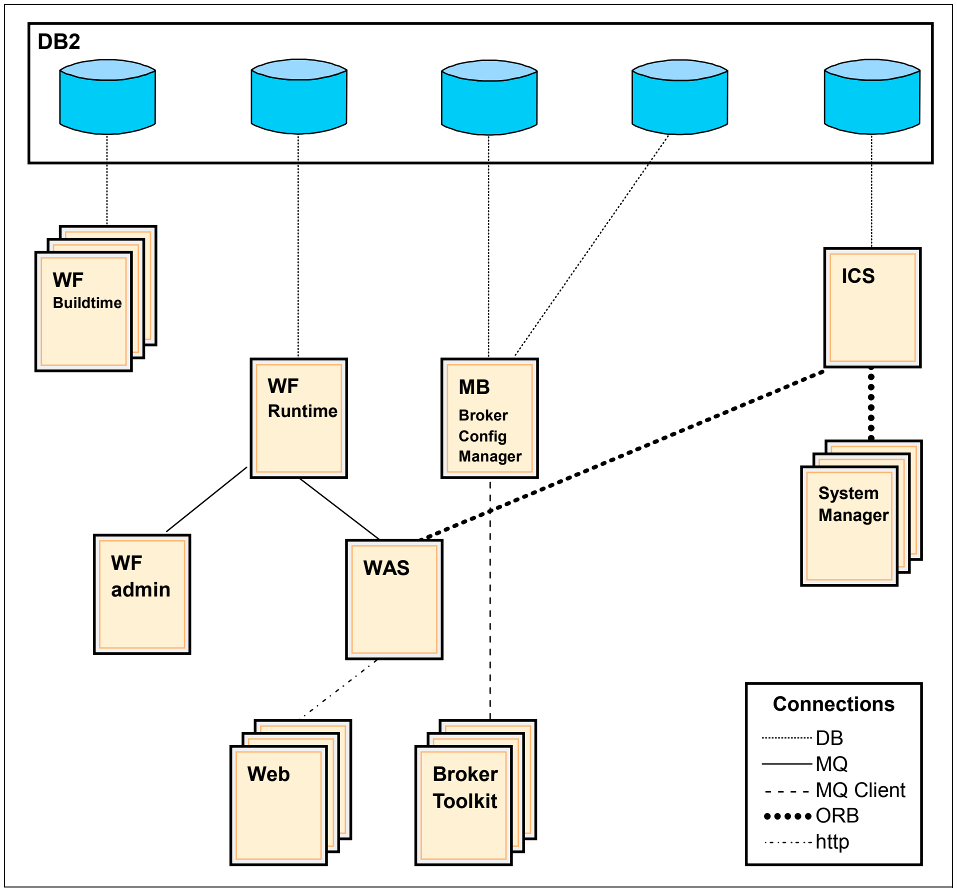


Figure 2-3 Runtime topology

2.5 Planning considerations

Before all, implementing a WebSphere Business Integration infrastructure starts by reviewing hardware and software requirements. Up-to-date information is available online at:

`http://www-306.ibm.com/software/integration/wbiserver/requirements/`

In this book, we briefly discuss the installation steps and the options that are selected for the environment shown in Figure 2-3. However, more generic installation information is available in the product documentation, which can be reviewed here:

`http://publib.boulder.ibm.com/infocenter/wbihelp/index.jsp`

The first product-installation task is checking the prerequisites and implementing missing prerequisites if necessary. While, you can find this information in the product documentation, it is presented here for your convenience.

When additional fixes or CSDs are required, you can obtain them by accessing the IBM Support Web site at:

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

A consistent naming scheme makes both installing and operating the solution much easier. Before installation starts, names for resources should be defined. These include:

- Database names
- Queue manager name
- Queue names
- InterChange Server names
- User names
- Passwords

2.5.1 WebSphere MQ Workflow

Figure 2-4 on page 34 shows the topology layout that we use in this book for the WebSphere MQ Workflow infrastructure. It consists of three server machines:

- A dedicated database server
- An application server
- A workflow server

Figure 2-4 on page 34 also shows the development client and the runtime client in a browser.

Workflow server infrastructure

The workflow server utilizes the services of a queue manager to communicate with outside components and uses a database to store process templates and process state date.

When creating this queue manager, you must decide about several parameters and choose names for several objects. A queue manager has a name and belongs to an MQ cluster. This cluster can be an existing cluster or a newly defined cluster where this queue manager is the one and only member. A queue manager usually has a listener component; thus we must decide about the protocol and any protocol parameters, such as a port number.

The configuration tool also defines an MQ cluster receiver object, which is an object that is published to the cluster and that provides other members of the cluster with sufficient information to communicate with the workflow queue manager. As a result, the configuration tool asks you for the host name of the workflow server so that it can create this cluster receiver object. The workflow system uses several queues with names that have a fixed prefix. Finally, decide about the type of logging for the queue manager and the location of the logging files. The queue manager acts as the transaction coordinator, so you should use linear logging for this queue manager. The location of the log files should be on a dedicated and fast disk subsystem. To use this destination for MQ logging, you must update the default log directory for WebSphere MQ because the workflow configuration tool does not provide this option. To update the logging default, use the application IBM MQSeries® services.

The workflow server uses a database. As a result, during the configuration of the workflow system, we must provide several parameters. The database has a name and will be created on a remote machine, so we need information such as remote instance name, remote host name, and communication parameters to set up a database-client connection. The database connection must be performed with the authority of a user ID that is defined on the database server. Therefore, you must provide the name and password of this user ID and make sure that this user ID has sufficient authorities to work with the database instance.

When creating a workflow configuration, you store all provided parameters in a profile, which is also called a *configuration identifier*. The configuration tool usually generates one for you, but you might consider using a selected name. A workflow server belongs by default to a server system, and a server system belongs to a server group. You can define multiple servers on a single server system. A server system can be considered as a single physical machine. Both concepts, the server group and the server system, are named objects. Those names are again provided during the configuration of the server.

To assist in building any piece of the WebSphere Business Integration Server infrastructure, you should prepare a table that lists the values that you must provide during installation and setup. Table 2-1 on page 39 shows a sample table for the creation of the workflow server.

Web Client infrastructure for WebSphere MQ Workflow

The Web Client feature is a Java application that is deployed in an application server, such as WebSphere Application Server. (Setup of a multi-server Web infrastructure is a topic in itself.) For our scenario, one Web server is sufficient. The Web Client feature usually communicates with its own queue manager to the workflow queue manager. Technically speaking, the Web Client queue manager will join the MQ cluster that is created during the setup of the workflow server itself. As a result, when configuring the Web Client feature, you must know values of some parameters listed in Table 2-1 on page 39. Besides that, you must choose a name of the queue manager on the Web server machine. To enable the MQ communication, the configuration tool asks for the host name and port number so that it can create an MQ listener and a cluster receiver channel. The Web Client feature will be accessed using a certain root URI, which you can choose during the contradiction. All of this information is stored in a profile for which you can provide a name. Table 2-2 on page 40 lists these parameters and the values that were used for our scenario.

Development infrastructure for WebSphere MQ Workflow

The Buildtime tool uses a database to store process definitions. Again, this database is created on a remote database server so that process modelers can share the same repository. Thus, during the creation of the WebSphere MQ Workflow profile on the development machine, you provide database parameters similar to what was required for the runtime database and information about the workflow runtime system for which this development environment will be used. Table 2-3 on page 41 lists the Buildtime-specific parameters.

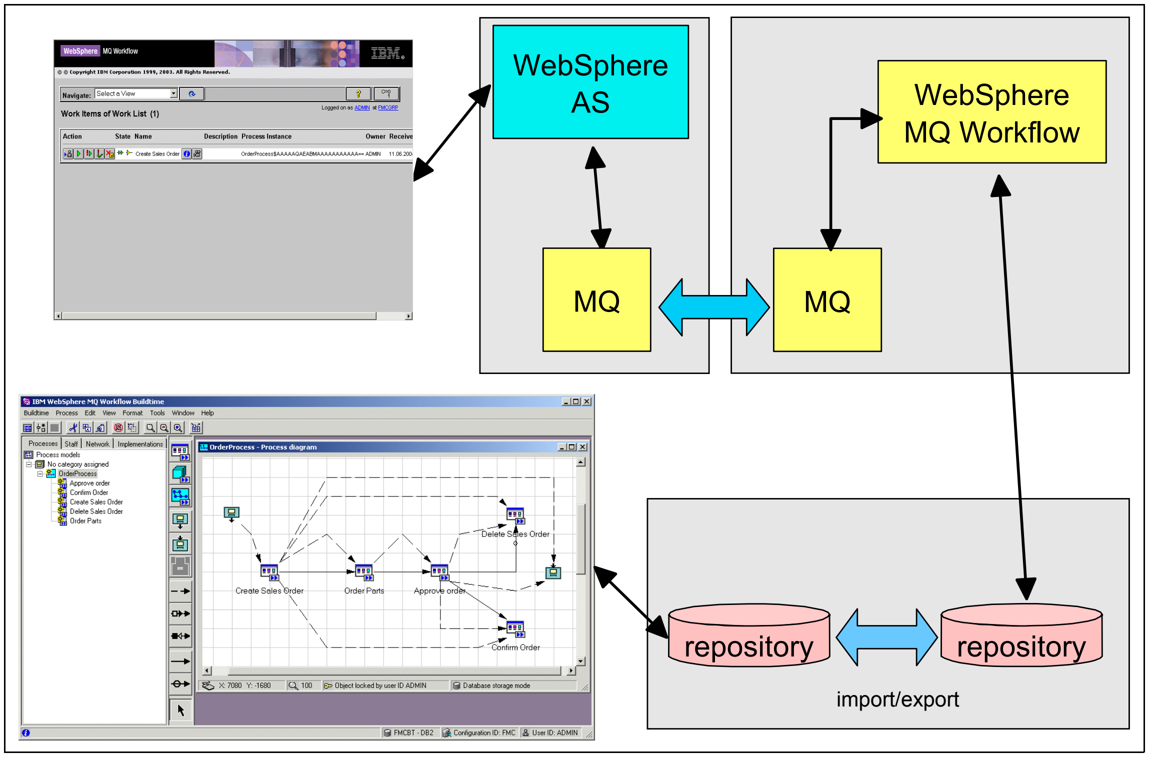


Figure 2-4 System infrastructure for WebSphere MQ Workflow

This setup enables the workflow environment to start small and grow without pain. It also allows for very easy scaling options. Depending on the workload, the number of users, or the workflow complexity, the relevant server component can be upgraded.

If the database server becomes the bottleneck of the system, it can be upgraded independent of the workflow server. Adding additional workflow systems also is easy. Such an additional workflow server will have its own queue manager that is part of the same cluster and uses the existing runtime database.

The MQ workload that is generated by workflow clients is separated on the Web server machine. When using traditional GUI workflow clients, we used an MQ client connection for each client on the workflow server. Adding more users can then have a significant impact on the performance of the server. In our setup, all workflow clients are routed through a single MQ channel pair, reducing memory usage and CPU usage for MQ tasks on the workflow server.

Because the Web server acts as a client concentrator and uses MQ cluster channels to communicate with the workflow server, we achieve a simple load-balancing between the workflow servers. As such, it also increases the availability of the workflow system. If one server is down, the Web server queue manager will route new requests to the available workflow servers.

Attention: The fail-over that is part of WebSphere MQ clustering relies on the detection of the unavailability of the queue manager. If the workflow server is down but the queue manager is still running, then messages will still be routed to the unavailable workflow server.

2.5.2 InterChange Server

Figure 2-5 on page 37 shows the topology layout that we use for the InterChange Server infrastructure in this book. It consists of three server machines:

- A dedicated database server
- An application server
- An InterChange Server

Figure 2-5 on page 37 also shows the development and management client.

InterChange Server infrastructure

The InterChange Server uses the services of a queue manager to interact with connectors. It also uses a database to store deployed solutions and runtime state data. We use a single database in our scenario, but it is easy to split repository information from runtime information in separate databases. Another component of the InterChange Server infrastructure is a name server or object request broker (ORB). It is used as a central naming service from which other components obtain information to connect to the actual InterChange Server. For example, during the installation of the development and management client, the installation program will ask about the host name and port that is used by the name service component.

To create the queue manager, you can use a script that is provided by the product or create it yourself using the WebSphere MQ tools. In either case, decide about the usual MQ parameters for a queue manager, such as its name, logging parameters, and listener port number. Also, an MQ client connection is usually created to support connectors that use MQ as the transport mechanism. The standard name for this channel is CHANNEL1, but it is a good idea to customize its name.

The database for the InterChange Server is mostly created and customized outside the InterChange Server tools. During setup of the InterChange Server, provide the name of the database, the type or vendor of the database system, and how many connections the InterChange Server can make. To make a connection, the InterChange Server has to know the name and password of an authorized database user ID. The InterChange Server itself has a name, as well.

The location of the connectors versus the InterChange Server is another design point. In the general case, you can deploy in three different ways:

- The connector is installed on the InterChange Server machine.
- The connector is installed on the application platform.
- The connector is installed on a separate machine.

Several factors play a role in deciding for one or the other option. For example, your application might be running on a platform for which there is no connector available. (Not all connectors are available for every possible platform.) Also, some applications do not tolerate an integration solution on their servers. For installations where the resource usage of the connectors is significant, it is probably better to deploy the connector on a separate platform. When the integration workload is light, installing the connectors on the same machine as the InterChange Server is probably the better solution.

System Monitor Web application

When installing InterChange Server on a machine that already has WebSphere Application Server, the System Monitor Web application is deployed automatically. However, to distribute the workload, the Web server is usually on a separate machine. The installation of the System Monitor as a Web application requires several modules and scripts. It also requires the host name of the InterChange Server machine and the port number that is used by the name server.

Development and management client

The infrastructure requirements are the same for the development client and the management client. During the installation, you must provide the host name and port number so that the System Manager tool can connect to the name server.

Table 2-4 on page 41 lists the important parameters and values that we used when building our infrastructure.

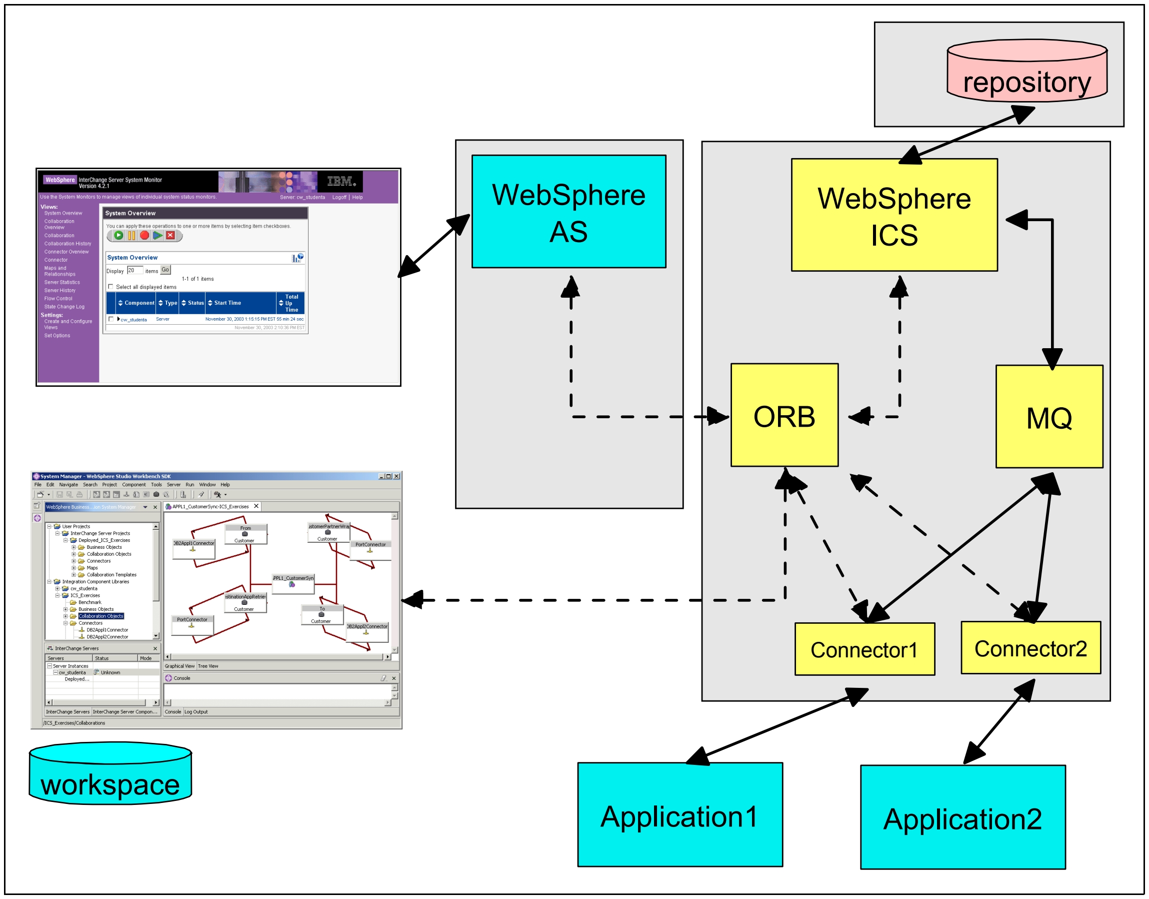


Figure 2-5 System infrastructure for WebSphere InterChange Server

2.5.3 WebSphere Business Integration Message Broker

Figure 2-5 depicts the topology layout that we use in this book for the WebSphere Business Integration Message Broker infrastructure. It consists of two server machines:

- A dedicated database server

- A server shared by the broker and the configuration manager

Figure 2-6 on page 39 also shows the development and management client.

The WebSphere Business Integration Message Broker product has two main components: the broker domain, which is a collection of brokers, and the configuration manager. Brokers are available for several platforms, while the configuration manager is only available for Windows. While the configuration manager is not a named object, the broker itself has a name, which you give it.

Both components rely on a database to store information about the broker domain and deployed components. As with the database server for our other runtime components, we use remote databases. The databases must be named and they require an authorized user ID for access.

Both the broker and the configuration manager use a queue manager to interact with each other and with the Broker Toolkit. These queue managers are named objects that must be able to communicate with each via either traditional MQ channels or MQ cluster channels. Thus, we must create two MQ listeners and need therefore two port numbers.

Note: With the broker and the configuration manager running on the same machine, we could use the same queue manager for both components. However, to make the scenario and scripts more generic, we used separate queue managers.

The broker and the configuration manager are running as Windows services. However, they do have their own user ID, not simply the system user ID. Therefore, before creating these two components, you must define one or two user IDs that can be used for Windows services and that are members of the mqm group.

The Broker Toolkit communicates with the MQ client interface to the queue manager of the configuration manager. Therefore, we ensure that the user ID of the broker developer is known and authorized to use MQ facilities on the broker server. The requirements for MQ and MQ authorizations are highly dependent on the type of Windows domain infrastructure that is in place and on whether you want to use it for securing connections between the Broker Toolkit and the configuration manager. The type of security infrastructure is passed to the configuration manager during its creation.

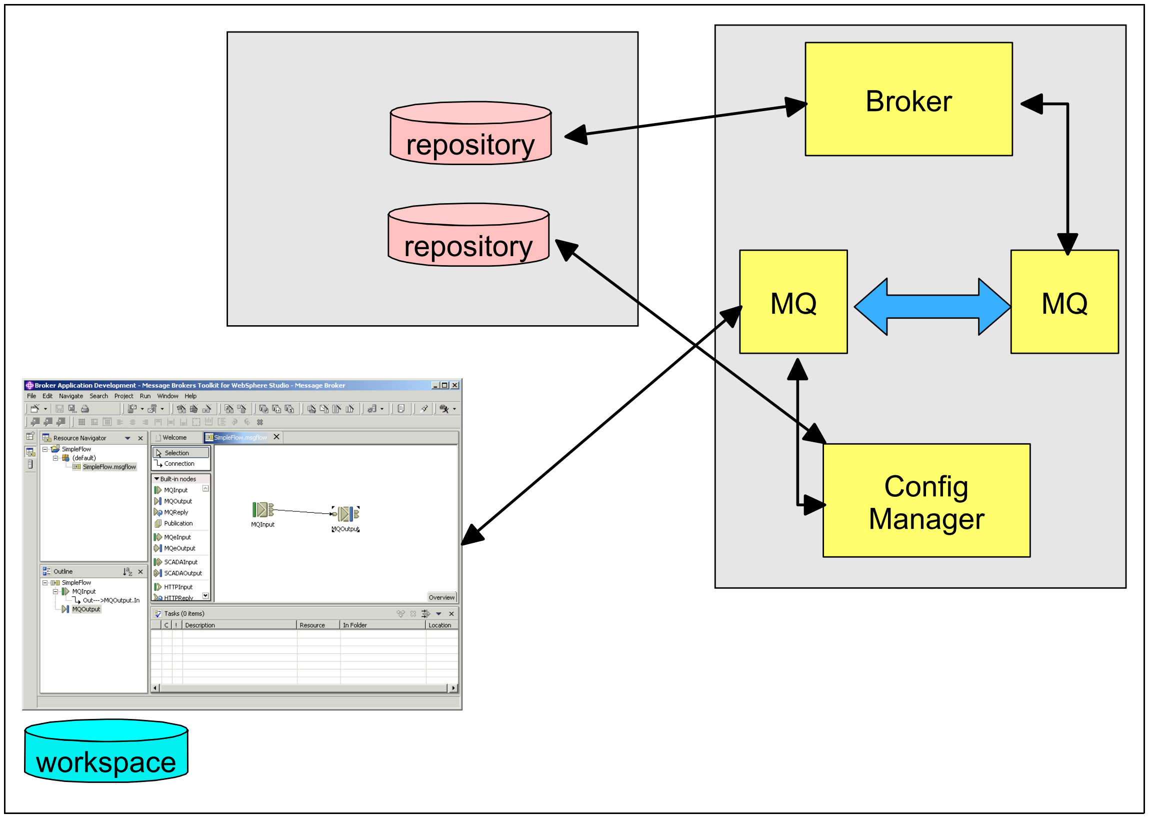


Figure 2-6 System infrastructure for WebSphere Business Integration Message Broker

2.6 Planning documents

This section provides several tables that we used during the implementation of our business integration infrastructure. While Table 2-1 through Table 2-5 on page 42 are not complete for every possible implementation of development, management, and runtime components, they do offer an idea of the settings that you need or should consider when building each component.

Table 2-1 WebSphere MQ Workflow Runtime component

Component	**Our selection**	**Reader selection**
TCP/IP address or host name	wbiwf	
Queue Manager Name	WFQM	

Component	Our selection	Reader selection
Queue Manager TCP/IP Port	5010	
Queue prefix	FMC	
Type of MQ logging	Linear	
Log files location	WebSphere MQ default	
MQ cluster name	FMCGRP	
Configuration identifier	FMC	
Configuration administrator	fmc	
Database server host name	wbidb	
Runtime Node host name	wbiwf	
WebSphere MQ Workflow Runtime database name	FMCDB	
Runtime database location	d:\WMQWF\rt_db\wbidb\FMCDB	
Runtime database logging	d:\WMQWF\rt_db\wbidb\FMCDB	
DB2 user ID to access Runtime database	wbiadmin	
Password for DB2 user ID	sab414r	
DB2 service name (port number)	50000	
System group name	FMCGRP	
System name	FMCSYS	

Table 2-2 WebSphere Application Server

Component	Our selection	Reader selection
Application Server Queue Manager Name	ASQM	
Queue Manager Port	5010	
Web server host name	wbiwas	
Web Client URL	MQWFClient	

Table 2-3 WebSphere MQ Workflow Buildtime component

Component	Our selection	Reader selection
Configuration Identifier	FMC	
Database server host name	wbidb	
WebSphere MQ Workflow Runtime database name	FMCBTDB	
Buildtime Database Location	d:\WMQWF\bt_db\wbidb\FMCBTDB	
Buildtime Database Logging	d:\WMQWF\bt_db\wbidb\FMCBTDB	
DB2 user ID to access Runtime database	wbiadmin	
Password for DB2 user ID	sab414r	
DB2 service name (port number)	50000	

Table 2-4 InterChange Server component

Component	Our election	Reader election
Host Name	wbiics	
InterChange Server name	ICS	
Queue Manager Name	ICS.queue.manager	
Queue Manager Port	1414	
Channel	CLIENTS.ISQM	
Logging	Linear	
Primary files	61	
Secondary files	2	
Dead letter queue	DLQ	
LogBufferPages	17	
LogFilePages	2048	
Admin User ID	admin	
Admin Password	null	

Component	Our election	Reader election
Database Driver	DB2	
Database Max Connections	120	
Database name	ISDB	
Database login	wbiadmin	
Database password	sab414r	
Database Port	50000	

Table 2-5 WebSphere Business Integration Message Broker

Component	Our election	Reader election
Broker Name	BROKER	
Broker Queue Manager Name	BKQM	
Broker Queue Manager Port	1414	
Broker Admin User ID	bkadmin	
Broker Admin Password	sab414r	
Broker Database	BKDB	
Broker Database User ID	wbiadmin	
Broker Database Password	sab414r	
Configuration Manager Queue Manager Name	CMQM	
Configuration Manager Queue Manager Port	1415	
Configuration Manager Admin User ID	cmadmin	
Configuration Manager Admin Password	sab414r	
Configuration Manager Database	CMDB	
Configuration Manager Database UserID	wbiadmin	
Broker Database Password	sab414r	

3

Implementing the runtime components

This chapter discusses the installation and configuration of WebSphere MQ Workflow, WebSphere InterChange Server, and WebSphere Business Integration Message Broker, which are the main runtime components of WebSphere Business Integration Server.

3.1 WebSphere MQ Workflow installation and configuration

In this section we describe the installation and configuration for WebSphere MQ Workflow Runtime, Buildtime, and client. We also discuss simple validation steps to ensure that everything is working before proceeding to the next component.

Prerequisites: WebSphere MQ and DB2 must be installed and configured prior to beginning WebSphere MQ Workflow installation and configuration.

Figure 3-1 shows the topology for our installation.

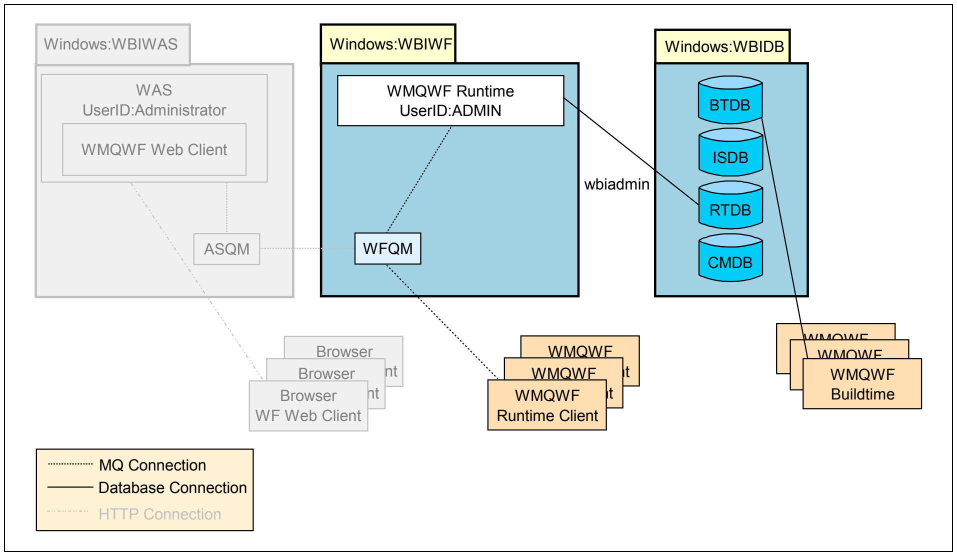


Figure 3-1 WebSphere MQ Workflow topology

The grayed-out sections of Figure 3-1 represent the WebSphere MQ Workflow Web client. We discuss the Web client in 4.1.3, “WebSphere MQ Workflow Web Client configuration” on page 128.

The topology depicted in Figure 3-1 is very flexible with regard to scaleability. Depending on where performance problems occur, it is possible to scale the database access, the Runtime engine, and the Web client access to the system independently. Experience shows that WebSphere MQ Workflow Runtime scales almost linearly when adding more processors or more machines that are connected to the shared remote database.

For a more general discussion of WebSphere MQ Workflow installation, refer to the *IBM WebSphere MQ Workflow V3.5 Installation Guide*, SH12-6288.

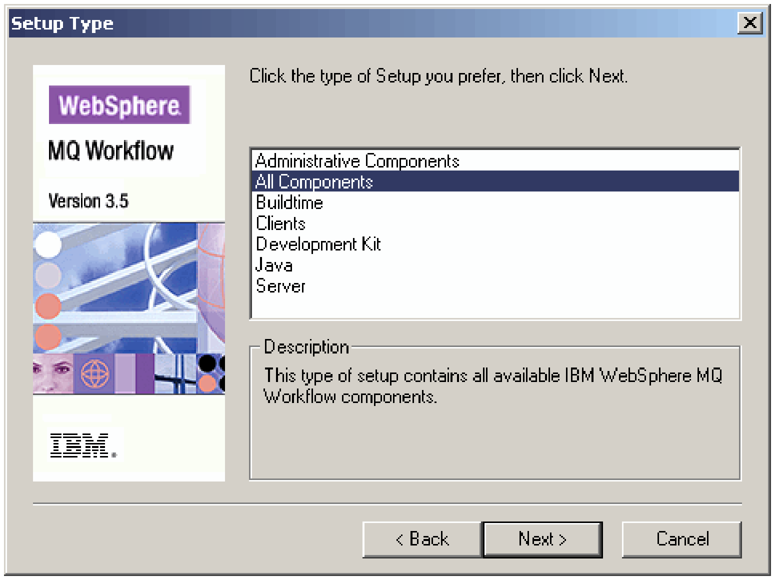
3.1.1 Install WebSphere MQ Workflow Runtime

The installation of WebSphere MQ Workflow is not remarkably different from the installation of any other software product on Windows.

The preliminary steps of the installation entail selection of an installation language, acceptance of the license agreement, and the choice of an installation folder for the software.

Next, the user is prompted to choose which of the Workflow components they want to install. By choosing a specific setup (Administrative components, Buildtime, Clients, and so on), the subset of the Workflow components that are applicable to that particular environment is installed.

On the server machine, we chose to install all components. While it is not required, we chose to install the Buildtime and Runtime clients on the server. This enables the administrator to perform development and runtime verification tasks independent of other network-connected machines. Figure 3-2 and Figure 3-3 on page 48 show the components that we installed on the Workflow server.

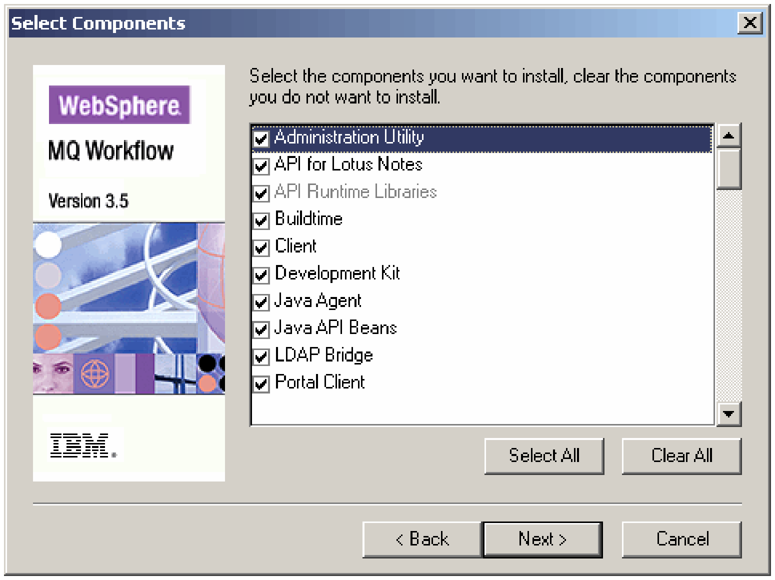


Click the type of Setup you prefer, then click Next.

Administrative Components
All Components
Buildtime
Clients
Development Kit
Java
Server

Description
This type of setup contains all available IBM WebSphere MQ Workflow components.

Figure 3-2 Selecting components to install

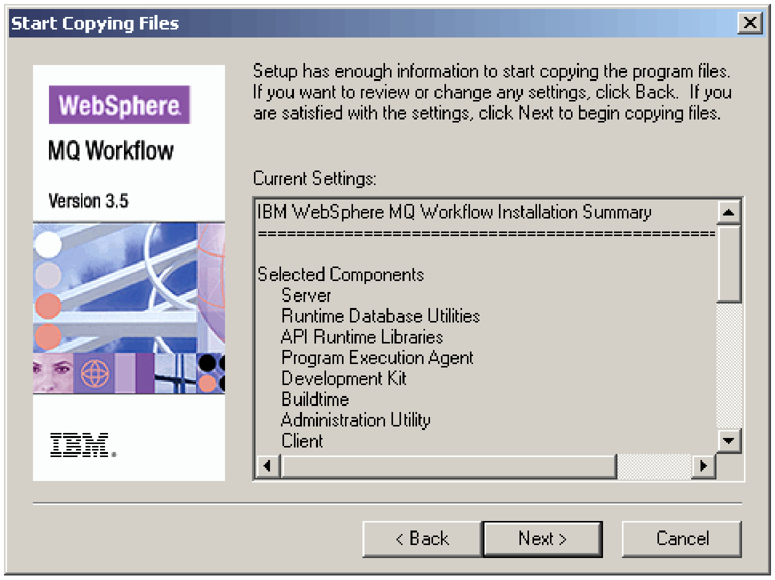


Select the components you want to install, clear the components you do not want to install.

- ✓ Administration Utility
- ✓ API for Lotus Notes
- ✓ API Runtime Libraries
- ✓ Buildtime
- ✓ Client
- ✓ Development Kit
- ✓ Java Agent
- ✓ Java API Beans
- ✓ LDAP Bridge
- ✓ Portal Client

Figure 3-3 Selecting components to install

If necessary, you can click **<Back** to verify or revise any previously selected choices from the Summary window (Figure 3-4 on page 48).



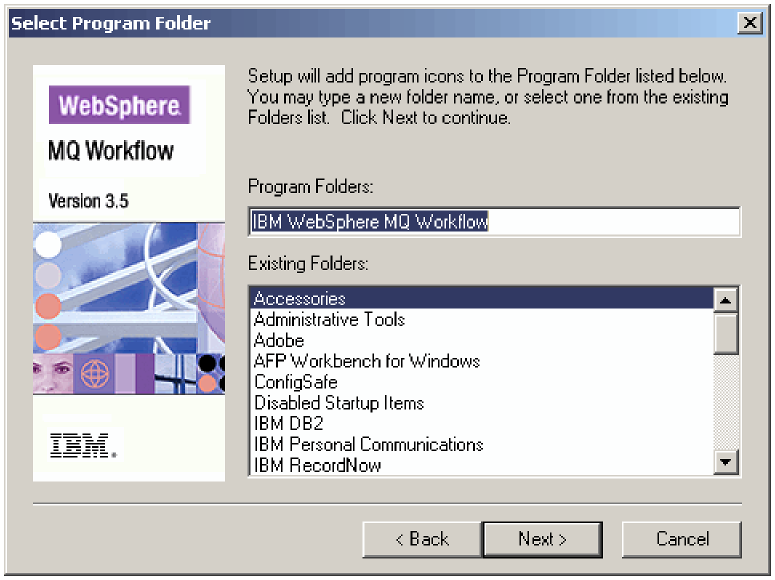
Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Next to begin copying files.

Current Settings:

```
IBM WebSphere MQ Workflow Installation Summary
==============================================

Selected Components
    Server
    Runtime Database Utilities
    API Runtime Libraries
    Program Execution Agent
    Development Kit
    Buildtime
    Administration Utility
    Client
```

Figure 3-4 Installation summary

To complete the installation process, you are given the option to name the Workflow program folder where the icons are to be held (Figure 3-5).



Setup will add program icons to the Program Folder listed below. You may type a new folder name, or select one from the existing Folders list. Click Next to continue.

Program Folders:
IBM WebSphere MQ Workflow

Existing Folders:
Accessories
Administrative Tools
Adobe
AFP Workbench for Windows
ConfigSafe
Disabled Startup Items
IBM DB2
IBM Personal Communications
IBM RecordNow

Figure 3-5 Program Folder

When the installation is finished, a shortcut to the configuration utility appears in the Start Programs menu. The use of this utility is discussed in, “Create a WebSphere MQ Workflow configuration” on page 51. Based on the type of configuration that is created using this utility, more shortcuts will be added.

For a new system, it is good idea to apply the latest service pack before creating a Workflow configuration or a Workflow database.

Note: You can apply the service pack after the creation of a configuration, but then you must rebind the database. This can be done using the administration utility which is part of the Workflow server installation.

The latest service pack for WebSphere MQ Workflow can obtained from:

`http://www-306.ibm.com/software/integration/wmqwf/support/`

3.1.2 Configure WebSphere MQ Workflow

WebSphere MQ Workflow configuration consists of several steps:

- Prepare the DB2 environment.

- Catalog the existing remote DB2 environment.
- Create a WebSphere MQ Workflow configuration.

The WebSphere MQ Workflow configurator creates the Runtime DB2 database, the WebSphere MQ queue manager and all necessary queues, and all WebSphere MQ Workflow Runtime components.

Note: It is not difficult to change the WebSphere MQ Workflow configuration if any mistakes are made. The configurator handles the deletion and creation of all supporting components such as DB2 databases and WebSphere MQ queue managers.

Prepare the DB2 environment

Because we are creating a three-tier environment (remote database) we must also set the DB2 isolation level to **read stability**.

1. Log on to the machine that will host the Runtime database using a user ID with DB2 administration rights.
2. Execute the **`db2set`** command first to verify its current value.

   If it is not `YES`, stop DB2 with the command:

   ```
   db2stop
   ```
3. Enter the following command to change the value:

   ```
   db2set DB2_RR_TO_RS=YES
   ```
4. Start DB2 with the command:

   ```
   db2start
   ```

Repeat this process on the WebSphere MQ Workflow server machine with a user ID with DB2 administration rights.

Catalog the existing remote DB2 instance

In order to access the remote database from our WebSphere MQ Workflow Runtime server, we catalog it on the WebSphere MQ Workflow Runtime machine:

1. Log on to the WebSphere MQ Workflow server with a user ID with DB2 administration rights and open a DB2 command window. Enter the command:

   ```
   CATALOG TCPIP NODE WBIDB REMOTE <server name> SERVER 50000 REMOTE_INSTANCE
   WBIDB SYSTEM <server name> OSTYPE  WIN
   ```
2. On the machine that will host the WebSphere MQ Workflow server, enter the following two commands:

   ```
   db2 UPDATE DBM CFG USING TP_MON_NAME mqmax
   ```

```
db2 TERMINATE
```

3. Verify that the Runtime database user can attach to the remote instance by entering the following command:

   ```
   db2 ATTACH TO wbidb USER wbiadmin USING RTDBpassword
   ```

   In this command, `wbidb` is the host name of the database server and `wbiadmin` and `RTDBpassword` are a valid ID and password for the remote database server.

4. Follow this up with the command:

   ```
   db2 DETACH
   ```

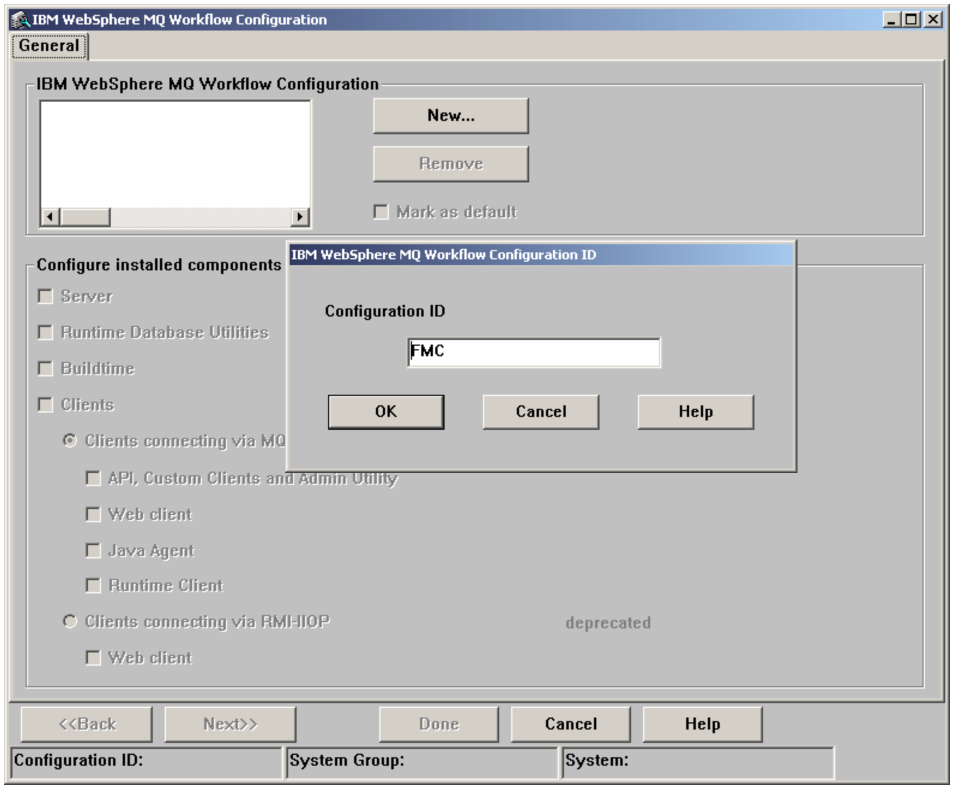
5. When using the WebSphere MQ Workflow Configuration Utility, you must provide folder names for the database containers. These folder names should exist in the file system on the remote database server. Log on to the DB2 server with a user ID that has DB2 administration rights. Create the Runtime database location and Buildtime Database Location directories as planned in Table 2-1 on page 39 and Table 2-3 on page 41, respectively.

Create a WebSphere MQ Workflow configuration

This section outlines the steps needed to configure the WebSphere MQ Workflow Runtime environment. With the aid of the Workflow Configuration Utility, all necessary elements specific to the Workflow domain are created and the information is stored in a configuration file. This configuration file can be replicated to ensure accuracy and proper execution during the build phase.

If there are any problems during the execution of this section, make sure that all sections before this one were executed successfully.

1. Start the configuration process by selecting **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **IBM WebSphere MQ Workflow Configuration Utility**.
2. The configuration utility opens on the General tab where you can define a new profile and select the components that you want to configure in this profile. Figure 3-6 on page 52 shows this screen. Click **New** and provide a name for the profile, as planned in Table 2-1 on page 39 (see Configuration Identifier).


IBM WebSphere MQ Workflow Configuration
General
IBM WebSphere MQ Workflow Configuration
New...
Remove
Mark as default
Configure installed components
Server
Runtime Database Utilities
Buildtime
Clients
Clients connecting via MQ
API, Custom Clients and Admin Utility
Web client
Java Agent
Runtime Client
Clients connecting via RMI-IIOP
deprecated
Web client
IBM WebSphere MQ Workflow Configuration ID
Configuration ID
OK
Cancel
Help
<<Back
Next>>
Done
Cancel
Help
Configuration ID:
System Group:
System:

Configuration ID: FMC

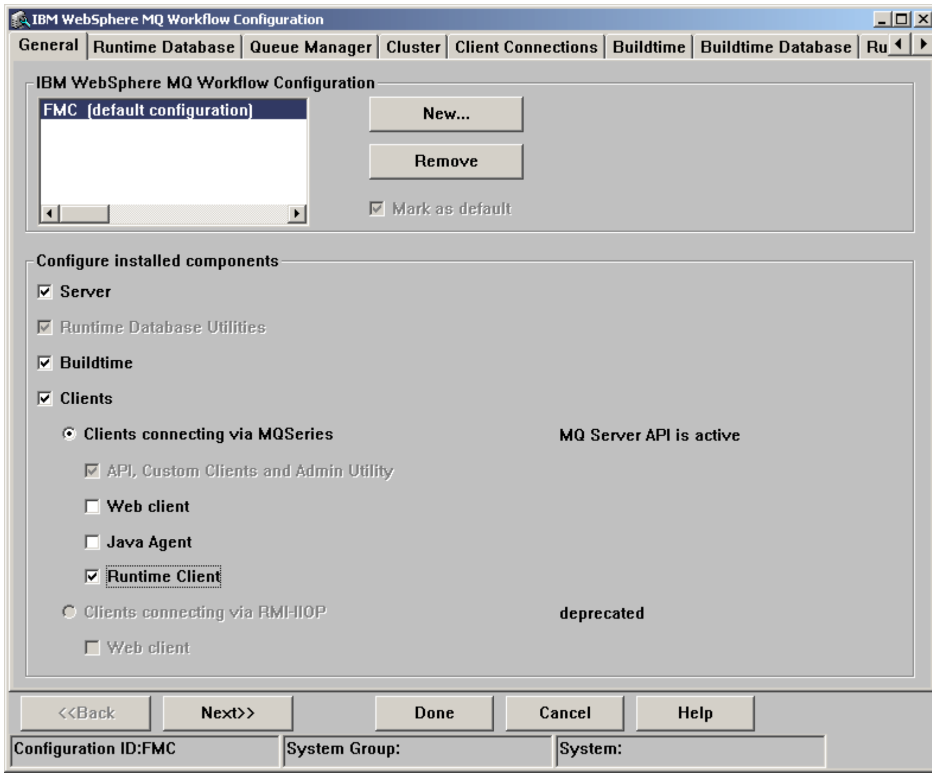
Figure 3-6 Configuration ID

3. After the new profile is named, select the components that you want to configure as part of this profile.

   Figure 3-7 on page 53 shows the components that were selected for this configuration. Note that the selection of certain components implies the selection of other components. For example, the Runtime Database Utilities option is selected as soon as the Server component is selected.

   We choose to install the Buildtime and Runtime Client features to allow us to perform some local validation. Even when a proper development environment is in place, it is worthwhile to have the option to log on locally to the server using the Runtime client. It can also be useful to install this client feature because it provides a bird's-eye view of your environment from a GUI perspective. This is especially helpful if your runtime environment is installed on a non-Windows platform.

   Note also that every selected option may result in one or more tabs being added to enable you to specify configuration parameters for that component.


IBM WebSphere MQ Workflow Configuration

General | Runtime Database | Queue Manager | Cluster | Client Connections | Buildtime | Buildtime Database | Ru

IBM WebSphere MQ Workflow Configuration

FMC (default configuration)

New...

Remove

☑ Mark as default

Configure installed components

☑ Server

☑ Runtime Database Utilities

☑ Buildtime

☑ Clients

● Clients connecting via MQSeries — MQ Server API is active

☑ API, Custom Clients and Admin Utility

☐ Web client

☐ Java Agent

☑ Runtime Client

○ Clients connecting via RMI-IIOP — deprecated

☐ Web client

<<Back | Next>> | Done | Cancel | Help

Configuration ID:FMC | System Group: | System:

Figure 3-7 General tab - components selected for this configuration

4. Click **Next** to proceed to the Runtime Database tab (Figure 3-8 on page 54).

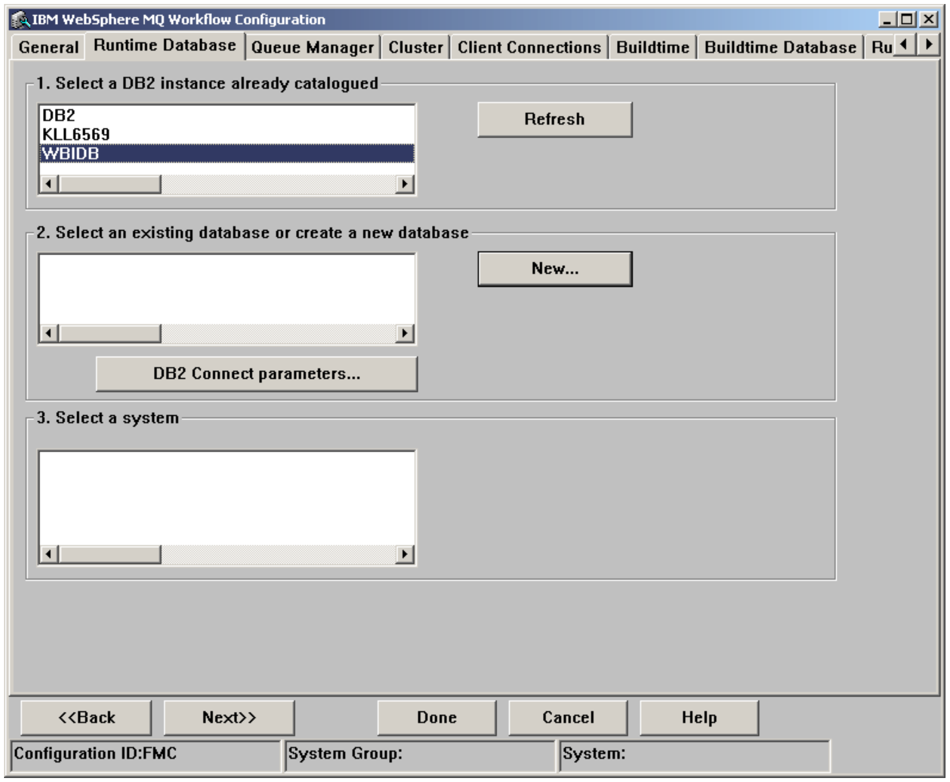
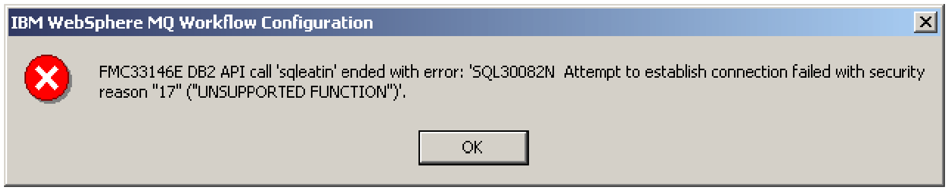


Figure 3-8 Runtime Database tab

The top list box contains the known DB2 instances. This includes the default local instance, DB2, and the remote instance, **WBIDB**, that we configured earlier.

Before selecting the appropriate instance, we must specify the correct connection parameters. If an instance is selected before the database connection parameters are provided, the utility will try to connect to the instance using your current logon ID, which may or may not be authorized to do that.

Figure 3-9 on page 55 shows the result of selecting the WBIDB instance without providing the correct connection parameters.


IBM WebSphere MQ Workflow Configuration

FMC33146E DB2 API call 'sqleatin' ended with error: 'SQL30082N Attempt to establish connection failed with security reason "17" ("UNSUPPORTED FUNCTION")'.

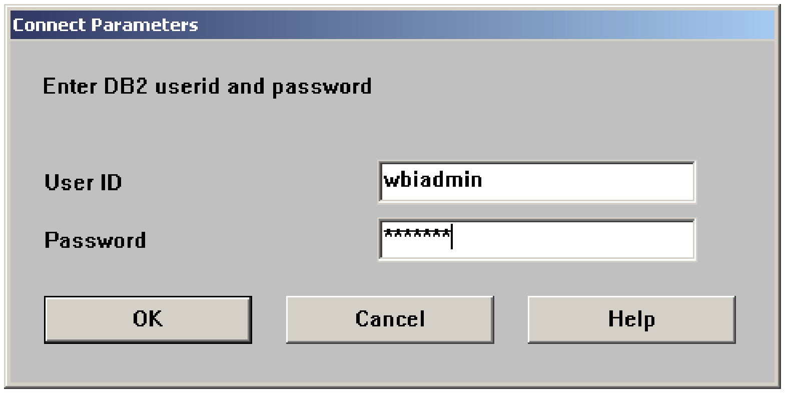
Figure 3-9 DB2 error due to wrong connection parameters

Tip: If you selected the instance before providing the correct connection parameters, you can follow these steps to recover:

1. Close the error message box by clicking **OK**.
2. Click **DB2 Connect™ parameters**.
3. When the user ID and password are provided, click **Refresh**.

You can then select the correct instance again and the configuration utility should not return an error this time.

4. The correct approach is to click **DB2 Connect parameters...** first. This opens the window that is shown in Figure 3-10, where you enter the database user ID and password as planned in Table 2-1 on page 39.


Connect Parameters

Enter DB2 userid and password

User ID wbiadmin

Password *******

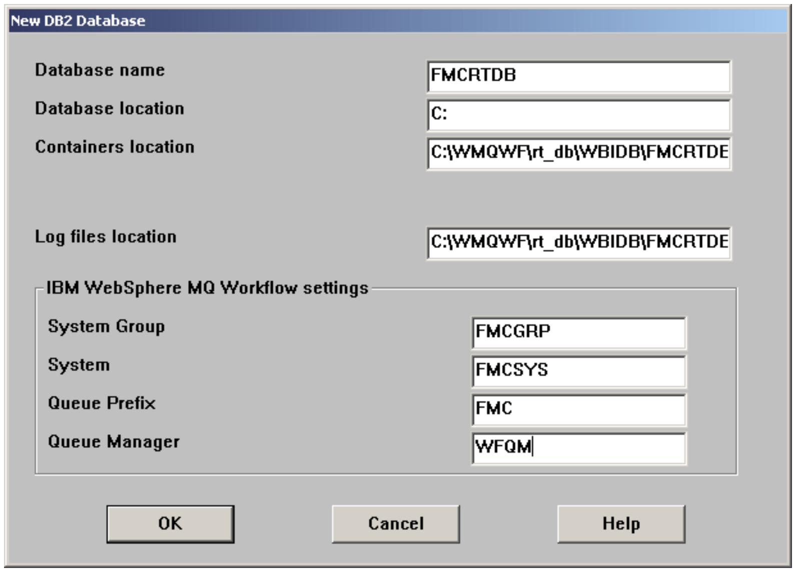
Figure 3-10 Provide DB2 connection parameters

Now that we have successfully connected to the database server, we have the option to create a new runtime database. If the configuration utility detects an existing database that was created according to the WebSphere MQ Workflow schema, it will list it in the second list box seen in Figure 3-8 on page 54. This feature enables you to redefine a server based on an existing database;

something that you might want to do when migrating or moving a server to a new hardware platform.

In our case, we create a new server:

1. Click **New** to open another window (Figure 3-11) in which you specify the parameters for the creation of the new database, WebSphere MQ parameters, and WebSphere MQ Workflow system domain parameters.
2. Enter the name of the database as planned in Table 2-1 on page 39 (see WebSphere MQ Workflow Runtime database name) and the database, container, and log file location. Remember that these values refer to folder names on the remote system where the database server is running. The names are listed in Table 2-1 on page 39 (see Runtime database location).
3. In the lower section of the window (Figure 3-11), provide values for the System Group and System properties, as well as the name of the queue manager and a prefix for the queues owned by this server. Every WebSphere MQ Workflow server communicates via WebSphere MQ to the other components or servers in a multimachine solution.


New DB2 Database

Database name: FMCRTDB
Database location: C:
Containers location: C:\WMQWF\rt_db\WBIDB\FMCRTDE
Log files location: C:\WMQWF\rt_db\WBIDB\FMCRTDE

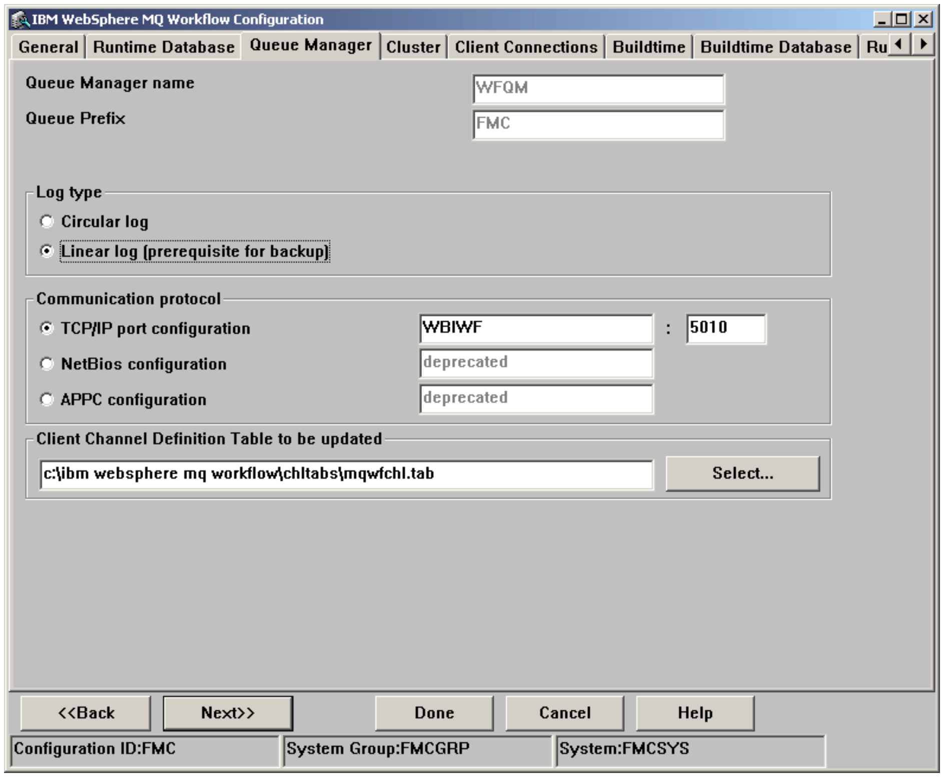
IBM WebSphere MQ Workflow settings
System Group: FMCGRP
System: FMCSYS
Queue Prefix: FMC
Queue Manager: WFQM

Figure 3-11 Provide runtime database parameters

4. The next section of the configuration is the **Queue Manager** tab. (See Figure 3-12 on page 57.) Here you specify more options that will direct the definition of MQ components used by Workflow such as the queue manager name and log type (circular or linear) and the desired queue prefix. Note that

linear logging is required if you want to be able to recover persistent messages in all circumstances, including hard disk failure.

MQ communication parameters are also specified on this tab. These values are used to define the listener component of the queue manager and to define a client channel to the queue manager. This client channel definition is stored in a client channel definition table in the file that you specify. The client channel definition table is required for use of the standard WebSphere MQ Workflow clients and tools. (The Web-based client does not require the client channel definition table. So it is not necessary to distribute the client channel definition table to every Workflow end user machine.)



Queue Manager name: WFQM
Queue Prefix: FMC

Log type
Circular log
[✓ Linear log (prerequisite for backup)]

Communication protocol
[✓ TCP/IP port configuration] WBIWF : 5010
NetBios configuration deprecated
APPC configuration deprecated

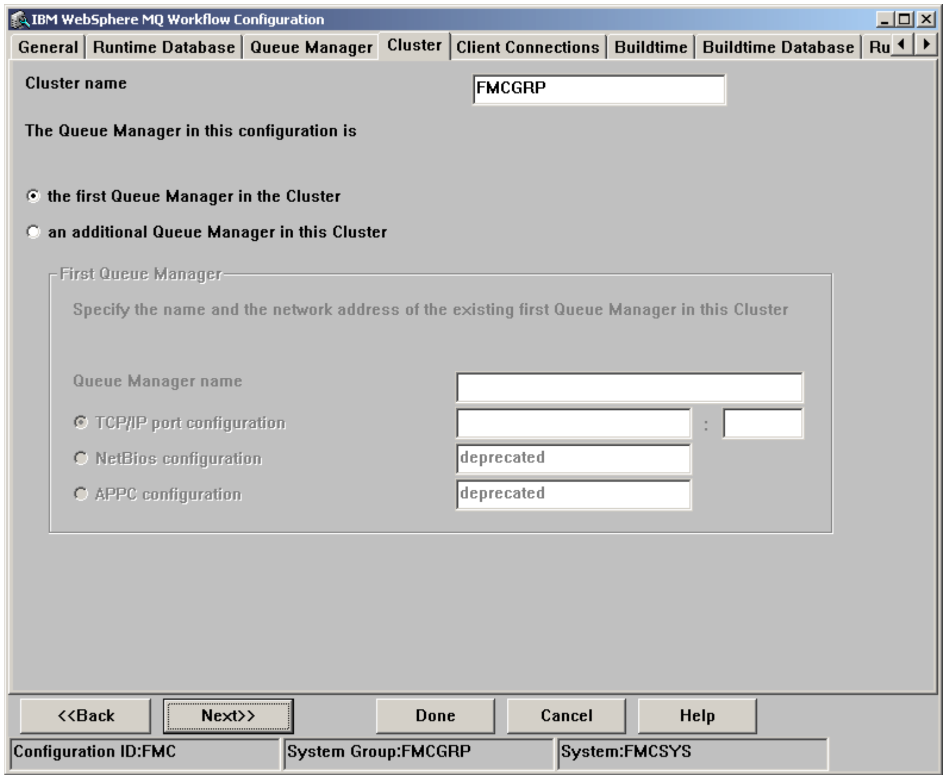
Client Channel Definition Table to be updated
c:\ibm websphere mq workflow\chltabs\mqwfchl.tab

Configuration ID:FMC
System Group:FMCGRP
System:FMCSYS

Figure 3-12 Queue Manager tab

5. WebSphere MQ Workflow utilizes WebSphere MQ queue manager clustering technology. A *cluster* is a network of queue managers that are logically associated in some way. The primary benefits of queue manager clustering are reduced system administration, increased fault tolerance, and workload balancing. Queue manager clustering is relevant when building a group of Workflow servers.

The Cluster tab (Figure 3-13) allows you to define your Workflow server queue managers as members of a WebSphere MQ queue manager cluster. When defining the first Workflow server in a cluster, you specify the name of the cluster. The configuration utility then creates a queue manager belonging to that cluster and labels this queue manager as a full repository for the cluster. For any additional Workflow servers, use the Cluster tab to provide the name of the existing cluster and to provide connection parameters to the first queue manager. The newly defined queue manager can then join the cluster and interrogate the full repository queue manager about any objects that are defined to the cluster. A full repository queue manager knows all about clustered WebSphere MQ objects and can provide connection parameters automatically to any other queue managers in the cluster.



Cluster name: FMCGRP

The Queue Manager in this configuration is

[✓ the first Queue Manager in the Cluster]

an additional Queue Manager in this Cluster

First Queue Manager

Specify the name and the network address of the existing first Queue Manager in this Cluster

Queue Manager name

[✓ TCP/IP port configuration] :

NetBios configuration: deprecated

APPC configuration: deprecated

Configuration ID:FMC | System Group:FMCGRP | System:FMCSYS

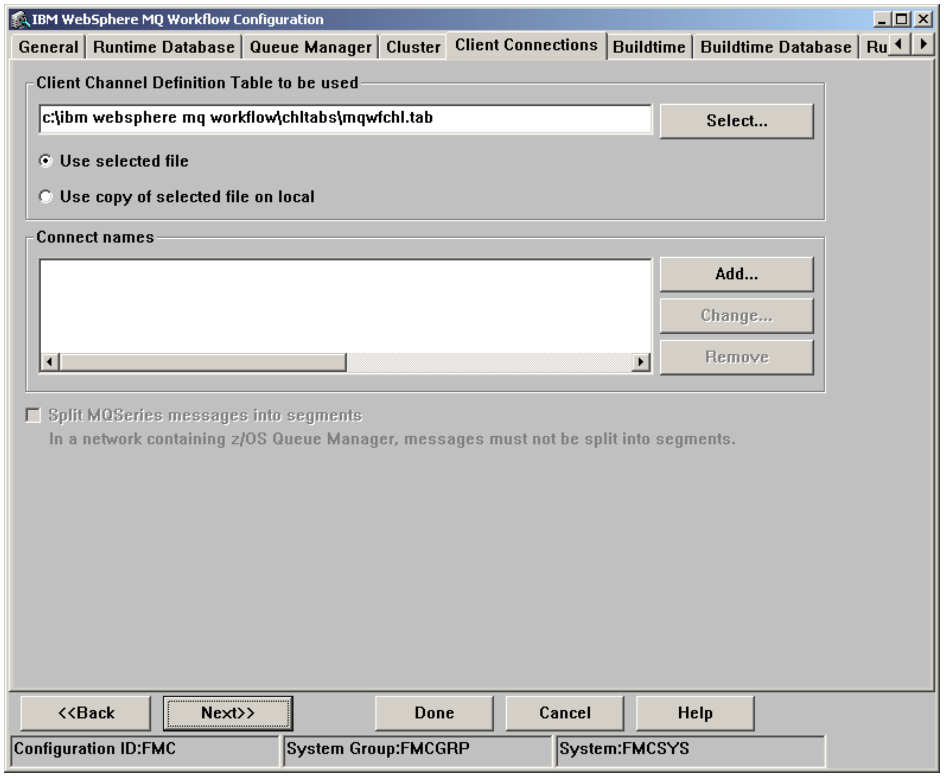
Figure 3-13   Cluster tab

6. The Client Connections tab (Figure 3-14 on page 59) enables you to manage the use of client channel definition files. It also enables you to point to an existing channel definition file and add connection parameters for a new queue manager to the chosen file. This technique is used when you are running multiple Workflow servers and want the end user to be able to

connect a standard WebSphere MQ Workflow client application to any server in the group.

The upper section is used to select the Client Channel Definition Table to be used. Click **Select...** to browse for the desired file. Or, you can directly enter the fully qualified name of the desired Client Channel Definition Table file.

The lower section allows you to add a connect name to your Workflow server.


IBM WebSphere MQ Workflow Configuration

General | Runtime Database | Queue Manager | Cluster | Client Connections | Buildtime | Buildtime Database | Ru

Client Channel Definition Table to be used

c:\ibm websphere mq workflow\chltabs\mqwfchl.tab

Select...

(●) Use selected file

( ) Use copy of selected file on local

Connect names

Add...

Change...

Remove

[ ] Split MQSeries messages into segments

In a network containing z/OS Queue Manager, messages must not be split into segments.

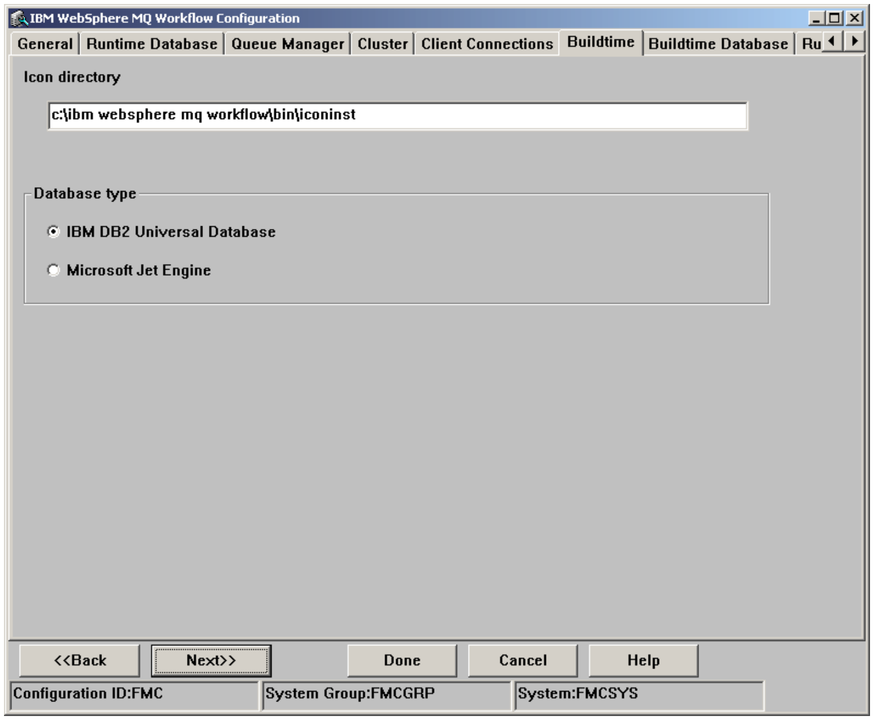
<<Back | Next>> | Done | Cancel | Help

Configuration ID:FMC | System Group:FMCGRP | System:FMCSYS

Figure 3-14 Client Connections tab

7. The Buildtime tab (Figure 3-15 on page 60) allows you to specify the type of your Buildtime database. Allowable choices are IBM DB2 Universal Database™ or Microsoft Jet Engine. The dedicated database server in our example makes use of IBM DB2 Universal Database. So, we select **IBM DB2 Universal Database**.

   Click **Next**.



Icon directory

c:\ibm websphere mq workflow\bin\iconinst

Database type

(●) IBM DB2 Universal Database

( ) Microsoft Jet Engine

Configuration ID:FMC

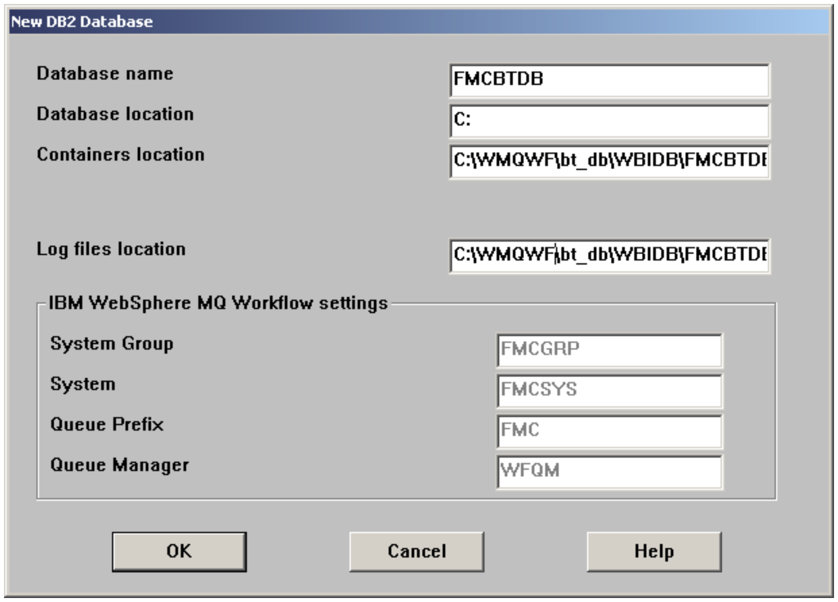
System Group:FMCGRP

System:FMCSYS

Figure 3-15 Buildtime tab

8. The Buildtime Database tab is structured similarly to the Runtime Database tab (Figure 3-8 on page 54). Click **DB2 Connect parameters** to provide the database user ID and password, and select the instance name (**WBIDB**).

   Click **New** to open the New DB2 Database window (Figure 3-16). The values that are provided here from Table 2-3 on page 41.


New DB2 Database

Database name	FMCBTDB
Database location	C:
Containers location	C:\WMQWF\bt_db\WBIDB\FMCBTDI
Log files location	C:\WMQWF\bt_db\WBIDB\FMCBTDI

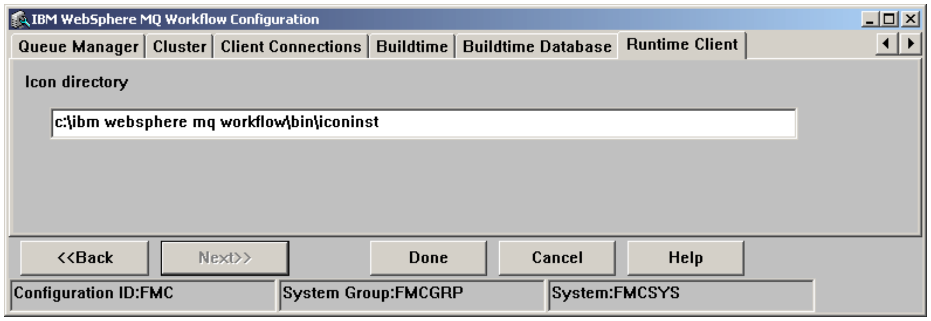
IBM WebSphere MQ Workflow settings

System Group	FMCGRP
System	FMCSYS
Queue Prefix	FMC
Queue Manager	WFQM

Figure 3-16 Provide Buildtime database parameters

9. The Runtime Client tab (Figure 3-17) is used to specify a directory where custom icons are kept. These icons are utilized to define your business processes, and define and change your MQ Workflow process model, topology, and staff definitions.

   Click **Done** and your configuration will be executed.


IBM WebSphere MQ Workflow Configuration

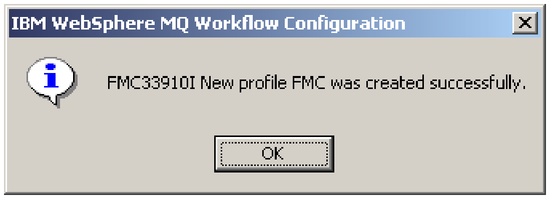
Icon directory

c:\ibm websphere mq workflow\bin\iconinst

Configuration ID:FMC
System Group:FMCGRP
System:FMCSYS

Figure 3-17 Runtime Client tab

You receive a successful completion code of FMC33910I if the new profile is created successfully (Figure 3-18).


IBM WebSphere MQ Workflow Configuration

FMC33910I New profile FMC was created successfully.

Figure 3-18 Successful completion

3.1.3 Verify WebSphere MQ Workflow server

This is a short verification procedure to help you ensure that your WebSphere MQ Workflow environment is operational.

Starting and stopping the WebSphere MQ Workflow system

First we will look at starting the supporting components individually.

Starting the components individually

1. Start the Runtime database. This can be done from a DB2 command window on the database server (this is a remote database):

   ```
   dbstart
   ```

   Alternatively, on the Microsoft windows platforms, you can use the Windows Services applet to start the DB2 service.

2. WebSphere MQ Workflow runs on top of WebSphere MQ. So, our next step is to start the WebSphere MQ objects associated with the Workflow environment.

   You can start the WebSphere MQ components using either the Windows Services applet or the MQSeries Services application. Or, if you prefer to use the command line, the commands are:

   a. Start the Workflow queue manager:

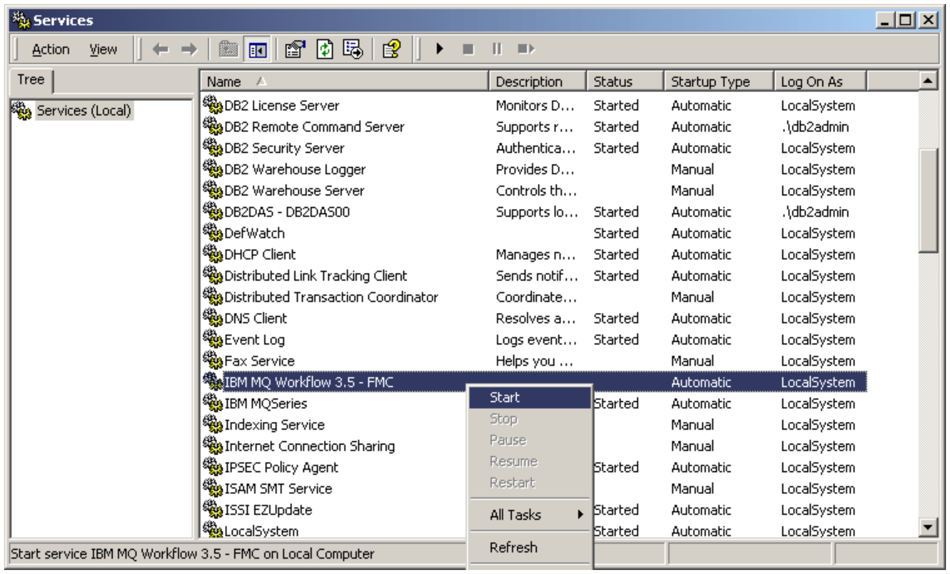
      ```
      strmqm WFQM
      ```

   b. Start the Workflow MQ trigger monitor:

      ```
      runmqtrm -m WFQM -q FMCTRIGGER
      ```

3. Start the Workflow server with the Windows Services applet. Having logged on to your IBM WebSphere Workflow server machine (WBIWF), go to **Start** → **Settings** → **Control Panel** → **Administrative Tools** → **Services**

Here you can start the server by paging down to the WebSphere MQ Workflow server and highlighting it. Then, right-click the high-lighted item and select **Start** (Figure 3-19).



Name	Description	Status	Startup Type	Log On As
DB2 License Server	Monitors D...	Started	Automatic	LocalSystem
DB2 Remote Command Server	Supports r...	Started	Automatic	.\db2admin
DB2 Security Server	Authentica...	Started	Automatic	LocalSystem
DB2 Warehouse Logger	Provides D...		Manual	LocalSystem
DB2 Warehouse Server	Controls th...		Manual	LocalSystem
DB2DAS - DB2DAS00	Supports lo...	Started	Automatic	.\db2admin
DefWatch		Started	Automatic	LocalSystem
DHCP Client	Manages n...	Started	Automatic	LocalSystem
Distributed Link Tracking Client	Sends notif...	Started	Automatic	LocalSystem
Distributed Transaction Coordinator	Coordinate...		Manual	LocalSystem
DNS Client	Resolves a...	Started	Automatic	LocalSystem
Event Log	Logs event...	Started	Automatic	LocalSystem
Fax Service	Helps you ...		Manual	LocalSystem
IBM MQ Workflow 3.5 - FMC			Automatic	LocalSystem
IBM MQSeries		Started	Automatic	LocalSystem
Indexing Service			Manual	LocalSystem
Internet Connection Sharing			Manual	LocalSystem
IPSEC Policy Agent		Started	Automatic	LocalSystem
ISAM SMT Service			Manual	LocalSystem
ISSI EZUpdate		Started	Automatic	LocalSystem
LocalSystem		Started	Automatic	LocalSystem

Start service IBM MQ Workflow 3.5 - FMC on Local Computer

Figure 3-19 Window Services application- Start IBM WebSphere MQ Workflow

4. Check the Windows Event Viewer application log for start-up information regarding the application. This way you can verify that the server has started successfully or, if it has not, begin to diagnose and resolve any problems related to an unsuccessful start of the server.

5. Connect to the Buildtime components. Go to **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **Workflow Buildtime - FMC**.

   An IBM WebSphere MQ Workflow Buildtime - ODBC Logon window will appear. Logon with the User ID (`ADMIN`) and Password (`password`) to verify connectivity.

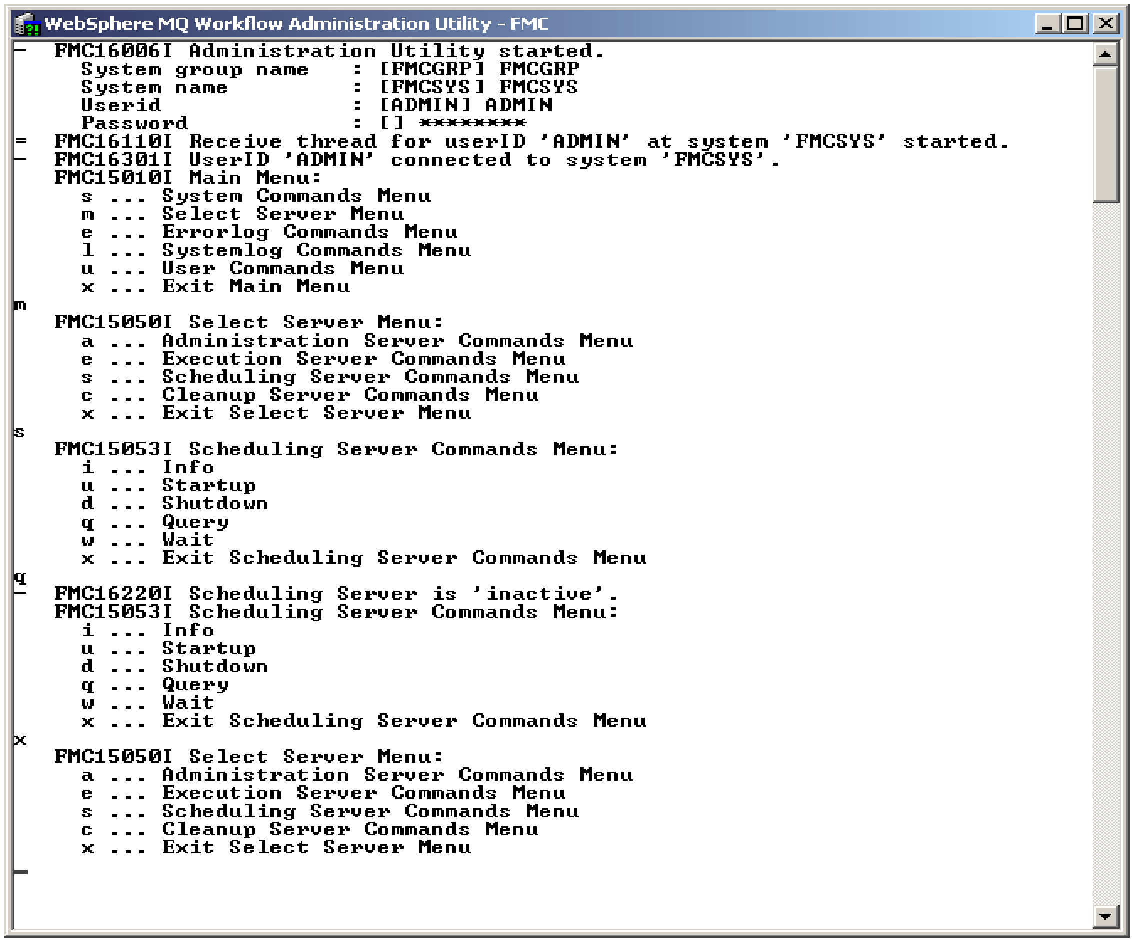
6. The Administration Utility can be accessed by going to **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **Workflow Administration Utility (Configuration file name)**.

   Another way of initiating the Administration Utility is by entering the following command:

   ```
   fmcautil -u user -p password
   ```

   The Administration Utility can be used to start other Workflow related components. As you can see in Figure 3-20 on page 64, the utility provides several server related options. Using the Administration Utility you can also query the status, shutdown, and restart your server environment.

Consult the *IBM WebSphere MQ Workflow Administration Guide Version 3.5*, SH12-6289 on command options for administering your environment.



```
-  FMC16006I Administration Utility started.
     System group name    : [FMCGRP] FMCGRP
     System name          : [FMCSYS] FMCSYS
     Userid               : [ADMIN] ADMIN
     Password             : [] *********
=  FMC16110I Receive thread for userID 'ADMIN' at system 'FMCSYS' started.
-  FMC16301I UserID 'ADMIN' connected to system 'FMCSYS'.
   FMC15010I Main Menu:
     s ... System Commands Menu
     m ... Select Server Menu
     e ... Errorlog Commands Menu
     l ... Systemlog Commands Menu
     u ... User Commands Menu
     x ... Exit Main Menu
m
   FMC15050I Select Server Menu:
     a ... Administration Server Commands Menu
     e ... Execution Server Commands Menu
     s ... Scheduling Server Commands Menu
     c ... Cleanup Server Commands Menu
     x ... Exit Select Server Menu
s
   FMC15053I Scheduling Server Commands Menu:
     i ... Info
     u ... Startup
     d ... Shutdown
     q ... Query
     w ... Wait
     x ... Exit Scheduling Server Commands Menu
q
-  FMC16220I Scheduling Server is 'inactive'.
   FMC15053I Scheduling Server Commands Menu:
     i ... Info
     u ... Startup
     d ... Shutdown
     q ... Query
     w ... Wait
     x ... Exit Scheduling Server Commands Menu
x
   FMC15050I Select Server Menu:
     a ... Administration Server Commands Menu
     e ... Execution Server Commands Menu
     s ... Scheduling Server Commands Menu
     c ... Cleanup Server Commands Menu
     x ... Exit Select Server Menu
_
```

Figure 3-20 Administration utility

7. Finally, a verification of system start-up can be viewed in the logs directory specific to the identifier profile.

   `c:\IBM WebSphere MQ Workflow\cfgs\fmc\log\fmcsys.log`

   See Example 3-1 on page 65 for the output of the fmcsys.log.

Example 3-1 fmcsys.log

```
10/12/2004 7:30:00 AM FMC10100I Administration server starting.
10/12/2004 7:30:00 AM FMC10110I Administration server for system FMCSYS
started.
10/12/2004 7:30:02 AM FMC10200I Execution server for system FMCSYS started.
10/12/2004 7:30:02 AM FMC10500I Execution server instance started.
10/12/2004 7:30:06 AM FMC10000I System startup complete. System FMCSYS in
system group FMCGRP is now running.
```

Sequence for stopping the components

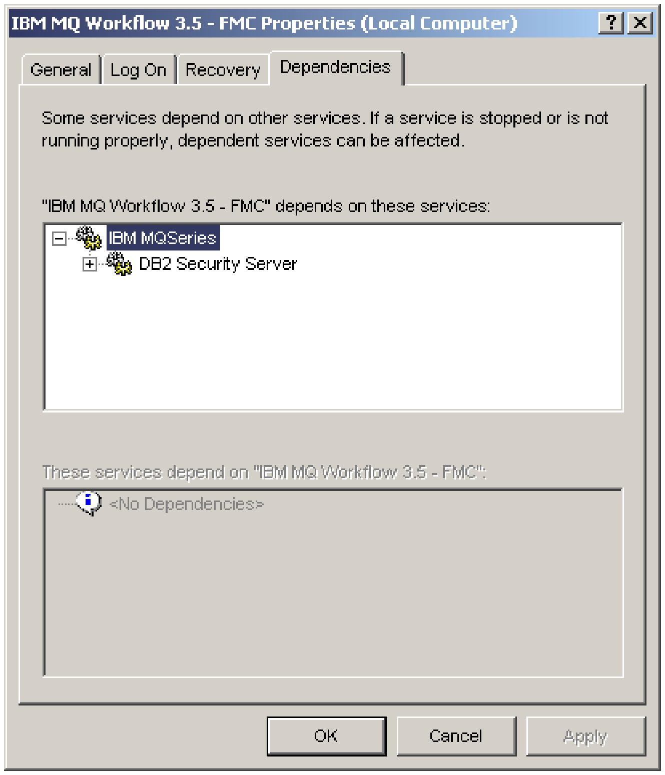
When stopping the entire IBM WebSphere Workflow environment, the components should be stopped in the following sequence:

1. Stop all WebSphere MQ Workflow components.
2. Stop the WebSphere MQ services.
3. Stop DB2.

Start-up using the WebSphere MQ Workflow Windows service

We just described the start-up of each component in the WebSphere MQ Workflow environment individually. We can also start the environment using the Windows service WebSphere MQ Workflow, which starts the dependent components automatically. This is not true for the remote database server, of course. You will always have to make sure that the database server and the database instance are up and running before you can do any work with the workflow server.

The link between the workflow service and the IBM MQSeries service can be inspected. When you double-click the WebSphere MQ Workflow service in the Windows Services applet, a window opens with several configuration options. Select the **Dependencies** tab to inspect the dependent services, IBM MQSeries and DB2 Security Server. See Figure 3-21 on page 66. Note that the local DB2 instance is not a dependent service. Whenever the service WebSphere MQ Workflow is started, the operating system will make sure that the dependent services get started first.


IBM MQ Workflow 3.5 - FMC Properties (Local Computer)

Some services depend on other services. If a service is stopped or is not running properly, dependent services can be affected.

"IBM MQ Workflow 3.5 - FMC" depends on these services:

- IBM MQSeries
  - DB2 Security Server

These services depend on "IBM MQ Workflow 3.5 - FMC":

- <No Dependencies>

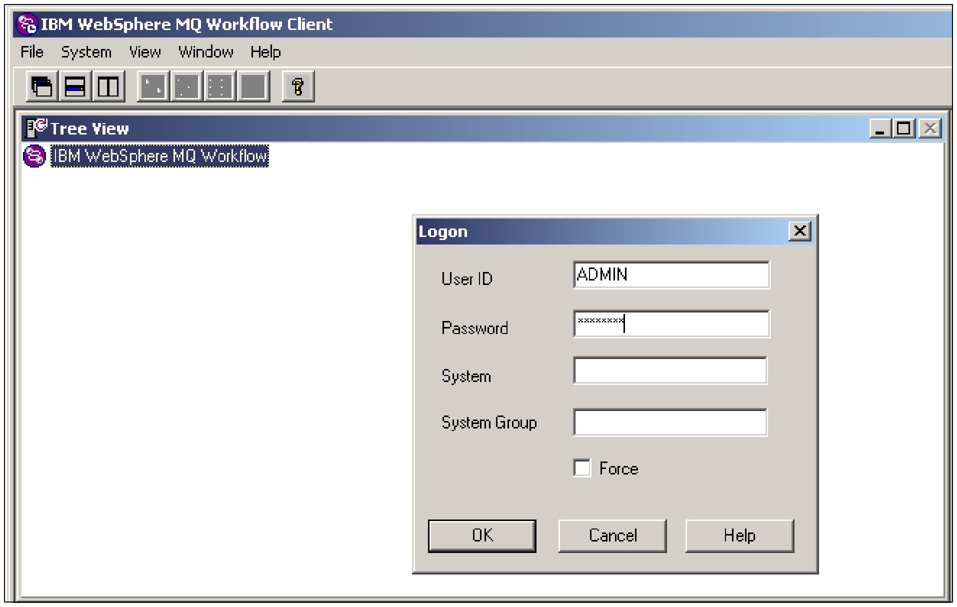
Figure 3-21 Details of the MQSeries Workflow service

While this feature simplifies the tasks of a system administrator, it also can have some side effects. Assume that a Windows server™ has several Automatic services and that this server has more than a single queue manager or other resource-hungry services. This means that starting up the workflow server can take some time. On the other hand, the operating system only allows a limited amount of elapsed time for each service to report its start-up completion. If the start-up is not completed within this time frame, the operating system will report that the start-up has failed. To avoid this, you might want to configure the service for a manual start-up.

Using a local Runtime client

By choosing all components during the installation phase of WebSphere MQ Workflow, you will have access to a Runtime client. As mentioned earlier in the text, the Client feature provides a medium for local and remote validation. To login to the Runtime Client, follow these steps:

1. Select **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **WebSphere MQ Workflow Client - FMC**.


IBM WebSphere MQ Workflow Client
File System View Window Help
Tree View
IBM WebSphere MQ Workflow
Logon

User ID ADMIN

Password ********

System

System Group

☐ Force

Figure 3-22 Runtime client login

2. As in Figure 3-22, enter the User ID (`ADMIN`) and Password (`password`). There is no need to enter the System or System Group information because the Workflow Client automatically defaults to the local profile identifier (FMC). The **Force** option allows you the ability to log-on in the event that a Client session is already open.
3. After a successful logon, a Tree View of your System Group appears (Figure 3-23). Along with the System Group, you can see undefined Process Templates, Instances, and a Worklist. These objects will be configured later in the text.

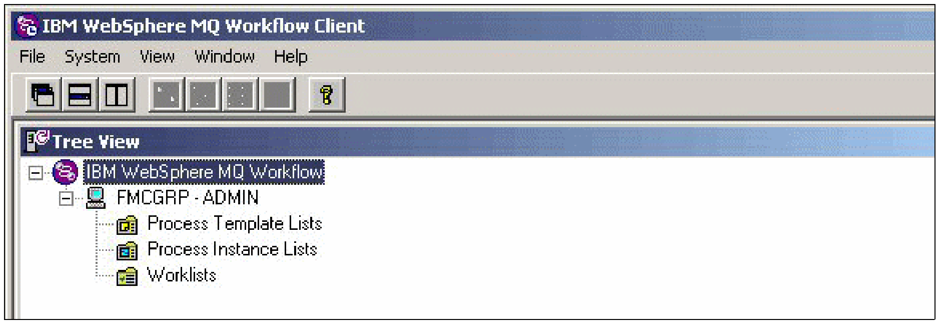

IBM WebSphere MQ Workflow Client
File System View Window Help
Tree View
IBM WebSphere MQ Workflow
FMCGRP - ADMIN
Process Template Lists
Process Instance Lists
Worklists

Figure 3-23 Runtime client

This completes the setup for the Workflow server component. The setup of the Web client, a development client, and a management client is discussed in Chapter 4, “Implementing client components” on page 119.

3.2 InterChange Server installation and configuration

In this section, we install and configure WebSphere InterChange Server. Additionally, information pertaining to the installation and configuration of software upon which InterChange Server is dependent is discussed.

Each InterChange Server instance has an associated repository database. The database will be created on a remote machine.

Note: WebSphere MQ and a database system such as DB2 should be installed before installing WebSphere InterChange Server.

The installation and configuration process includes the following tasks:

- Confirm that your system meets the hardware and software requirements that are appropriate to your planned InterChange Server environment.
- Ensure that all supporting software is available. This includes Java runtime, DB2, and WebSphere MQ.
- Create and configure a database to act as the InterChange Server repository.
- Install and configure InterChange Server.

3.2.1 Prerequisite tasks

Before installing the InterChange Server component, ensure that all of the necessary prerequisite software is installed. Up-to-date information about system requirements can be found at:

http://www.ibm.com/software/integration/wbiserver/requirements/

Configuring the database server

With the architecture upon which we have decided, we have to create a database on the WBIDB machine, which is our DB2 server machine:

1. Execute the following command in a DB2 command window:

   ```
   db2 CREATE DATABASE ISDB ALIAS ISDB
   ```

2. We give our WebSphere Business Integration Server administrator the necessary permissions by executing these commands in a DB2 command window:

```
db2 CONNECT TO ISDB USER db2admin USING sas313r
db2 GRANT DBADM, CREATETAB, BINDADD, CONNECT, CREATE_NOT_FENCED_ROUTINE,
IMPLICIT_SCHEMA, LOAD, CREATE_EXTERNAL_ROUTINE, QUIESCE_CONNECT ON DATABASE
TO USER WBIADMIN;
```

Configuring the database client

To make this database available to the InterChange Server, we must catalog the instance from the database server and then the database.

1. To catalog the instance execute the following command:

   ```
   CATALOG TCPIP NODE WBIDB REMOTE server name SERVER 50000 REMOTE_INSTANCE
   WBIDB SYSTEM server name OSTYPE  WIN
   ```

2. To validate that the DB2 instance is available from our InterChange Server machine, execute the command:

   ```
   db2 LIST NODE DIRECTORY
   ```

3. To catalog the database that we created initially, execute the following command, which catalogs this database on the client machine.

   ```
   db2 CATALOG DATABASE ISDB AS ISDB AT NODE wbidb AUTHENTICATION SERVER
   ```

4. To validate our database catalog, execute:

   ```
   db2 LIST DATABASE DIRECTORY
   ```

5. To confirm that the InterChange Server database user can access the database, issue the following command:

   ```
   db2 CONNECT TO isdb USER wbiadmin USING itso4you
   ```

Tuning the InterChange Server database

To configure our database instance and database, we executed the commands that are shown below. Use a database administrator account and execute these commands (using your own appropriate parameters) on the database server itself in a DB2 command window. You can also perform these updates by using the DB2 Control Center.

Attention: The values in the commands listed here are for our configuration and will not fit your requirements. Take this example as a guideline for the parameters to configure when running the database.

- `db2 update db cfg for ISDB using applheapsz 4096`
- `db2 update db cfg for ISDB using dbheap 4800`
- `db2 update db cfg for ISDB using maxappls 1000`
- `db2 update db cfg for ISDB using locklist 400`
- `db2 update db cfg for ISDB using logbufsz 512`
- `db2 update db cfg for ISDB using logfilsiz 4000`

- `db2 update db cfg for ISDB using logprimary 6`
- `db2 update db cfg for ISDB using logsecond 10`
- `db2 update db cfg for ISDB using buffpage 64000`
- `db2 alter bufferpool ibmdefaultbp size -1`
- `db2 update dbm cfg using maxagents 1000`
- `db2 update dbm cfg using mon_heap_sz 1024`
- `db2set DB2_RR_TO_RS=YES`
- `db2set DB2COMM=tcpip`

Installing JDK

The InterChange Server system uses Java technology to execute the collaborations. Besides a Java runtime, the InterChange Server also needs a Java compiler to compile development objects such as maps and collaborations. The WebSphere InterChange Server requires JDK™ Version 1.4.2 for the compiler and runtime. This version is provided as part of both the InterChange Server product and the Adapter Framework. It consists of two self-extracting files that are stored in the IBM_JDK_WIN32 folder on the product CD. Copy the files to a temporary directory and execute them. After extraction, run the install program.

Tip: To check the Java version in a Microsoft Windows environment, use the command:

```
java -version
```

3.2.2 Installation of WebSphere BI Server components

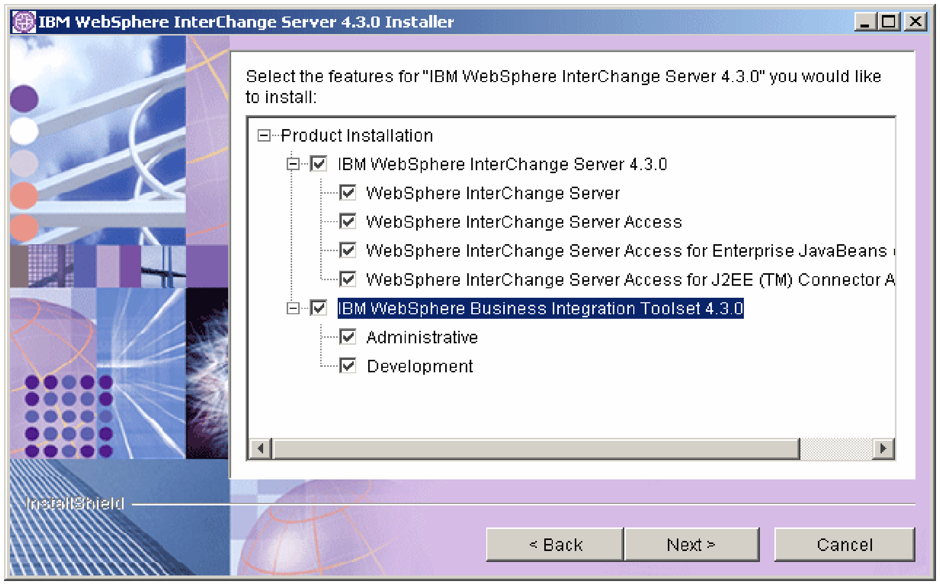
Besides installing the InterChange Server software itself, other WebSphere Business Integration components might be required depending on your integration project. These components may include WebSphere Business Integration Adapters, Data Handlers, or prebuilt Industry Collaborations.

Installing InterChange Server

The installation of the InterChange Server is a standard administrator’s task. When the installation program is started, the usual steps about language selection and license acceptance are presented.

1. The first major decision is the choice of the installation folder. This name should *not* include spaces. You can accept the default, which is C:\IBM\WebsphereICS.
2. In the next step, select the product components to be installed. We are currently installing the product for a runtime server, and we chose to install development and administration tools as well. (See Figure 3-24 on page 71.)

In the next chapter, we look at the installation of the product for the purpose of development or administration only.

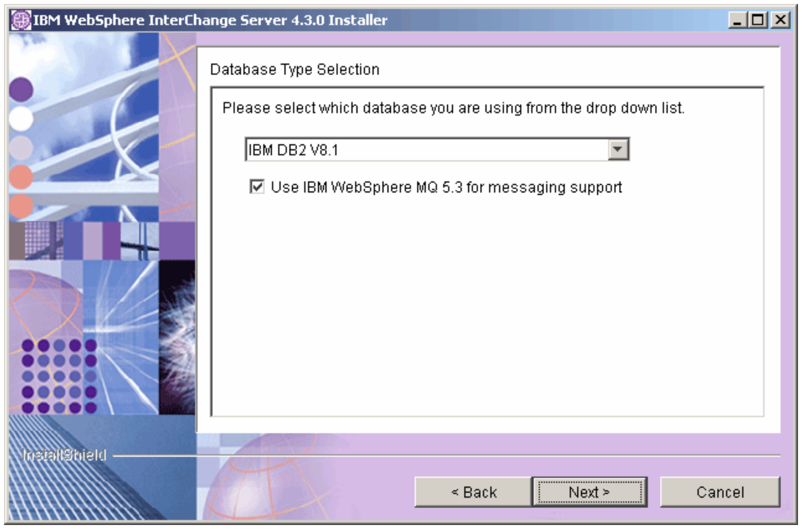

IBM WebSphere InterChange Server 4.3.0 Installer

Select the features for "IBM WebSphere InterChange Server 4.3.0" you would like to install:

- Product Installation
  - [✓] IBM WebSphere InterChange Server 4.3.0
    - [✓] WebSphere InterChange Server
    - [✓] WebSphere InterChange Server Access
    - [✓] WebSphere InterChange Server Access for Enterprise JavaBeans
    - [✓] WebSphere InterChange Server Access for J2EE (TM) Connector A
  - [✓] IBM WebSphere Business Integration Toolset 4.3.0
    - [✓] Administrative
    - [✓] Development

Figure 3-24   Select WebSphere InterChange Server components

3. In the next step, choose the database provider, **IBM DB2 V8** in our environment. Also, select the option to use WebSphere MQ as the provider for messaging services.



Database Type Selection

Please select which database you are using from the drop down list.

IBM DB2 V8.1

☑ Use IBM WebSphere MQ 5.3 for messaging support

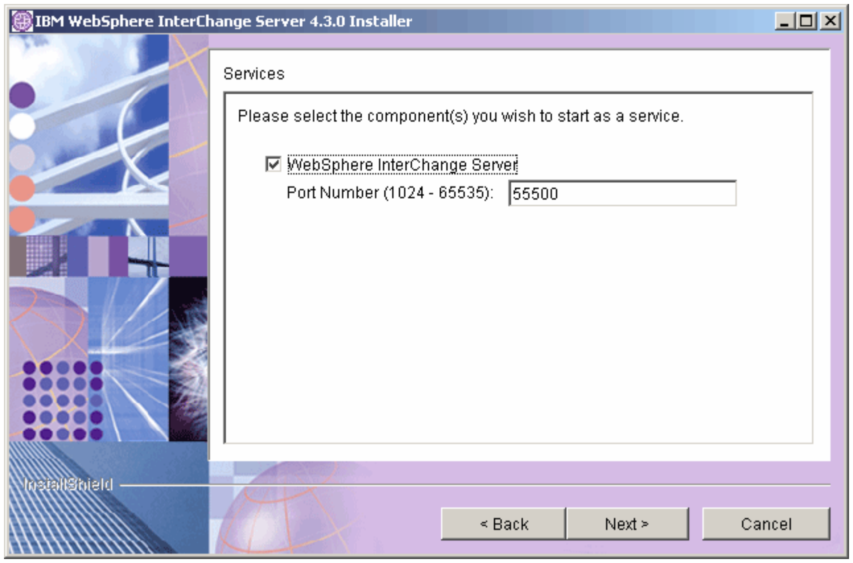
Figure 3-25 Provide details about the prerequisite software

4. Each InterChange Server instance has to have a name, and this name is set during installation. The name of the InterChange Server should be chosen in such a way that it is unique within the local network.
5. For each prerequisite software product, DB2 and WebSphere MQ, the installation program might ask additional questions, such as the installation folder for DB2 and the name of the folder that holds the Java libraries for WebSphere MQ.
6. During the installation process, you can also choose components to run as Windows services. We do not perform this step at this time.

When the installation is finished, the configuration program is launched. You could perform the configuration task at this time, but we choose to end this program at this point and continue with the installation of additional WebSphere Business Integration components.

If you are familiar with earlier versions of the InterChange Server, you might note that we did not install the familiar VisiBroker component. This component was replaced in Version 4.2.2 by a new Object Request Broker (which is installed by the installer). There is no specific configuration to be performed now. See Figure 3-26 on page 73.

Note: If you choose to run the InterChange Server as a Windows service during the installation, you must select a valid port during the installation for use by the ORB. However, this port number could be chosen later, as well.



Services

Please select the component(s) you wish to start as a service.

☑ WebSphere InterChange Server

Port Number (1024 - 65535): 55500

Figure 3-26 Select port number for use by the ORB

3.2.3 Configuration of the InterChange Server

Now that all WebSphere InterChange Server components are installed, we can start configuring it. This configuration could be done right after the WebSphere InterChange Server installation, but we prefer to complete all of the installations before starting to configure the InterChange Server.

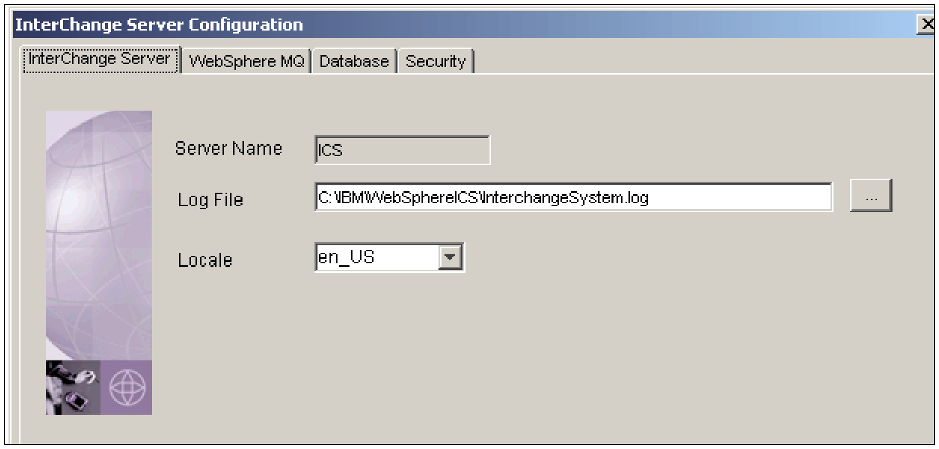
You can configure the InterChange Server in two ways. First, a configuration wizard is used to create a basic configuration that enables you to start the InterChange Server for the first time. This utility is used to set the following parameters:

- InterChange Server: log file path and locale configuration
- WebSphere MQ: parameters that are needed for a client WebSphere MQ connection
- Database configuration: parameters that are needed to connect to the database server

Then, using the System Manager, you can alter these and other configuration settings.

To start the configuration wizard, select **Start** → **Programs** → **IBM InterChange Server** → **IBM InterChange Server** → **IBM InterChange Server Configuration Wizard**.

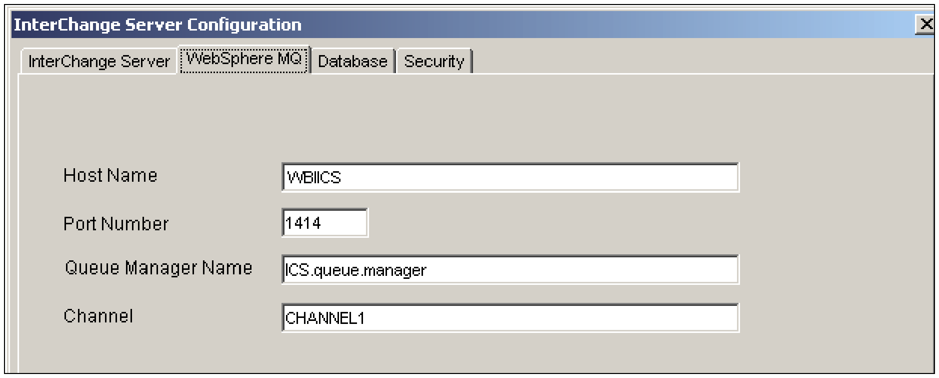
1. The first tab is used to configure the WebSphere InterChange Server parameters (Figure 3-27) by specifying the InterChange Server log location, name, and the locale. We accepted the default values.



Server Name: ICS
Log File: C:\IBM\WebSphereICS\InterchangeSystem.log
Locale: en_US

Figure 3-27 InterChange Server configuration for WebSphere InterChange Server

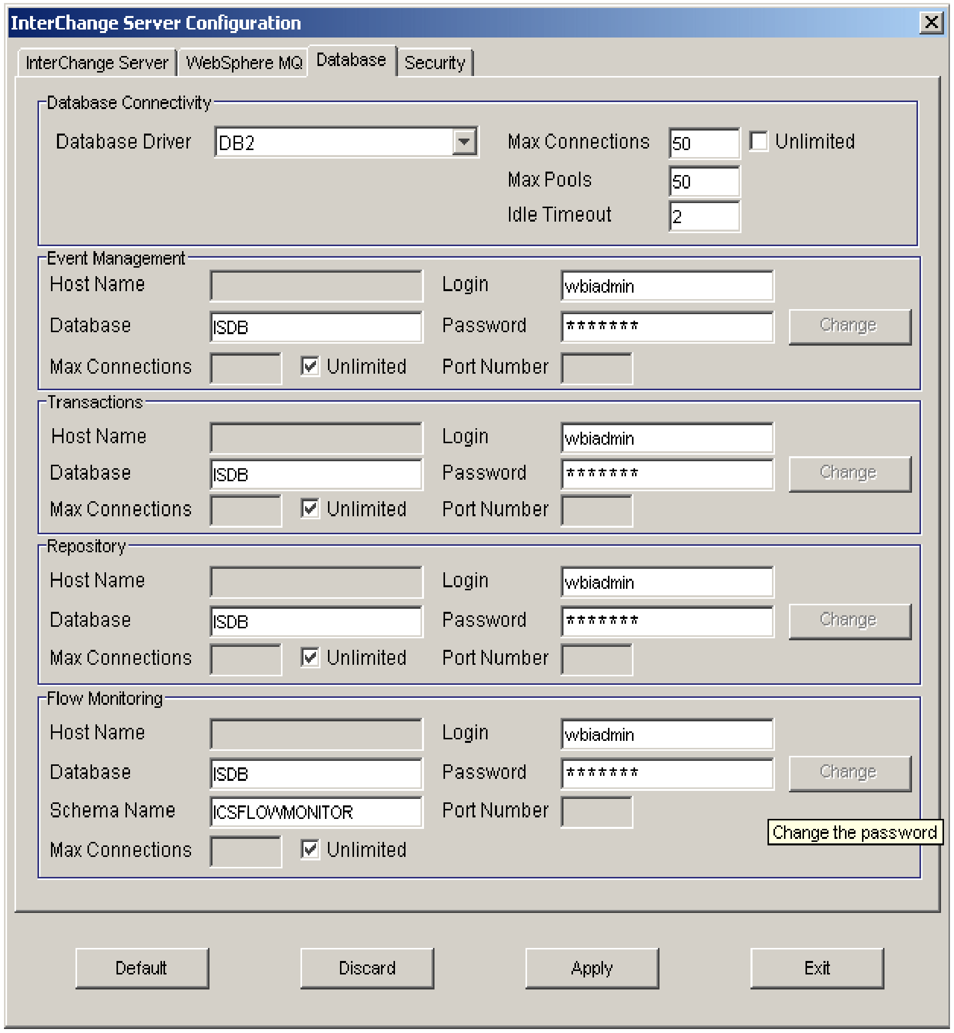
2. When the InterChange Server configuration is complete, click the **WebSphere MQ** tab to proceed with the WebSphere MQ configuration. In this tab (Figure 3-28), we specify the Host Name, Queue Manager Port, Queue Manager Name, and Channel from Table 2-4 on page 41.



Host Name: WBIICS
Port Number: 1414
Queue Manager Name: ICS.queue.manager
Channel: CHANNEL1

Figure 3-28 WebSphere MQ configuration for WebSphere InterChange Server

3. When the WebSphere MQ configuration is complete, click the **Database** tab to proceed with the database configuration.


InterChange Server Configuration

InterChange Server | WebSphere MQ | Database | Security

Database Connectivity

Database Driver	DB2	Max Connections	50	☐ Unlimited
		Max Pools	50	
		Idle Timeout	2	

Event Management

Host Name		Login	wbiadmin	
Database	ISDB	Password	*******	Change
Max Connections	☑ Unlimited	Port Number		

Transactions

Host Name		Login	wbiadmin	
Database	ISDB	Password	*******	Change
Max Connections	☑ Unlimited	Port Number		

Repository

Host Name		Login	wbiadmin	
Database	ISDB	Password	*******	Change
Max Connections	☑ Unlimited	Port Number		

Flow Monitoring

Host Name		Login	wbiadmin	
Database	ISDB	Password	*******	Change
Schema Name	ICSFLOWMONITOR	Port Number		
Max Connections	☑ Unlimited			

Change the password

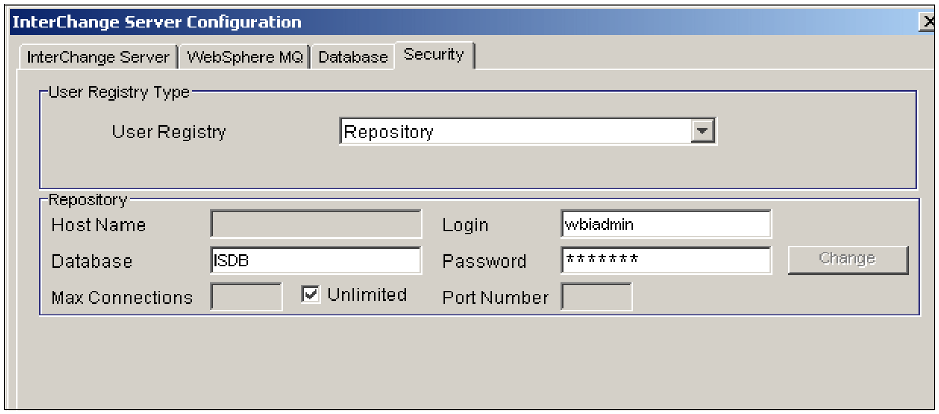
Default | Discard | Apply | Exit

Figure 3-29 Database configuration for WebSphere InterChange Server

In this tab (Figure 3-29), we specify the Database Driver, Database Max Connections, Database name, Database login, and Database password from Table 2-4 on page 41. Notice that you can distribute the database workload over one to four different databases. For information about database tuning and sizing, refer to Chapter 11, “Tuning a WebSphere BI Server infrastructure” on page 461.

Note: If a different database driver is used, you might have to specify different parameters.

4. When the database configuration is complete click the **Security** tab. In this tab, we specify the user registries to be either a database repository or LDAP. Select **Repository** and specify the name of the database, login user ID and the password as shown in Figure 3-30.



User Registry: Repository

Host Name:
Database: ISDB
Max Connections: [✓ Unlimited]
Login: wbiadmin
Password: *******
Port Number:

Figure 3-30 Security configuration for WebSphere InterChange Server

5. After all of the parameters have been modified to match the configuration, click **Apply** and **Exit**.

This configuration utility generates an XML file called InterchangerSystem.cfg, which is located in the installation directory. The InterChange Server uses the configuration file when it starts. Other parameters in this file enable further tuning. However, for a first start-up of the server, the generated configuration file should be sufficient.

Configuring WebSphere MQ

The installation of the InterChange Server has created several scripts in the D:\IBM\WebSphereICS\mqseries folder that can be used to configure WebSphere MQ. We should configure these scripts to match our WebSphere MQ configuration. Table 3-1 lists the scripts that are used by WebSphere MQ.

Table 3-1 WebSphere MQ scripts

Script name	Script description
configure_mq.bat	Creates WebSphere MQ queue manager.
crossworld_mq.tst	Objects to create in WebSphere MQ queue manager.
start_mq.bat	Starts WebSphere MQ queue manager and listener.
test_mq.bat	Stops WebSphere MQ queue manager.

Using configure_mq.bat

This command file is called with a shortcut in the **Start** → **Programs** menu. It is used to create a queue manager. The same command file is also used to define the WebSphere MQ objects that are listed in the file crossworld_mq.tst. This latest file contains a template of queue names that you would define for each adapter instance.

The shortcut in the **Start** menu has several parameters, including the name of the queue manager and the name of the file that contains MQ definitions (crossworld_mq.tst). Before using the shortcut, you should ensure that it has the name of the queue manager that you expect. This shortcut was created during the installation (that is, before we used the configuration wizard), which is where we designated what queue manager to use.

The **`crtmqm`** command in configure_mq.bat is used with only one parameter, `-q`, which indicates that the queue manager is the default queue manager. This is fine as long as your system does not contain a queue manager that happens to be the default as well. Using the parameter `-q` simply means that now this new queue manager will be the default, which can break applications or scripts that you might have for the existing queue manager. As a general rule, for machines that have multiple queue managers, it is probably better to not label a certain queue manager as the default queue manager. No other parameters are used for the **`crtmqm`** command. However, you should consider altering the command to include the parameter `-u` for naming the dead letter queue. You should also consider changing the logging parameters for the queue manager.

Example 3-2 on page 78 shows the MQ definitions that are executed as part of using configure_mq.bat. Except for the channel definition, all of these objects are required for adapters only. The file should be considered as a template. As such, when we configure an adapter, we will use this template to define the queues that the adapter requires.

Be sure that the name of the client channel matches your settings in Figure 3-28 on page 74.

Example 3-2 crossworlds_mq.tst

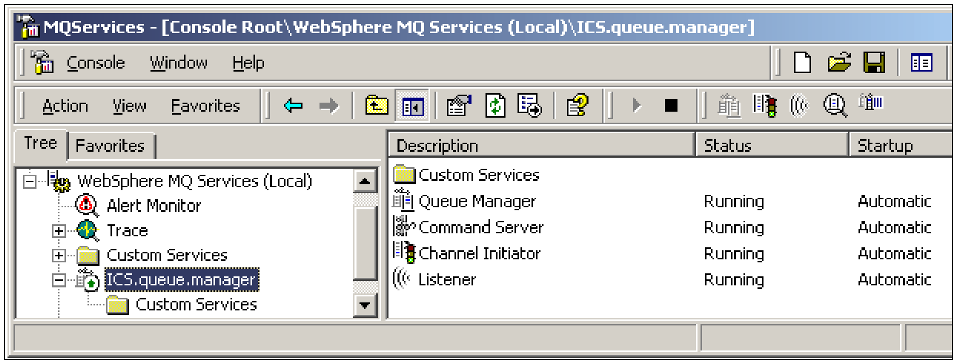
```
DEFINE QLOCAL(IC/SERVER_NAME/DestinationAdapter)
DEFINE QLOCAL(AP/DestinationAdapter/SERVER_NAME)

DEFINE QLOCAL(AdapterName/AdminInQueue)
DEFINE QLOCAL(AdapterName/AdminOutQueue)
DEFINE QLOCAL(AdapterName/DeliveryQueue)
DEFINE QLOCAL(AdapterName/RequestQueue)
DEFINE QLOCAL(AdapterName/ResponseQueue)
DEFINE QLOCAL(AdapterName/FaultQueue)
DEFINE QLOCAL(AdapterName/SynchronousRequestQueue)
DEFINE QLOCAL(AdapterName/SynchronousResponseQueue)
DEFINE QLOCAL(DLQ)

DEFINE CHANNEL(CHANNEL1) CHLTYPE(SVRCONN) TRPTYPE(TCP)
```

1. After these scripts are configured, execute **`configure_mq`** by selecting **Start** → **Programs** → **IBM WebSphere InterChange Server** → **IBM WebSphere MQ** → **Configure queue manager**.

   As an alternative to using the **`configure_mq.bat`** script, you can use the WebSphere MQ Explorer application to define the queue manager and the client channel. Besides the fact that you do not need to remember command line option, it has the additional advantage that the IBM MQSeries service will be configured correctly for the queue manager and the TCP/IP listener for WebSphere MQ. This then removes the need to use the Start Listener shortcut in the **Start** → **Programs** menu for the InterChange Server.

   For WebSphere MQ tuning purposes, we should modify the LogBufferPages parameter. This increases the performance when using persistent messages. In general, adapters communicate with the InterChange Server by sending and receiving persistent messages.

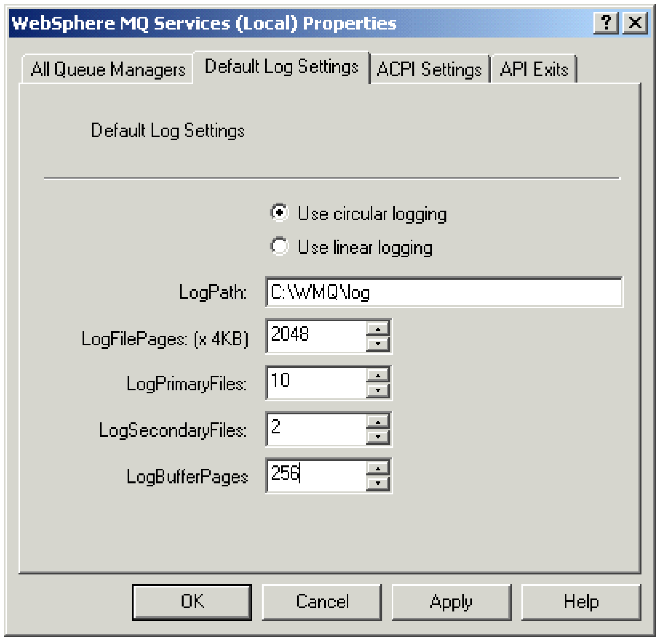
   To change this parameter, start the IBM MQSeries services application. See Figure 3-31 on page 79.



Description	Status	Startup
Custom Services		
Queue Manager	Running	Automatic
Command Server	Running	Automatic
Channel Initiator	Running	Automatic
Listener	Running	Automatic

Figure 3-31   Using WebSphere MQ Services application

2. Right-click the root element **WebSphere MQ Services (Local)** and select **Properties**. Select the **Default Log Settings** tab and increase the value for the LogBufferPages parameter, as shown in Figure 3-32.



(● Use circular logging) ○ Use linear logging

LogPath: C:\WMQ\log
LogFilePages: (x 4KB) 2048
LogPrimaryFiles: 10
LogSecondaryFiles: 2
LogBufferPages 256

Figure 3-32   Increase the log buffer

3. In order to check the WebSphere MQ configuration related to InterChange Server a test program is shipped with WebSphere InterChange Server. This script is located in <ICS>/bin and it is called testMQ.bat. Execute the test program to test the configuration.

3.2.4 Verifying the configuration

To verify the configuration of the InterChange Server, we start it for the first time, connect to it using the System Manager, and change a few more server settings.

Starting the InterChange Server

The InterChange Server relies mostly on the database manager and also on WebSphere MQ during its operations. Before starting the InterChange Server, make sure that the remote database server is running. The InterChange Server also relies on the name server for communication between the server and tools such as the System Manager. If the name server has not been configured to run as a Windows service, start it manually by launching the PersistentNameServer.bat script, which can be found in D:\IBM\WebSphereICS\bin.

When the name server is active, you can start the InterChange Server by selecting **Start** → **Programs** → **IBM InterChange Server** → **IBM InterChange Server** → **IBM InterChange Server**.

This launches a command window that shows logging and error information. However, given the current configuration of the logging component, the complete log is available at this time only in the InterChangeSystem.log file in D:\IBM\WebSphereICS, which we had specified in Figure 3-27 on page 74. This file shows that the InterChange Server first creates several tables in the repository server. When the initialization of the server is finished, it reports a Ready message.

For easy visualization of the log messages, you can use the Log Viewer tool (**Start** → **Programs** → **IBM InterChange Server** → **IBM WebSphere Business Integration Toolset** → **Administrative)**.

Using System Manager

During the installation of the InterChange Server, we selected the option to also install the toolset, which includes the System Manager. The System Manager is a specialized perspective within WebSphere Studio Workbench that is used to administer and configure the InterChange Server.

In this section, we use this locally installed version of the System Manager to verify that the InterChange Server is running correctly and that we can make changes to it by using the System Manager. However, in Chapter 4, "Implementing client components" on page 119, we create a dedicated administration and monitoring client, which enables administration of remote InterChange Servers, including non-Windows instances of the InterChange Server.

1. The first time that you start WebSphere Studio Workbench, it likely will open on the Resource perspective instead of the System Manager perspective. To switch perspectives, select **Window** → **Open Perspective** → **Other**. Select **System Manager** from the list (Figure 3-33), and click **OK**.

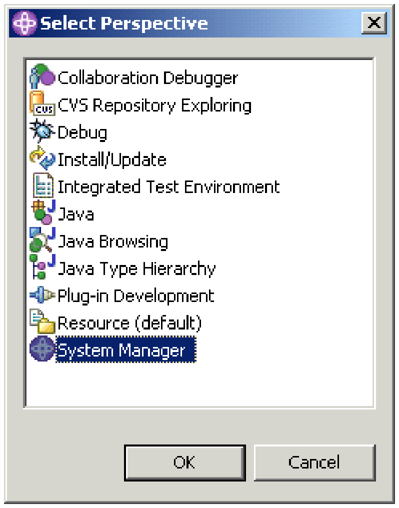
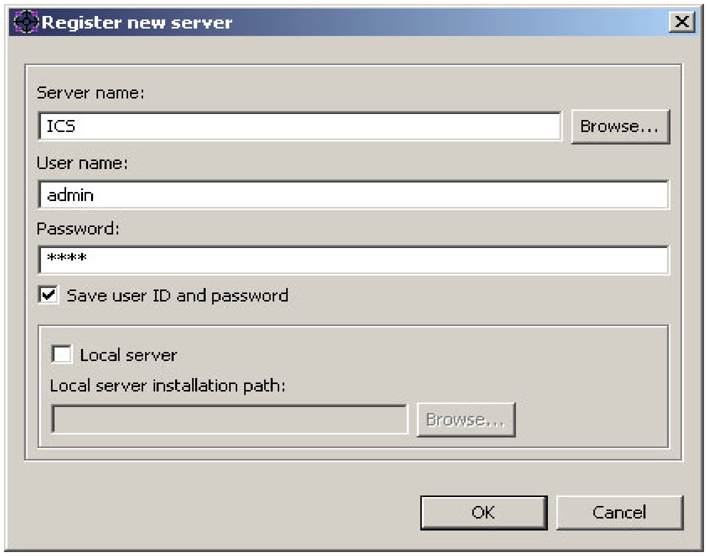


Figure 3-33 Open the System Manager perspective

A *Studio perspective* is a collection of specialized views, editors, and tools to manage a given collection of resources. In the bottom-left pane, you can set up the connection between the System Manager and any InterChange Server instance in the network.

1. To add our new and local instance called ICS, right-click the **InterChange Server Instances** folder in the InterChange Server Component Management view and select **Register Server**.

   This opens the window shown in Figure 3-34 on page 82. Enter the Server name (`ICS`), the administrative user ID (`admin`), and its default password (`null`). Click **OK**.


Register new server

Server name: ICS

User name: admin

Password: ****

[✓] Save user ID and password

[ ] Local server

Local server installation path:

Figure 3-34 Register a server in System Manager

The System Manager connects to the InterChange Server and retrieves the names of deployed components in that InterChange Server instance. No components are deployed for now, so this results in an empty tree structure (Figure 3-35 on page 83).

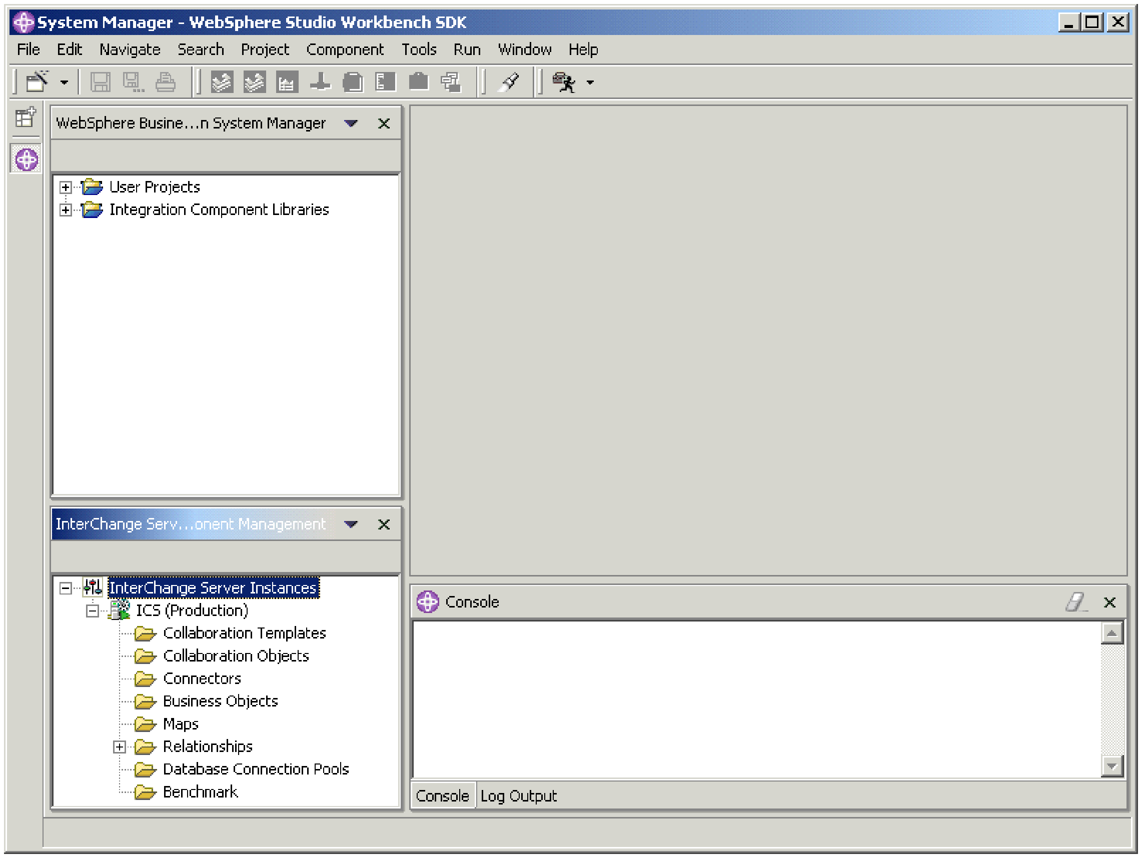
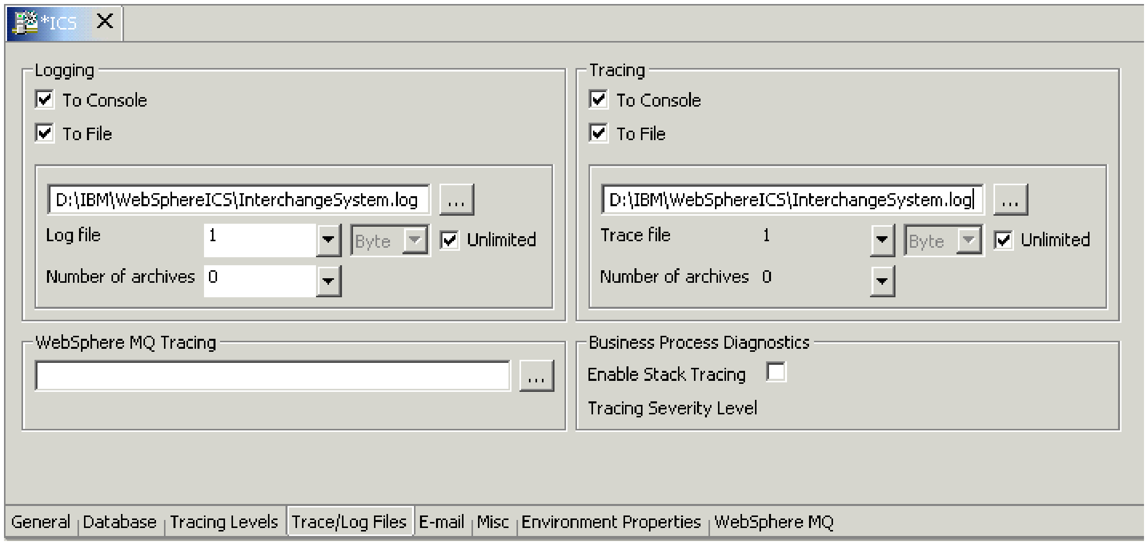


Figure 3-35 System Manager connected to InterChange Server

2. To further configure the InterChange Server, right-click the new instance and select **Edit Configuration** to open the system editor. We want to change the logging and tracing component in such a way that log messages are written to both the log file and the console window. Select the **Trace/Log Files** tab. Select the **To Console** and **To File** options for both Logging and Tracing. Enter the same file name for both types of information, as shown in Figure 3-36 on page 84.


*ICS

Logging
- [✓] To Console
- [✓] To File

D:\IBM\WebSphereICS\InterchangeSystem.log
Log file: 1 Byte [✓] Unlimited
Number of archives: 0

Tracing
- [✓] To Console
- [✓] To File

D:\IBM\WebSphereICS\InterchangeSystem.log
Trace file: 1 Byte [✓] Unlimited
Number of archives: 0

WebSphere MQ Tracing

Business Process Diagnostics
Enable Stack Tracing [ ]
Tracing Severity Level

Figure 3-36 Changing the logging and tracing components of InterChange Server

3. Save the changes and close the editor. Saving the changes results in a deployment process to the InterChange Server.

In previous releases of InterChange Server, the next step would have been to load the repository. However, in this release, the Load Repository shortcut no longer exists. The repos_copy utility is still there, but most standard objects are now provided as .jar files, which we can import in System Manager. From there, we can deploy one or more objects as a user project to the InterChange Server.

3.2.5 Using role-based access control

WebSphere InterChange Server Version 4.3 introduces a new feature called *role-based access control* (RBAC). Role-based access control provides the ability to create users, roles, and the authorization policies associated with those roles and, using those roles, to authorize permissions for users accessing the system.

Roles can be defined easily by the Administrator and assigned to a group of users, restricting access to key components only to verified users. Roles can be assigned along functional associations and greatly reduce the administrative burden. Assigning a role to a user or users permits them to access only the components of the system included in the role definition. Also, the use of RBAC functionality ensures that only an Administrator, or users with permission to administer roles, would be allowed to create users and assign roles. Role-based access control addresses access control and authentication.

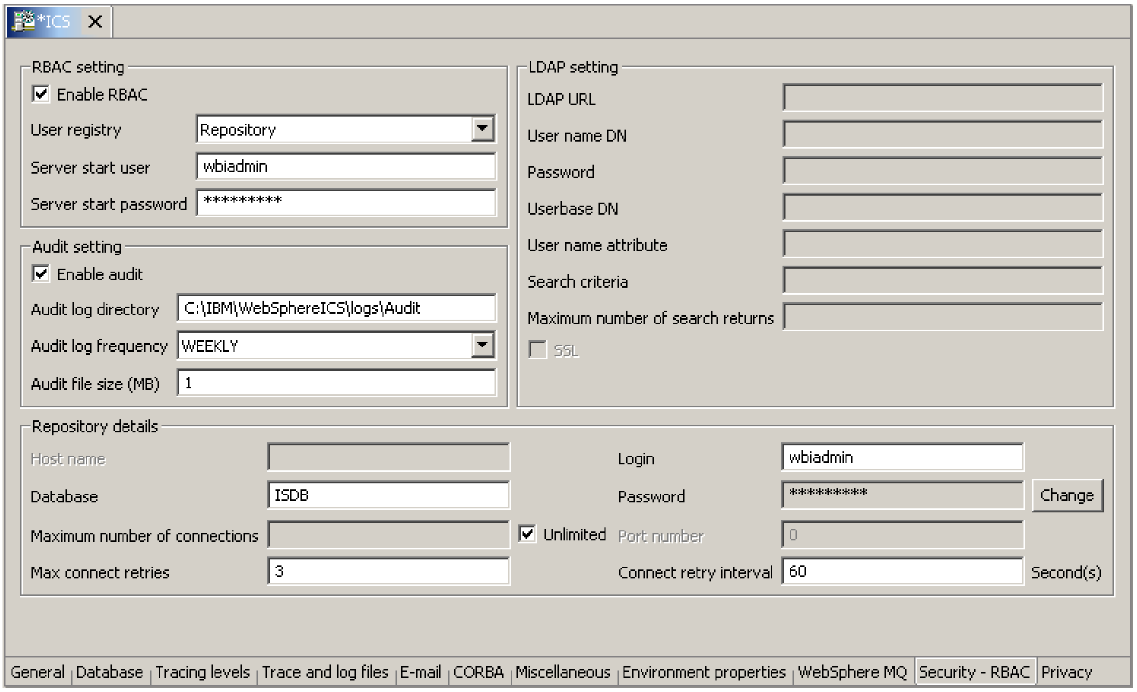
RBAC user information is stored in a user registry held in either a database or in a Lightweight Directory Access Protocol (LDAP) server. The roles and authorization policies are stored in the InterChange Server repository as they normally pertain to a particular server environment.

Steps for setting up RBAC

Before setting up RBAC, at least one user must be assigned the role of Administrator. If no user is assigned the Administrator role and RBAC is enabled, then InterChange Server, when restarted, will automatically disable RBAC.

Perform the following steps within the System Manager to set up role-based access control:

1. Right click the **InterChange Server name** in the Component Management view and select **Edit configuration**.
2. On the Security-RBAC tab, select the check box for **Enable RBAC** as seen in Figure 3-37.



RBAC setting
- [✓] Enable RBAC
- User registry: Repository
- Server start user: wbiadmin
- Server start password: *********

Audit setting
- [✓] Enable audit
- Audit log directory: C:\IBM\WebSphereICS\logs\Audit
- Audit log frequency: WEEKLY
- Audit file size (MB): 1

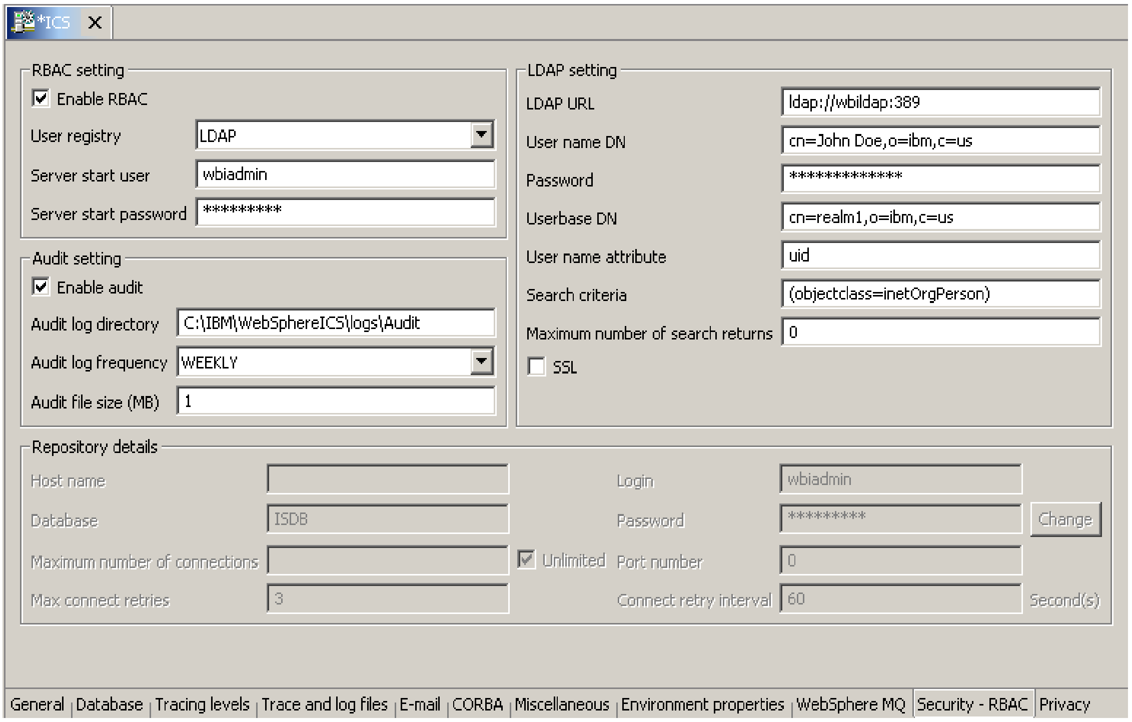
LDAP setting
- LDAP URL:
- User name DN:
- Password:
- Userbase DN:
- User name attribute:
- Search criteria:
- Maximum number of search returns:
- [ ] SSL

Repository details
- Host name:
- Database: ISDB
- Maximum number of connections: [✓] Unlimited
- Max connect retries: 3
- Login: wbiadmin
- Password: *********
- Port number: 0
- Connect retry interval: 60 Second(s)

Figure 3-37 Enable RBAC

3. Select the user registry to which to apply role-based access controls, that is, **Repository** or **LDAP**. We chose **Repository**.

4. In the Server Start User Name field, enter the user name to start the server.
5. In the Server Start Password field, enter the password associated with the user name.
6. If you selected **Repository**, enter details in the following fields as shown in the Repository details section in Figure 3-37:
   - Host name
   - Database
   - Port Number
   - User Name
   - Password
   - Max Connections, the maximum number of connections that a user can open
   - Max connect retries, the maximum number of times you can attempt to start a connection
   - Connect retry interval, the amount of time between connection retries
7. If you selected **LDAP**, enter details in the following fields as shown in Figure 3-38 on page 87:
   - LDAP URL, which is the URL of the LDAP installation
   - User name, which is the user account and is not case-sensitive
   - Password, which is the password for the user account
   - User base DN, which is the base distinguished name and acts as the root of all searches and updates
   - User name attribute, which the attribute in the schema that InterChange Server uses as a user name
   - Search criteria, which is the search criteria to use when retrieving LDAP users and is optional
   - Max search returns, which is the maximum number of entries returned from a search
   - SSL, which when set to True secures the connection using SSL protocol



RBAC setting

☑ Enable RBAC

User registry	LDAP
Server start user	wbiadmin
Server start password	*********

Audit setting

☑ Enable audit

Audit log directory	C:\IBM\WebSphereICS\logs\Audit
Audit log frequency	WEEKLY
Audit file size (MB)	1

LDAP setting

LDAP URL	ldap://wbildap:389
User name DN	cn=John Doe,o=ibm,c=us
Password	*************
Userbase DN	cn=realm1,o=ibm,c=us
User name attribute	uid
Search criteria	(objectclass=inetOrgPerson)
Maximum number of search returns	0

☐ SSL

Repository details

Host name		Login	wbiadmin
Database	ISDB	Password	*********
Maximum number of connections	☑ Unlimited	Port number	0
Max connect retries	3	Connect retry interval	60 Second(s)

Figure 3-38   RBAC LDAP configuration

Note: For details on configuring an LDAP server as a user registry for role-based access control for our scenario, see Appendix B, "Configuring LDAP for use with RBAC" on page 507.

8. To turn on audit settings, select the check box for **Enable Audit** and enter details in the following fields:
   - Audit log directory, which is the path of the audit log file
   - Audit log frequency, for example, Daily, Weekly or Monthly
   - Audit file size, which is the maximum size for the audit file in MB

How to deactivate RBAC

If it is necessary to deactivate RBAC on a particular InterChange Server, select the **Security-RBAC** tab of the InterChange Server configuration within the Systems Manager and *deselect* the **Enable RBAC** configuration property. This will cause the other configuration parameter fields to become grayed out.

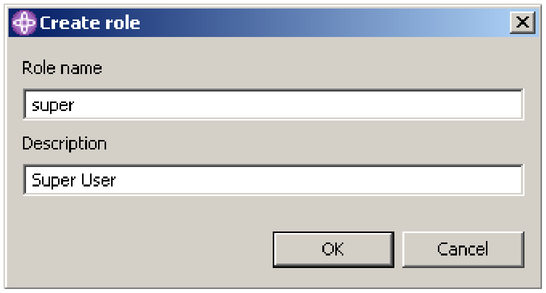
Administering roles

Role-based access control supports multiple users and enhanced security features based on roles. A role is a collection of users who share common administrative authority. Assigning authority policies into roles allows the administrator to work more effectively by reducing the burden on the administrator during the assignment of permissions.

Steps for creating roles

Perform the following steps to create a role:

1. Right-click the **InterChange Server name** in the Components Management view and select **Users and roles**. The view opens in the right upper pane.
2. Click the **Roles** tab at the bottom of the view.
3. Right-click the **Roles** tree and select **New role**. This displays the Create role dialog box as shown in Figure 3-39.



Role name: super

Description: Super User

Figure 3-39 Create RBAC role

4. Enter the role name. Note that once you name a role, it cannot be renamed.
5. Enter a role description, if necessary. *Role description is an optional field*.

Steps for deleting roles

Perform the following steps to delete a role:

1. Right-click the desired role.
2. Select **OK** to delete the role.

Note: Role names are case-sensitive. After a role is deleted, it cannot be restored without creating it as a new role. The role *administrator* is the default and cannot be deleted.

Administering users

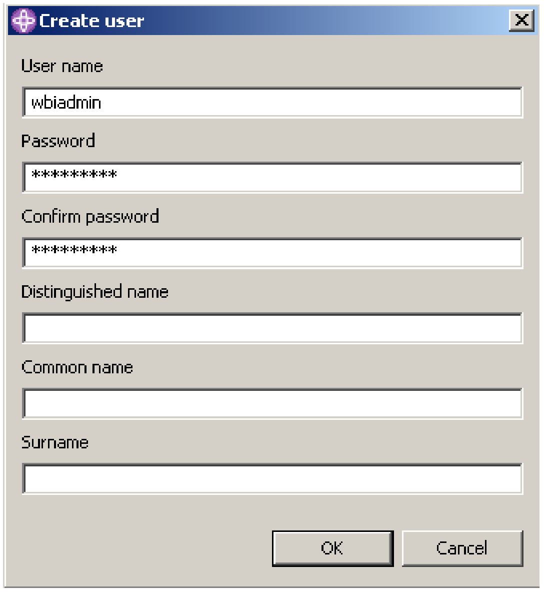
On the User and Roles Management screen, roles are listed downward in a tree directory display. You can assign a user to any number of roles. Users assigned to a role are listed in the tree directory beneath the role to which they are assigned, making for a quick and easy scan of permissions and responsibilities.

Additionally, you can import or export user information for use with the RBAC functionality.

Steps for adding users

Perform the following steps to add users to RBAC:

1. Right-click the **InterChange Server name** in the Components Management view and select **Users and roles**. The view opens in the right upper pane.
2. Right-click the **Users** tree and select **New user**. The Create User dialog window opens, as seen in Figure 3-40.



User name: wbiadmin
Password: *********
Confirm password: *********
Distinguished name:
Common name:
Surname:

Figure 3-40 Create RBAC User

3. In the User name field, enter the name of the user.
4. In the Password field, enter the password for the user.

Steps for deleting users

Perform the following steps to delete users from RBAC:

1. Right-click the user name and select **Delete user**.
2. Select **OK** to delete the user.

Note: `guest` is the only default user and cannot be deleted.

Steps for importing users and passwords

Perform the following steps to import users and passwords into RBAC:

1. Right-click the **InterChange Server name** in the Components Management view and select **Import** → **User Registry**. This displays the Import dialog box, where you specify the path for the binary file. This path should be valid on the server machine which is running the InterChange Server.
2. Select the file to import.

Note: When DATABASE is the user registry, support is available for importing users. However, this function is not supported for the LDAP user registry. It is recommended that you create a central user registry database or central LDAP registry, enabling multiple InterChange Server machines to use this central repository as opposed to transferring the user registry across various InterChange Server machines.

Steps for exporting users and passwords

Perform the following steps to export users and passwords into RBAC:

1. Right-click the **InterChange Server name** in the Components Management view and select **Export** → **User Registry**. This displays the Export dialog box, where you can specify the file path.
2. Select the destination for the file to export. This path should be valid on the server machine which is running the InterChange Server.

Note: When DATABASE is the user registry, support is available for exporting users. However, this function is not supported for the LDAP user registry. It is recommended that you create a central user registry database or central LDAP registry, enabling multiple InterChange Server machines to use this central repository as opposed to transferring the user registry across various InterChange Server machines.

Administering user and role assignments

Assigning roles to the available users greatly reduces the burden upon the administrator to assign individual permissions to vital functionality. Users can be assigned to numerous roles, all regulated by the user's login ID. Users assigned to a role are listed in the tree directory beneath the role to which they are assigned.

Steps for assigning roles to users

Perform the following steps to assign roles to users:

1. If the Users and roles view is no longer active, right-click the **InterChange Server name** in the Components Management view and select **Users and roles**. The view opens in the right upper pane.
2. Locate the desired user in the **Users** tree and right-click the **user name** and select **Add roles**. This displays the Add Role dialog box, which lists all available roles as shown in Figure 3-41.

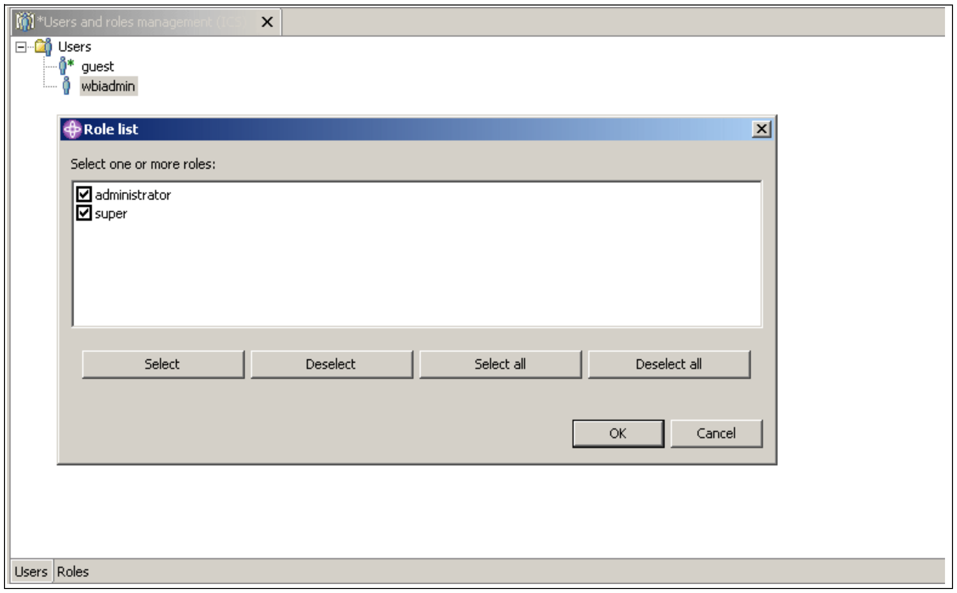


Figure 3-41 Add roles to RBAC user

3. Select single or multiple roles to assign to the user and click **OK**.

Steps for removing users from roles

Perform the following steps to remove users from the roles listing:

1. Expand the Users tree and right-click the role displayed under the desired user. Select **Remove role**.
2. Select **OK** to remove the role.

Administering security policy permissions

As an administrator, you can assign permissions to default roles within RBAC. These security policies are listed in a tree directory, along with the operations that each role is allowed to access.

The operations that can be secured on a server are shown in Table 3-2.

Table 3-2 Securable server operations

Securable component	**Access-controlled operations**
Server	► Start ► Shutdown ► Security/Administering users/roles ► Monitoring ► View Failed Events ► Deploy ► Export ► Delete ► Compile ► Export config files ► Deploy config files
Collaboration Templates	► Compile
Collaboration Objects	► Start ► Stop ► Pause ► Shutdown ► Execute (AccessFramework call) ► Resolve transactional status ► Submit Failed Events ► Delete Failed Events ► Cancel LLBP flow
Connectors	► Start ► Stop ► Pause ► Shutdown Agent ► Submit Failed Events ► Delete Failed Events

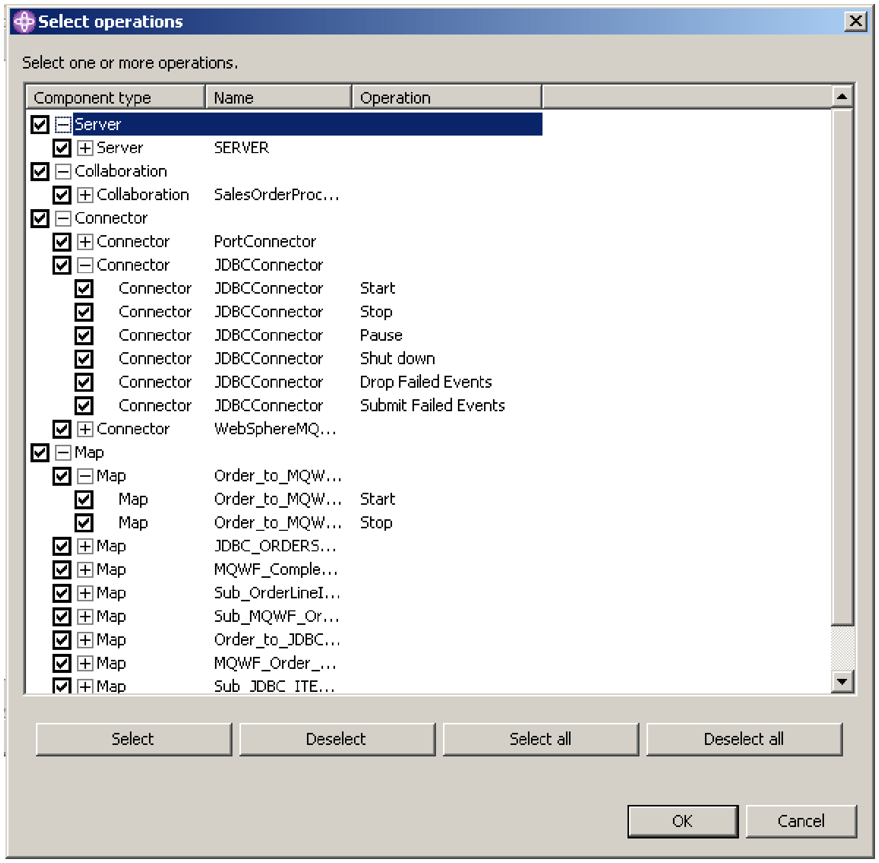
Securable component	Access-controlled operations
Business Objects	
Maps	► Compile ► Start ► Stop
Relationships	► Start ► Stop
BenchMark	► Start ► Stop
Scheduler	
DBConnectionCache	

Steps for assigning operations to roles

Perform the following steps to assign operations to roles:

1. Right-click the **InterChange Server name** in the Components Management view and select **Security policy**. The view opens in the right upper pane.
2. Locate the desired role in the **Roles** tree, right-click the role, and select **Grant access**. This displays the Select Operations dialog box, which lists all available operations as shown in Figure 3-42 on page 94.
3. Select single or multiple operations to assign to the role.
4. Click **OK** to create the policy.

Tip: Changes to security policies are not automatically saved to the repository database. Make sure to save the changes before or while exiting the view.


Select operations

Select one or more operations.

Component type	Name	Operation
☑ ⊟ Server		
☑ ⊞ Server	SERVER	
☑ ⊟ Collaboration		
☑ ⊞ Collaboration	SalesOrderProc...	
☑ ⊟ Connector		
☑ ⊞ Connector	PortConnector	
☑ ⊟ Connector	JDBCConnector	
☑ Connector	JDBCConnector	Start
☑ Connector	JDBCConnector	Stop
☑ Connector	JDBCConnector	Pause
☑ Connector	JDBCConnector	Shut down
☑ Connector	JDBCConnector	Drop Failed Events
☑ Connector	JDBCConnector	Submit Failed Events
☑ ⊞ Connector	WebSphereMQ...	
☑ ⊟ Map		
☑ ⊟ Map	Order_to_MQW...	
☑ Map	Order_to_MQW...	Start
☑ Map	Order_to_MQW...	Stop
☑ ⊞ Map	JDBC_ORDERS...	
☑ ⊞ Map	MQWF_Comple...	
☑ ⊞ Map	Sub_OrderLineI...	
☑ ⊞ Map	Sub_MQWF_Or...	
☑ ⊞ Map	Order_to_JDBC...	
☑ ⊞ Map	MQWF_Order_...	
☑ ⊞ Map	Sub_JDBC_ITE...	

Figure 3-42 Select operations for role

Administering membership and security policy information

Administrators can import membership and security policy information to be used with the RBAC functionality from any authorized server. Conversely, membership and security policy information can also be exported to a file for use on an additional server or for storage.

Importing membership and security policy information

Perform the following steps to import membership or security policy information:

1. Right-click the **InterChange Server name** in the Components Management view and select **Import** → **Role and Security Policy**. This displays the Import dialog box, where you specify the path for the binary file. This path should be valid on the machine that is running InterChange Server.

2. Select the file to import. If you import information when the User/Roles Management view is active, the changes will not display until you close and reopen the view.

Note: You can also import information using `repos_copy` with the -xmsp option. For information about using `repos_copy`, refer to "Using repos_copy" in the *IBM WebSphere InterChange Server System Administration Guide Version 4.3.0*, (30 September 2004).

Exporting membership and security policy information

Perform the following steps to export membership or security policy information:

1. Right-click the **InterChange Server name** in the Components Management view and select **Export** → **Role and Security Policy**. This displays the Export dialog box, where you can specify the file path.
2. Select the destination for the file to export. This path should be valid on the machine that is running InterChange Server.

Note: You can also import information using `repos_copy` with the -xmsp option. For information about using `repos_copy`, refer to "Using repos_copy" in the *IBM WebSphere InterChange Server System Administration Guide Version 4.3.0*, (30 September 2004).

Administering the RBAC password

Each user in RBAC has an associated password. When a user logs in to the server, the password is used to verify the roles assigned to that user. Occasionally it might become necessary to change or reset the user password. Perform the following steps to reset the user password:

1. Right-click the **InterChange Server name** in the Components Management view and select **Reset password**.
2. In the Reset Password dialog box, select the user name and enter the new password in both the New Password field and the Confirm Passworld field.
3. Click the **Reset Password** button to reset the password.

Security Administration

As an administrator, you can monitor the use of the roles in RBAC using the security administration functionality. InterChange Server lists active users in a table which displays user name, session ID, and the amount of time the user has spent logged onto the server. It also allows an administrator to force the logoff of users.

To display the Security Administration panel, right-click the **InterChange Server name** in the Components Management view and select **Security Administration**.

Tip: It is recommended that you refresh the user listing occasionally to retain an accurate user display.

3.3 Installing WebSphere Business Integration Adapters

The WebSphere Business Integration Adapters are an integral part of an integration solution. In our implementation, InterChange Server uses the adapters to communicate with external applications and technologies, thus it is important to discuss how to install these components successfully.

Our implementation involves installing the adapters on the same machine as the InterChange Server and communicating only within the local network. But, there are actually three ways in which you can configure and install an adapter within the local network:

- On the same machine as the integration broker.
- On a machine that hosts the target application.
- On a machine dedicated for one or more adapters that has access to both the target application and InterChange Server.

The WebSphere Business Integration Adapters are each standalone entities that rely upon the Adapter Framework for communication with InterChange Server.

The following order must be used when installing individual adapters.

1. Install the Adapter Framework. This only needs to be performed once for each server on which the adapters will be running.
2. Install any required data handlers. This only needs to be performed once for each server on which the adapters will be running.
3. Install the needed adapter.

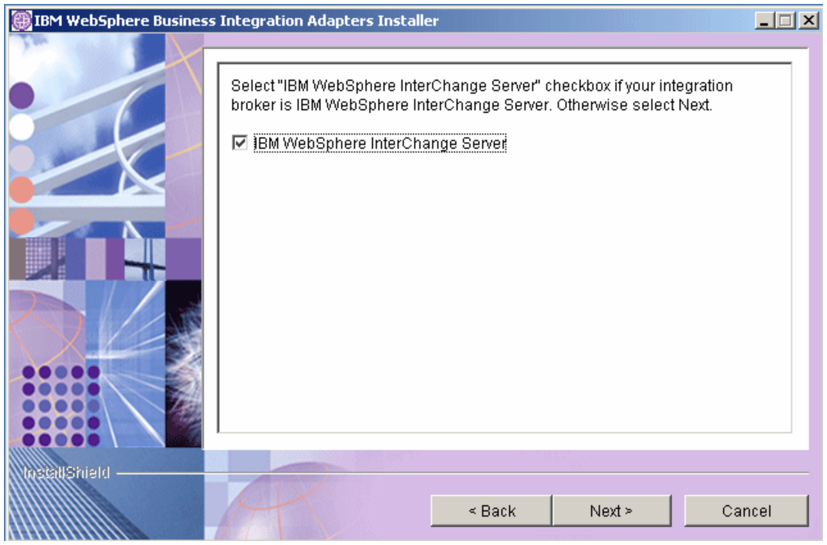
Note: In WebSphere InterChange Server V4.3.0 the Adapter Framework is not automatically installed. You must install it manually.

In Versions 4.2.2 and previous, the Adapter Framework was installed automatically and it was very important to not install the Adapter Framework again because it would overwrite critical InterChange Server files.

Installing WebSphere BI Adapter Framework

The installation of the Adapter Framework is a standard administrator's task. When the installation program is started, the usual steps about language selection and license acceptance are presented.

1. Check **IBM WebSphere InterChange Server** to indicate that InterChange Server is the integration broker to be used with the WebSphere Business Integration Adapters. (Figure 3-43)

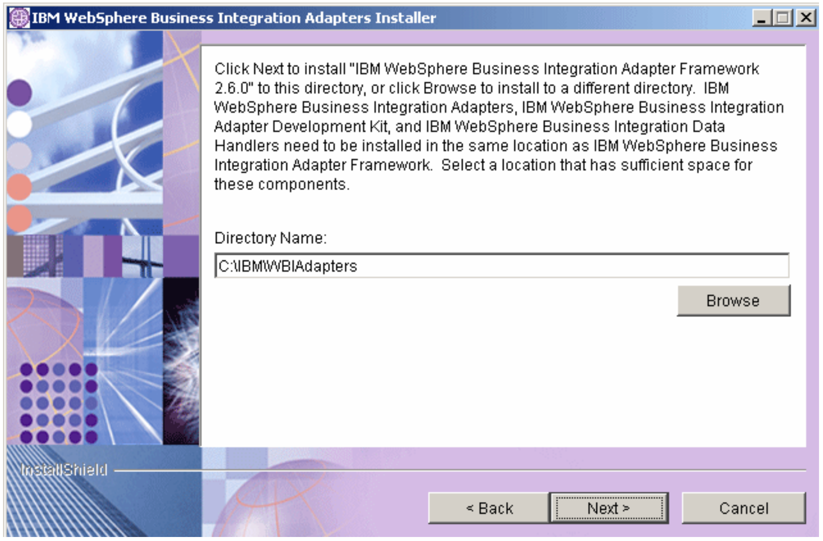


Select "IBM WebSphere InterChange Server" checkbox if your integration broker is IBM WebSphere InterChange Server. Otherwise select Next.

[✓] IBM WebSphere InterChange Server

Figure 3-43 Select broker for WebSphere Business Integration Adapters

2. Select the install location, which must be different than the base InterChange Server install directory (Figure 3-44).



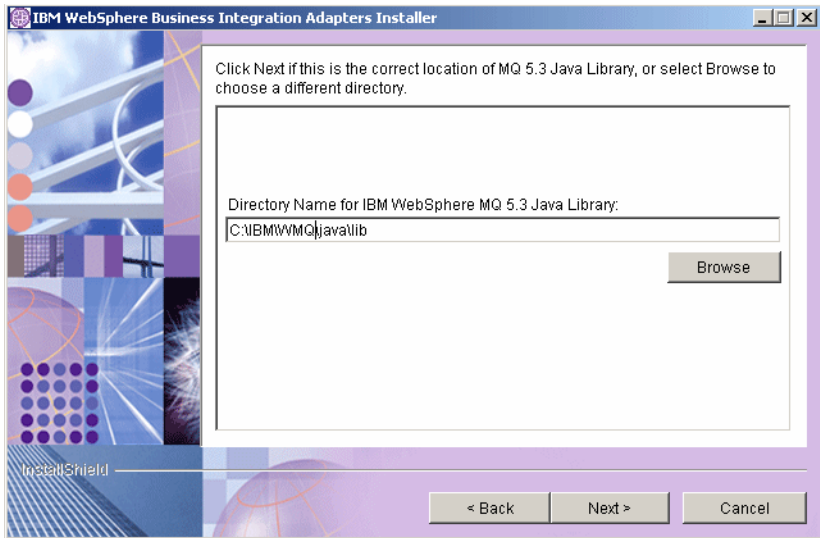
Click Next to install "IBM WebSphere Business Integration Adapter Framework 2.6.0" to this directory, or click Browse to install to a different directory. IBM WebSphere Business Integration Adapters, IBM WebSphere Business Integration Adapter Development Kit, and IBM WebSphere Business Integration Data Handlers need to be installed in the same location as IBM WebSphere Business Integration Adapter Framework. Select a location that has sufficient space for these components.

Directory Name:

C:\IBMWBIAdapters

Figure 3-44 Select Adapter Framework install location

3. The next step in the installation is to verify the location of the WebSphere MQ Java libraries. The value should be filled in correctly, but if it is not, modify the directory name to point to the correct location (Figure 3-45).



Click Next if this is the correct location of MQ 5.3 Java Library, or select Browse to choose a different directory.

Directory Name for IBM WebSphere MQ 5.3 Java Library:

C:\IBM\WMQ\java\lib

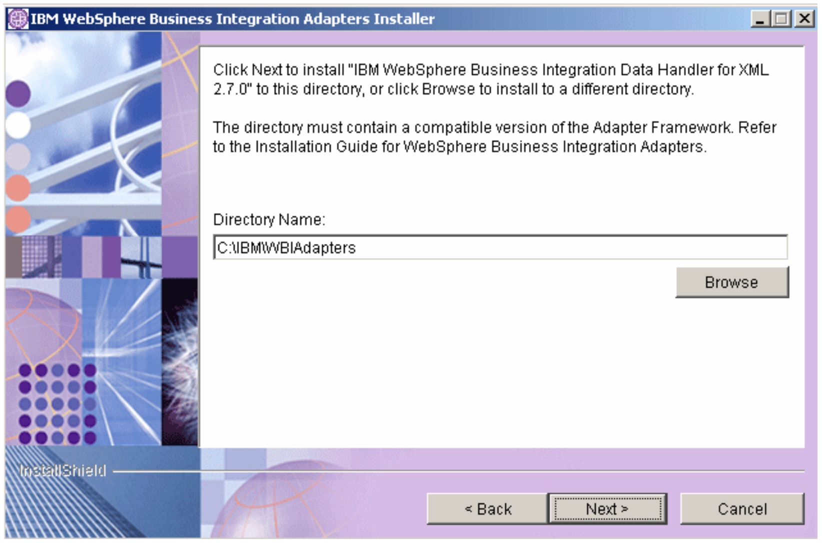
Figure 3-45 WebSphere MQ Java libraries location

4. Complete the installation by selecting **Next** on the final summary screen.

Installing data handlers

Data handlers are the components that allow for the translation of serialized data streams to and from the hierarchical structure of business objects. Particular adapters require the use of data handlers. A commonly used data handler is the XML data handler, the installation of which is described here.

The installation of data handlers is very simple, thus only one screen from the installation is shown. Verify the location of the already installed Adapter Framework (Figure 3-46).


IBM WebSphere Business Integration Adapters Installer

Click Next to install "IBM WebSphere Business Integration Data Handler for XML 2.7.0" to this directory, or click Browse to install to a different directory.

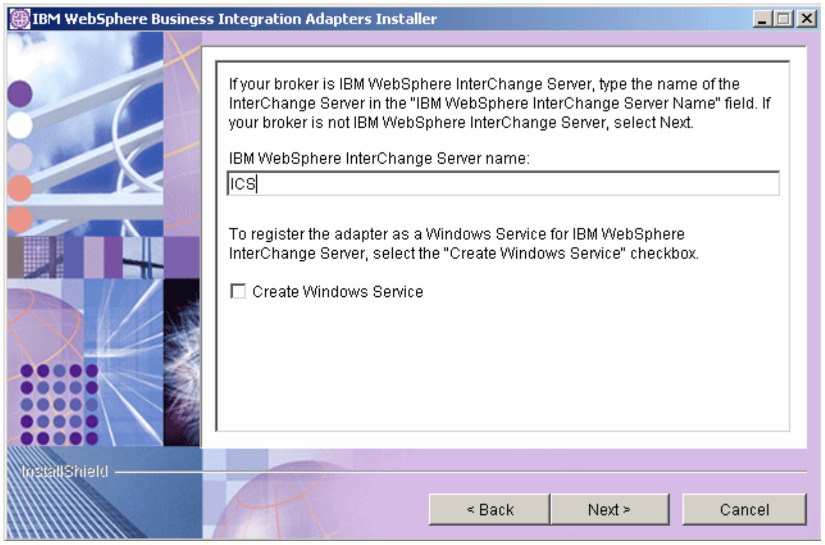
The directory must contain a compatible version of the Adapter Framework. Refer to the Installation Guide for WebSphere Business Integration Adapters.

Directory Name:
C:\IBM\WBIAdapters

Figure 3-46 Set the target location for the XML data handler

Installing the JDBC Adapter

Installing the JDBC Adapter is very similar to installing the Adapter Framework and the data handlers. There is one additional install screen, as seen in Figure 3-47, where the InterChange Server name is specified.



If your broker is IBM WebSphere InterChange Server, type the name of the InterChange Server in the "IBM WebSphere InterChange Server Name" field. If your broker is not IBM WebSphere InterChange Server, select Next.

IBM WebSphere InterChange Server name: ICS

To register the adapter as a Windows Service for IBM WebSphere InterChange Server, select the "Create Windows Service" checkbox.

☐ Create Windows Service

Figure 3-47 Provide InterChange Server name

Note: The installation of additional adapters involves the same procedures as outlined in "Installing the JDBC Adapter" on page 100. For the scenario solutions described later in this book, we also installed the WebSphere MQ Workflow Adapter.

3.4 WebSphere BI Message Broker installation and configuration

For our business integration infrastructure, we installed WebSphere Business Integration Message Broker on a Microsoft Windows 2000 server on which we configured the configuration manager and broker.

As Figure 3-48 on page 101 shows, both the broker and the configuration manager have their own queue manager. The broker database is created on the local server, and the configuration manager's database is created on the database server that also hosts the databases for WebSphere MQ Workflow and the InterChange Server.

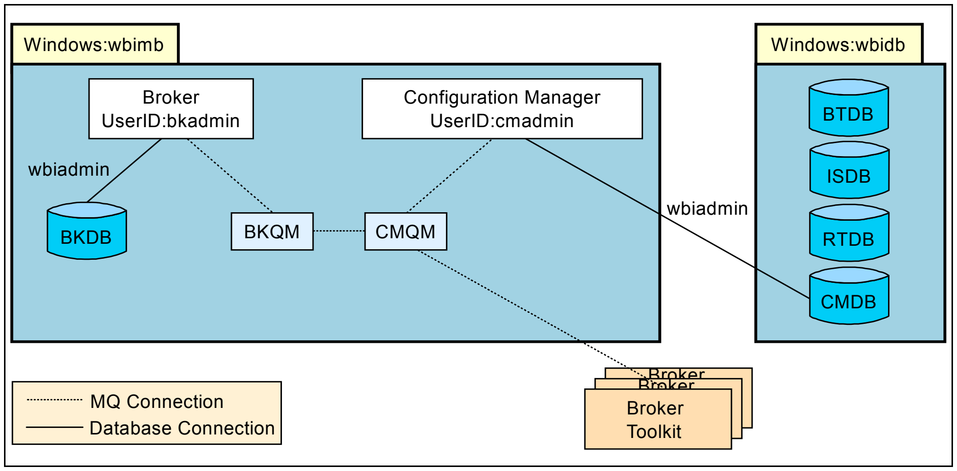


Figure 3-48 WebSphere Business Integration Message Broker configuration

Figure 3-48 also specifies the user IDs that will be used. The broker runs with user ID bkadmin, and the configuration manager logs on with user ID cmadmin. Database access is performed with user ID wbiadmin, which was used earlier to access the WebSphere MQ Workflow and InterChange Server database. All these user IDs are required to have Administrator access rights. Note that having specific user IDs for each component is not required.

3.4.1 Prerequisite software

The following components must be installed prior to installing WebSphere Business Integration Message Broker:

- WebSphere MQ Version 5.3 CSD 7.
- DB2 Version 8.1 FixPack 2.
- Microsoft Data Access Components (MDAC) is provided on the product CD.
- IBM Remote Access Agent,provided on the product CD

  This component is required for remote access to the broker.

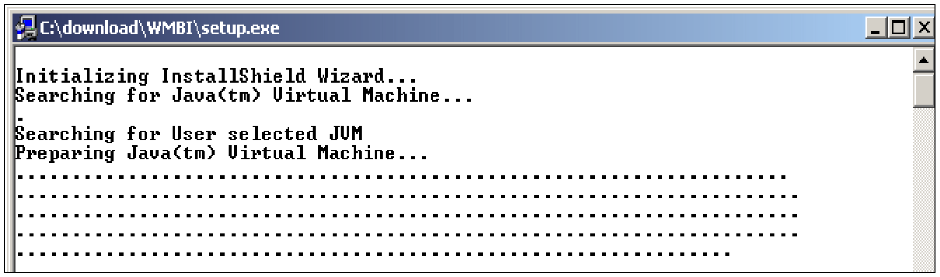
Refer to the corresponding installation guide if installation assistance is needed.

3.4.2 Installation

WebSphere Business Integration Server V4.3 includes an updated version of WebSphere Business Integration Message Broker (V5.0.1), which includes the first service pack for the WebSphere Business Integration Message Broker.

Installing this updated version is slightly different from installing Version 5.0. When using Version 5.0, you have to define user groups manually. In Version 5.0.1, a graphical utility is provided to perform this task, and this utility is launched immediately after the installation completes.

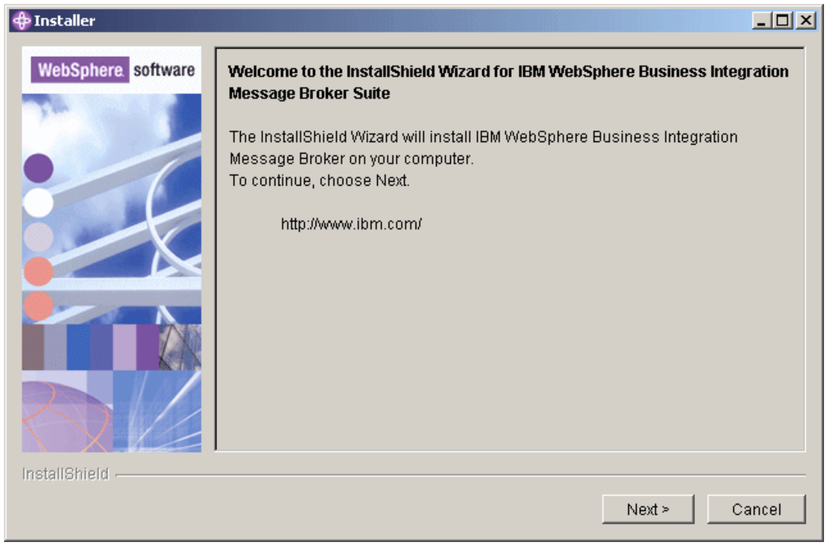
1. The installation of this component of WebSphere Business Integration Server begins with preparing the Java Virtual Machine environment (Figure 3-49).


C:\download\WMBI\setup.exe

```
Initializing InstallShield Wizard...
Searching for Java(tm) Virtual Machine...
.
Searching for User selected JVM
Preparing Java(tm) Virtual Machine...
.....................................................................
......................................................................
......................................................................
......................................................................
................................................................
```

Figure 3-49 Prepares Java Virtual Machine for the installation

2. The installer wizard provides a language selection screen. Select a language and click **OK**.
3. This is followed by a welcome screen and a license acceptance windows.
4. The installation program displays the screen in Figure 3-50. Click **Next** to proceed with the installation.



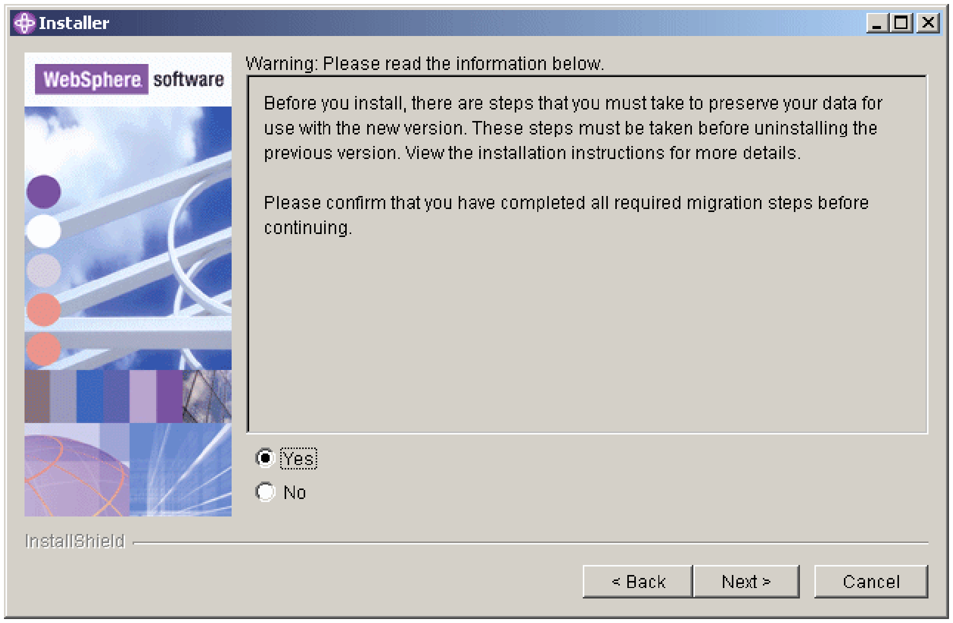
Welcome to the InstallShield Wizard for IBM WebSphere Business Integration Message Broker Suite

The InstallShield Wizard will install IBM WebSphere Business Integration Message Broker on your computer.
To continue, choose Next.

http://www.ibm.com/

Figure 3-50 The Install Shield Wizard will install Message Broker on your computer

5. The installation program asks you to confirm that you have performed the migration steps. (This would be the case where you are installing the latest version of the broker on top of an existing V2.x installation.) If you are performing a migration, refer to *Migration to WebSphere Business Integration Message Broker V5*, SG24-6995. The screen in Figure 3-51is displayed. Click **Yes** and **Next** if you are doing a fresh installation of WebSphere Business Integration Message Broker V5 or you have completed the necessary migration steps.



Warning: Please read the information below.

Before you install, there are steps that you must take to preserve your data for use with the new version. These steps must be taken before uninstalling the previous version. View the installation instructions for more details.

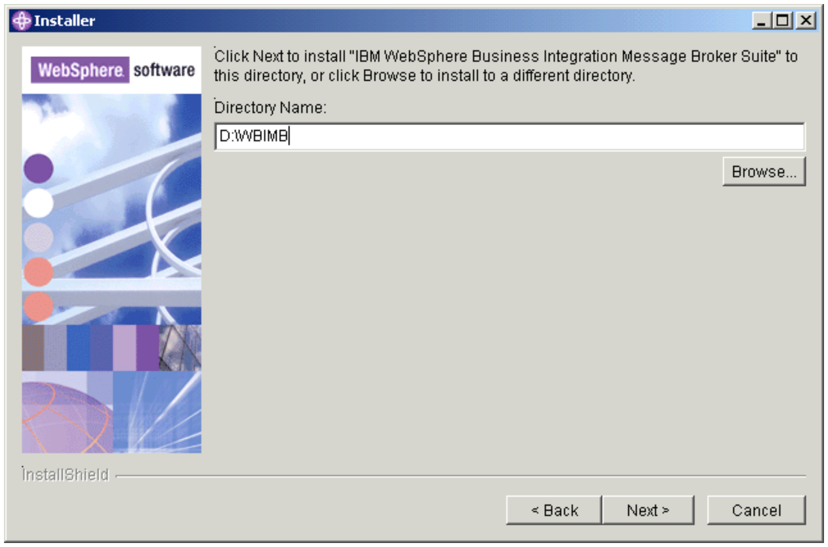
Please confirm that you have completed all required migration steps before continuing.

[✓ Yes] No

Figure 3-51 Migration step confirmation

6. This is followed by the software license screen. Click **Next** if you agree with the license agreement.

7. You are then asked to designate an installation directory (Figure 3-52). Do so and select **Next**.

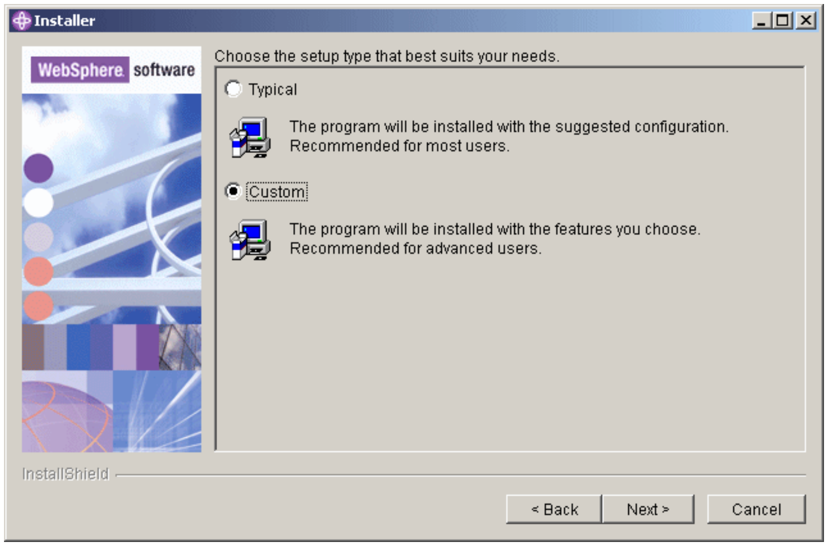


Click Next to install "IBM WebSphere Business Integration Message Broker Suite" to this directory, or click Browse to install to a different directory.

Directory Name: D:\WBIMB

Figure 3-52 Provide an installation directory for Message Broker

8. Select the Custom installation option (Figure 3-53).



Choose the setup type that best suits your needs.

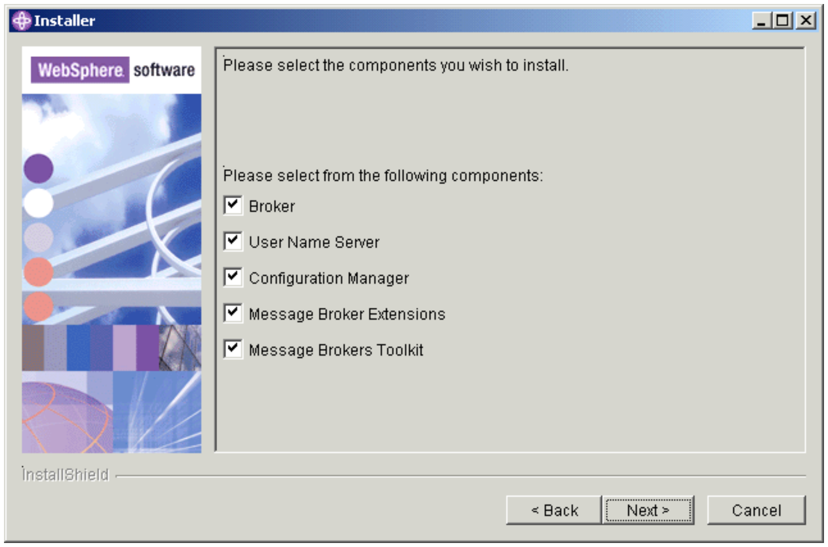
Typical [✓ Custom]

Typical: The program will be installed with the suggested configuration. Recommended for most users.

Custom: The program will be installed with the features you choose. Recommended for advanced users.

Figure 3-53 Choose the setup type that best suits your needs

9. Select the appropriate components, as shown in Figure 3-54. Select **Next**.



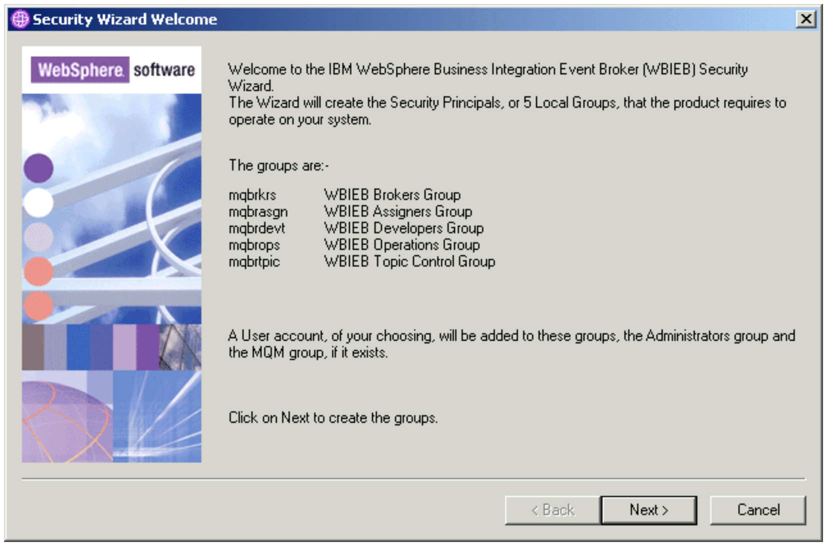
Please select the components you wish to install.

Please select from the following components:

- [✓] Broker
- [✓] User Name Server
- [✓] Configuration Manager
- [✓] Message Broker Extensions
- [✓] Message Brokers Toolkit

Figure 3-54 Select the components you wish to install

10. When the installation is finished, a new tool is launched - the Security Wizard (Figure 3-55 on page 106). This new tool automates the task of creating user groups and assigning a user ID to the broker process. You can run this tool at this time, or end it and run it later. Click **Next** to continue and create the groups.


Security Wizard Welcome

Welcome to the IBM WebSphere Business Integration Event Broker (WBIEB) Security Wizard.
The Wizard will create the Security Principals, or 5 Local Groups, that the product requires to operate on your system.

The groups are:-

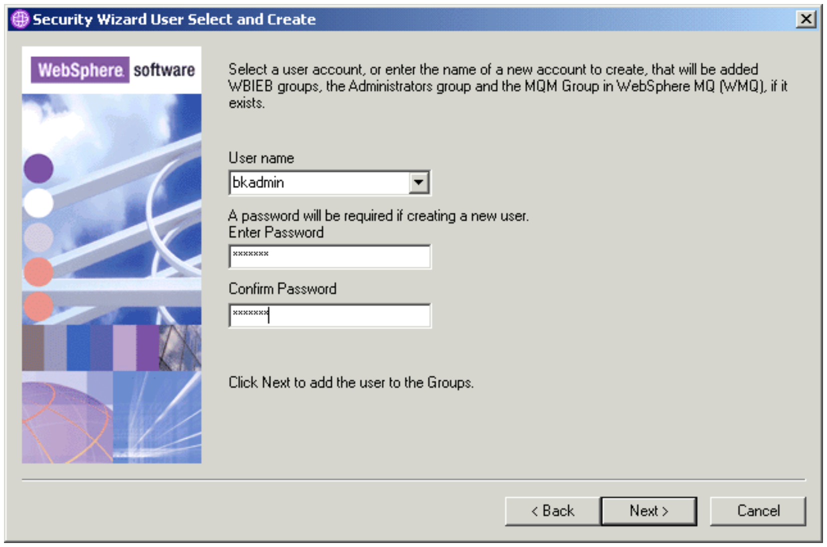
mqbrkrs	WBIEB Brokers Group
mqbrasgn	WBIEB Assigners Group
mqbrdevt	WBIEB Developers Group
mqbrops	WBIEB Operations Group
mqbrtpic	WBIEB Topic Control Group

A User account, of your choosing, will be added to these groups, the Administrators group and the MQM group, if it exists.

Click on Next to create the groups.

Figure 3-55 Security Wizard - create the local groups that Message Broker requires

11. The Security Wizard enables you to select an existing user ID to be used by the broker or to create a new user ID, and to provide a new password for this user ID (Figure 3-56).


Security Wizard User Select and Create

Select a user account, or enter the name of a new account to create, that will be added WBIEB groups, the Administrators group and the MQM Group in WebSphere MQ (WMQ), if it exists.

User name
bkadmin

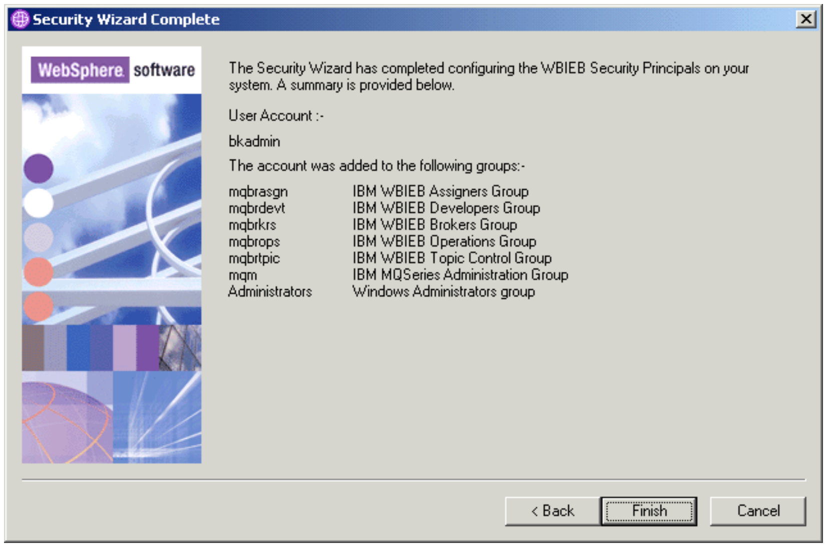
A password will be required if creating a new user.
Enter Password
****

Confirm Password
****

Click Next to add the user to the Groups.

Figure 3-56 Create a new user ID for the broker

12. This new user ID is created with all of the required authorities and group memberships, as shown in Figure 3-57.



The Security Wizard has completed configuring the WBIEB Security Principals on your system. A summary is provided below.

User Account :-

bkadmin

The account was added to the following groups:-

mqbrasgn	IBM WBIEB Assigners Group
mqbrdevt	IBM WBIEB Developers Group
mqbrkrs	IBM WBIEB Brokers Group
mqbrops	IBM WBIEB Operations Group
mqbrtpic	IBM WBIEB Topic Control Group
mqm	IBM MQSeries Administration Group
Administrators	Windows Administrators group

Figure 3-57 New user ID created

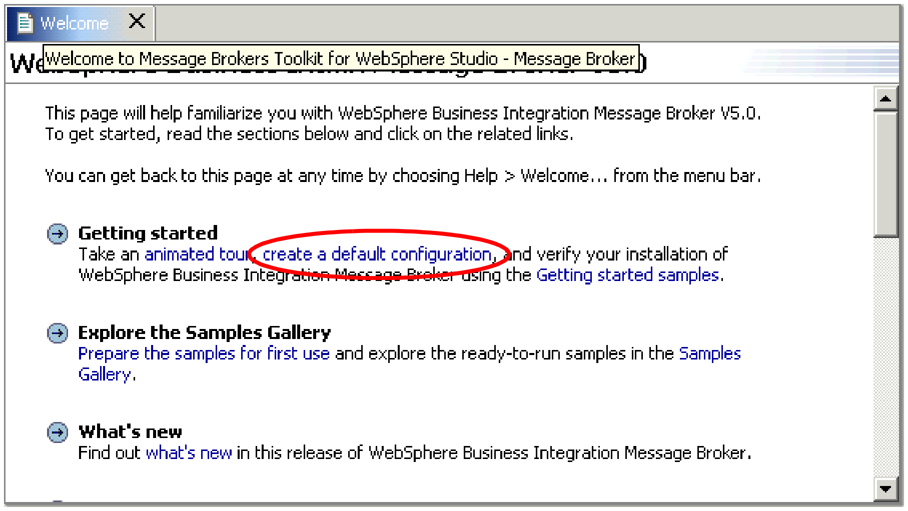
13. To match our graphical overview of the WebSphere Business Integration Message Broker configuration, we also create the user ID `cmadmin`, which will be used by the configuration manager. Also, you should consider setting the password for both user IDs as **nonexpiring**.

3.4.3 Create the WebSphere BI Message Broker infrastructure

To create the WebSphere Business Integration Message Broker infrastructure, we create the databases, WebSphere MQ-related components, the configuration manager and the broker.

These tasks can all be accomplished by using the new Getting Started Wizard, which you access by starting the Broker Toolkit with the Start menu.

When the toolkit is started for the first time, it might have to complete a few installation tasks, after which the Welcome view is shown. When the toolkit is not started for the first time, it might not display this Welcome view. To open it again, select **Help** → **Welcome** and select the Welcome view for the Message Broker (Figure 3-58 on page 108).


Welcome

Welcome to Message Brokers Toolkit for WebSphere Studio - Message Broker

This page will help familiarize you with WebSphere Business Integration Message Broker V5.0. To get started, read the sections below and click on the related links.

You can get back to this page at any time by choosing Help > Welcome... from the menu bar.

Getting started
Take an animated tour, create a default configuration, and verify your installation of WebSphere Business Integration Message Broker using the Getting started samples.

Explore the Samples Gallery
Prepare the samples for first use and explore the ready-to-run samples in the Samples Gallery.

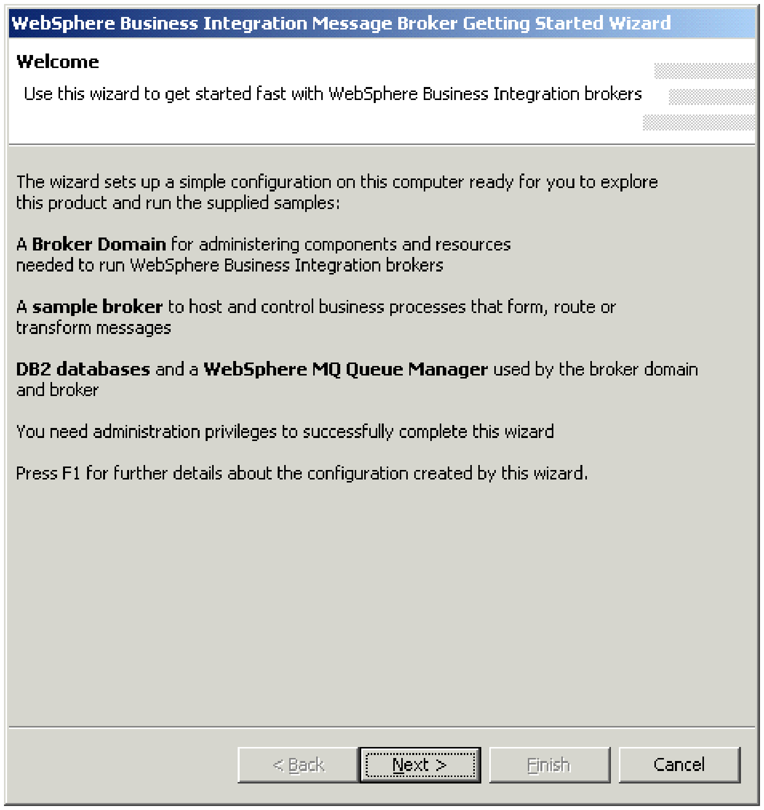
What's new
Find out what's new in this release of WebSphere Business Integration Message Broker.

Figure 3-58 Welcome window in Broker Toolkit

This Getting Started Wizard is well-suited for installations in which all components are on a single system. Figure 3-59 on page 109 shows the tasks that the wizard will complete:

- Create local databases.
- Create a queue manager shared by the broker and configuration manager.
- Create the broker and the configuration manager.

The wizard does not enable you to use or create a remote database.



WebSphere Business Integration Message Broker Getting Started Wizard

Welcome

Use this wizard to get started fast with WebSphere Business Integration brokers

The wizard sets up a simple configuration on this computer ready for you to explore this product and run the supplied samples:

A **Broker Domain** for administering components and resources needed to run WebSphere Business Integration brokers

A **sample broker** to host and control business processes that form, route or transform messages

DB2 databases and a **WebSphere MQ Queue Manager** used by the broker domain and broker

You need administration privileges to successfully complete this wizard

Press F1 for further details about the configuration created by this wizard.

Figure 3-59 Using the Getting Started Wizard

The following sections walk you through a manual configuration process to create the different broker components and their required resources.

Create and connect to databases

Figure 3-48 on page 101 shows that we are using a remote database for the configuration manager. To create a database, we can either:

- Log on to the database server, create the database locally, and catalog the existing database on the machine used by the configuration manager.
- Catalog the remote system on the broker machine and create the database from the client machine. These commands in Example 3-3 on page 109 perform this option (explanations follow):

Example 3-3 Creating the database remotely

```
CATALOG TCPIP NODE WBIDB REMOTE <server name> SERVER 50000 REMOTE_INSTANCE
WBIDB SYSTEM <machine name> OSTYPE  WIN
```

```
db2 attach to wbidb user db2admin using sas313r
db2 create database CMDB
db2 connect to CMDB user db2admin using sas313r
db2 GRANT DBADM, CREATETAB, BINDADD, CONNECT, CREATE_NOT_FENCED_ROUTINE,
IMPLICIT_SCHEMA, LOAD, CREATE_EXTERNAL_ROUTINE, QUIESCE_CONNECT ON DATABASE
TO USER WBIADMIN
db2 catalog system odbc data source CMDB
```

a. First, the remote system is cataloged.

b. Next, we attach to the remote database instance using the user ID db2admin, which is the database instance owner.

c. The next step creates the actual database on the remote server. When the database is created, we connect to it as the database instance owner. This enables you to give the necessary authorities to the user ID wbiadmin.

d. Finally, the database is cataloged as a system ODBC data source.

Similar commands are required to create the broker database on the local database instance (Example 3-4):

Example 3-4 Create the database locally

```
db2 attach to db2 user db2admin using sas313r
db2 create database BKDB
db2 connect to BKDB user db2admin using sas313r
db2 GRANT DBADM, CREATETAB, BINDADD, CONNECT, CREATE_NOT_FENCED_ROUTINE,
IMPLICIT_SCHEMA, LOAD, CREATE_EXTERNAL_ROUTINE, QUIESCE_CONNECT ON DATABASE TO
USER WBIADMIN
db2 catalog system odbc data source BKDB
```

Instead of using the DB2 command window, you can also use the DB2 Control Center to perform these tasks.

To verify that the user ID wbiadmin can log on to these databases using the ODBC interface, you can use the Data Sources utility in the Windows Control Panel’s Administrative Tools folder.

WebSphere MQ configuration

Before we create the broker and the configuration manager we create the WebSphere MQ infrastructure. This includes:

- Creating queue managers with the appropriate logging options.
- Creating listeners.
- Creating a channel initiator.
- Configuring channels between the queue managers.

These tasks can be performed using either a graphical tool, such as WebSphere MQ Explorer, or with the command line (explanations follow):

```
crtmqm -u SYSTEM.DEAD.LETTER.QUEUE -lf 2048 -lp 5 BKQM
amqmdain crtlsr BKQM tcp 1414
amqmdain auto BKQM
```

The first of these commands creates the queue manager with an associated dead letter queue, and its -lp and -lf parameters increase the logging capacity for the broker. The second command creates a TCP/IP listener for this queue manager on port 1414. The third command sets the start-up mode of the queue manager to automatic.

Nearly identical commands can be used to create the queue manager supporting the configuration manager.

```
crtmqm -u SYSTEM.DEAD.LETTER.QUEUE -lf 2048 -lp 5 CMQM
amqmdain crtlsr CMQM tcp 1415
amqmdain auto CMQM
```

Table 2-5 on page 42 shows the options that are used to create these queue managers.

To enable communication between the configuration manager and the broker, we set up channels and transmission queues between the two queue managers. You can use the runmqsc utility to create these objects or use WebSphere MQ Explorer. The commands in Example 3-5 are used for the queue manager BKQM:

Example 3-5 Creating transmission queues in BKQM

```
def chl(BKQM.TO.CMQM) chltype(sdr) trptype(tcp) conname(‘wbimb(1415)’)
xmitq(CMQM)
def chl(CMQM.TO.BKQM) chltype(rcvr)
def ql(CMQM) usage(xmitq) trigger trigtype(first) trigdata(BKQM.TO.CMQM)
initq(SYSTEM.CHANNEL.INITQ)
```

The commands in Example 3-6 are for the queue manager CMBK, and values in these commands match the commands for BKQM:

Example 3-6 Creating transmission queues in CMBK

```
def chl(CMQM.TO.BKQM) chltype(sdr) trptype(tcp) conname(‘wbimb(1414)’)
xmitq(BKQM)
def chl(BKQM.TO.CMQM) chltype(rcvr)
def ql(BKQM) usage(xmitq) trigger trigtype(first) trigdata(CMQM.TO.BKQM)
initq(SYSTEM.CHANNEL.INITQ)
```

Building command scripts is recommended for deploying brokers in a more controlled and reproducible manner.

Create the broker and configuration manager

Now that we have a prepared database and a queue manager, we can create the broker itself. One more command is required before we can create the components. When you obtain a license for WebSphere Business Integration Message Broker, you get several capacity units. The number of capacity units that is required to run WebSphere Business Integration Message Broker depends on your platform and number of processors. However, in all cases, you must inform the product about the number of purchased capacity units.

To assign one capacity unit, use this command:

```
mqsisetcapacity -c 1
```

To create the broker, use the **`mqsicreatebroker`** command:

```
mqsicreatebroker BROKER -i bkadmin -a sas313r -q BKQM -n BKDB -u wbiadmin -p
sas313r
mqsistart BROKER
```

To create the configuration manager, use this command:

```
mqsicreateconfigmgr -i cmadmin -a sas313r -q CMQM -l 0 -n CMDB -u wbiadmin -p
sas313r -m CMDB -e wbiadmin -r sas313r
mqsistart ConfigMgr
```

The option `-l 0` in the previous command implies that we do not use Windows Domain security, but only local security.

Both the broker and the configuration manager run as Windows services. To learn the status of these services, inspect the Windows Event Viewer. This is where WebSphere Business Integration Message Broker reports any warning or error messages during runtime operations.

If the Windows Event Viewer reported the following error messages during the configuration manager creation, ensure that the user ID `cmadmin` is a member of the Administrators group.

Example 3-7 Windows Evenbt Viewer ID error

```
( ConfigMgr ) Missing or blank configuration repository JDBC driver name.
The configuration repository JDBC driver name supplied to the Configuration
Manager is either blank or is missing. This is a mandatory property of the
Configuration Manager.
```

Using the Broker Toolkit

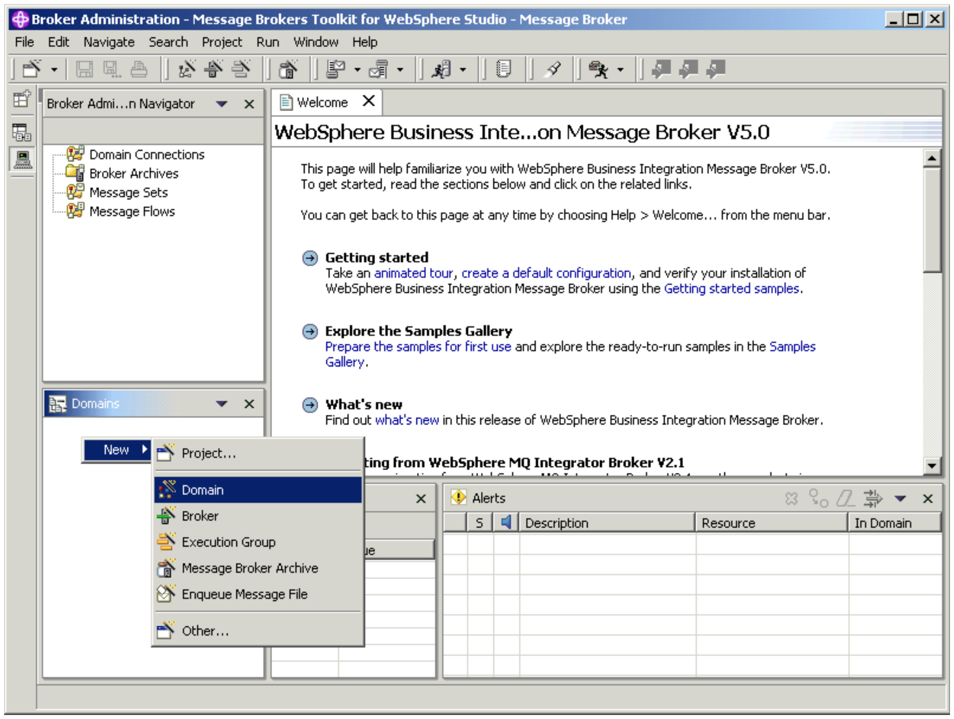
Similar to what we have done for WebSphere MQ Workflow and WebSphere InterChange Server, we have installed the administrative interface for WebSphere Business Integration Message Broker on the runtime server. In Chapter 4, “Implementing client components” on page 119, we describe the steps to make a development and management environment for WebSphere Business Integration Message Broker. Here, we describe how to connect the Broker Toolkit to the configuration manager and broker and how to complete the broker domain definition.

The Broker Toolkit provides two distinct perspectives:

- The Broker Development perspective is used to develop solution components for the broker.
- The Broker Administration perspective is used to manage the broker domain, assemble solutions, and deploy them to one or more brokers.

At this stage in the setup of WebSphere Business Integration Message Broker, we have a broker and a configuration manager that are not yet joined in a broker domain. To do this, we use the Broker Toolkit Broker Administration perspective:

1. Start the Broker Toolkit and switch to the Broker Administration perspective. Right-click in the Domains view and select **New** → **Domain**, as in Figure 3-60 on page 114.


Broker Administration - Message Brokers Toolkit for WebSphere Studio - Message Broker

WebSphere Business Inte...on Message Broker V5.0

This page will help familiarize you with WebSphere Business Integration Message Broker V5.0.
To get started, read the sections below and click on the related links.

You can get back to this page at any time by choosing Help > Welcome... from the menu bar.

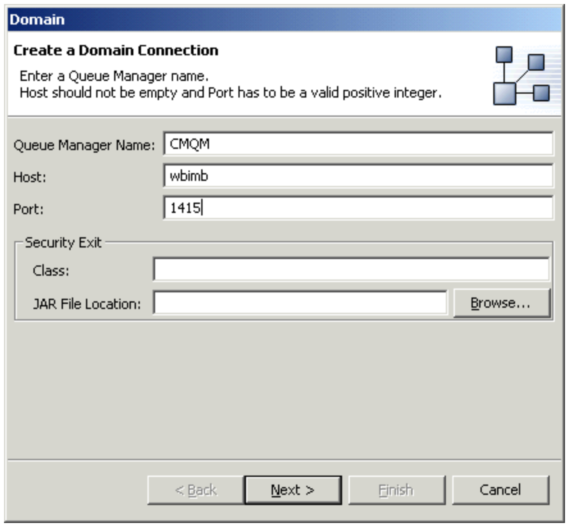
Getting started
Take an animated tour, create a default configuration, and verify your installation of WebSphere Business Integration Message Broker using the Getting started samples.

Explore the Samples Gallery
Prepare the samples for first use and explore the ready-to-run samples in the Samples Gallery.

What's new
Find out what's new in this release of WebSphere Business Integration Message Broker.

Figure 3-60 Create a new broker domain

2. In the Domain window (Figure 3-61), enter your environment's values for the configuration manager that we defined earlier: the name of the queue manager that supports the configuration manager, that machine's host name, and the port on which the TCP/IP listener is listening. Click **Next** to proceed.


Domain

Create a Domain Connection

Enter a Queue Manager name.
Host should not be empty and Port has to be a valid positive integer.

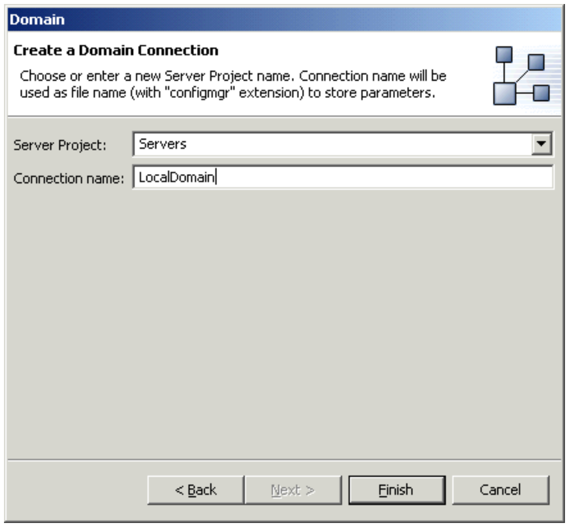
Queue Manager Name:	CMQM
Host:	wbimb
Port:	1415

Security Exit

Class:	
JAR File Location:	

Figure 3-61 Create a connection to a configuration manager

3. The Broker Toolkit holds connection information to one or more broker domains or configuration managers. The connection information for each domain is stored in a file that is located in a server project. The default name of that server project is **Servers**. The file holding the actual connection information can also be named. Figure 3-62 shows that we named this file `LocalDomain`. Click **Finish**.


Domain

Create a Domain Connection

Choose or enter a new Server Project name. Connection name will be used as file name (with "configmgr" extension) to store parameters.

Server Project: Servers

Connection name: LocalDomain

Figure 3-62 Create a connection to a configuration manager

The Broker Toolkit creates an MQ client connection to the queue manager that is used by the configuration manager. This implies that the logged-on user ID must be authorized to use that queue manager and that this ID must be a member of the groups that are used by the WebSphere Business Integration Message Broker product.

When the connection is successful, the Domain view shows an empty folder called Broker Topology (Figure 3-63). Right-click this folder and select **New** → **Broker**. This adds a new broker to the domain. However, it does not create the broker because the broker was created previously. You can think of this step as registering the broker in the domain.

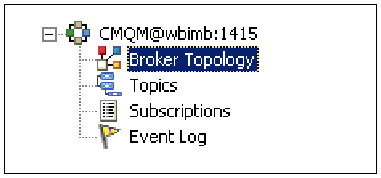
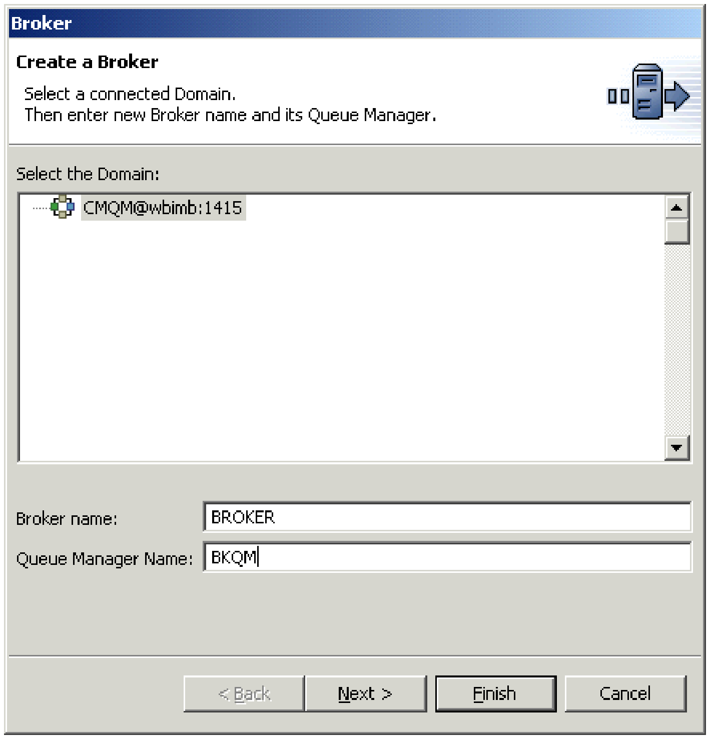


Figure 3-63 Empty broker domain

4. To add the broker to the domain, give the configuration manager the broker's name and the name of the queue manager that it uses (Figure 3-64).


Broker

Create a Broker

Select a connected Domain.
Then enter new Broker name and its Queue Manager.

Select the Domain:

CMQM@wbimb:1415

Broker name: BROKER

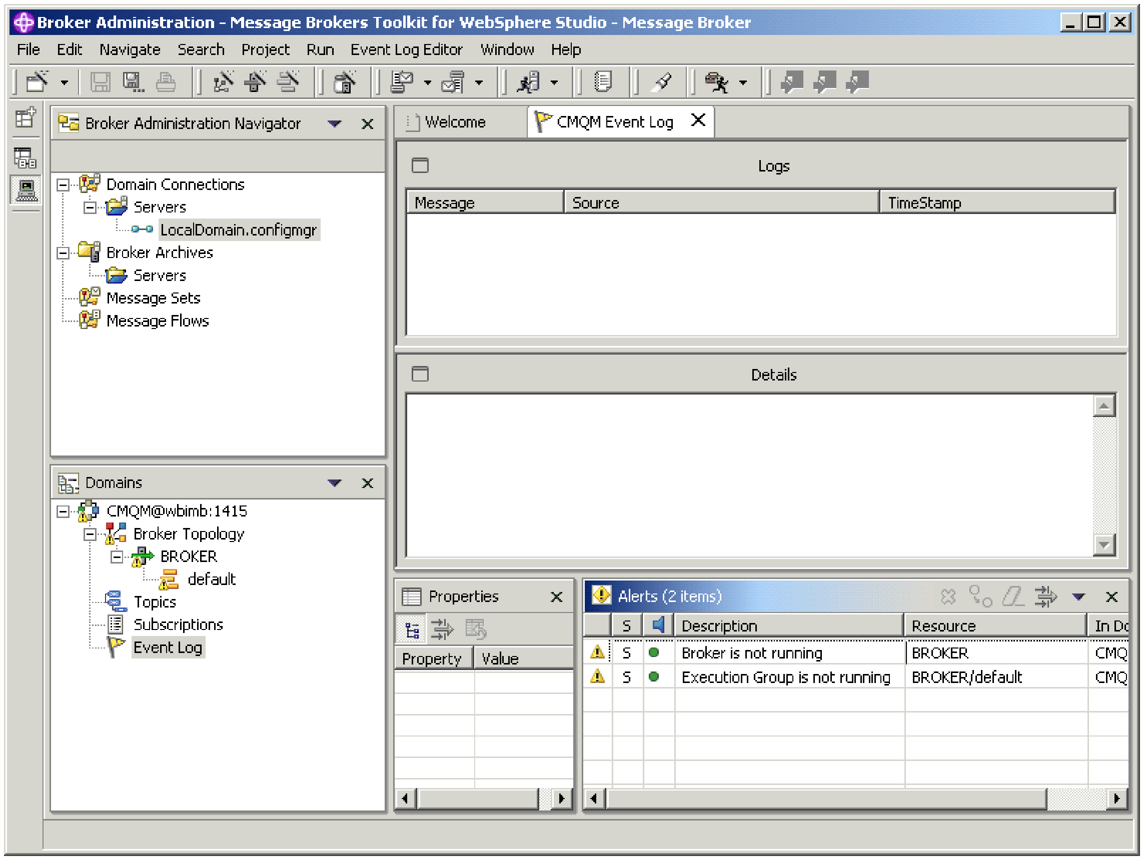
Queue Manager Name: BKQM

Figure 3-64 Create a new broker

When the configuration manager and the broker use different queue managers, as is the case in our environment, it is assumed that MQ messages can be exchanged between these two queue managers using standard MQ facilities such as channels and transmission queues. This is why we defined channels and transmission queues before we actually created the configuration manager and broker. (See "WebSphere MQ configuration" on page 110.)

After you click **Finish**, the configuration manager deploys the changes to its domain and contacts the broker. When successful, the Domains view lists the broker and a single execution group called Default. If the deployment operation fails, you can find detailed information about the error in the Windows Event Viewer or in the Event Log, which is part of the Broker Toolkit (Figure 3-65). The Event Log is available in the Domains view.

Figure 3-65 on page 118 also shows that the broker and the execution group are not really running. (See the Alerts view.) The alert about the broker will eventually clear. Figure 3-65 was taken before the deployment was completely finished. The alert about the execution group disappears when a broker solution has been deployed to this execution group. As long as there are no deployed components, the execution group will show as `not running` in the Broker Toolkit.



	S		Description	Resource	In Dc
⚠	S	●	Broker is not running	BROKER	CMQ
⚠	S	●	Execution Group is not running	BROKER/default	CMQ

Figure 3-65 Broker added to domain

3.5 Summary

In this chapter we described a multi-machine setup of a WebSphere Business Integration infrastructure containing runtime servers for workflow, process integration, and message-based integration. The next chapter describes how we can create a development and management environment for this infrastructure.

4

Implementing client components

This chapter describes the implementation of various clients for interaction with the WebSphere Business Integration Server components that were created in Chapter 3, “Implementing the runtime components” on page 45. We describe the implementation of:

- A WebSphere MQ Workflow Web client infrastructure.
- A development client for developing artifacts for WebSphere MQ Workflow, InterChange Server, and WebSphere Business Integration Message Broker.
- A management client to deploy solutions to the infrastructure and to administer the runtime environment.

4.1 Implementing WebSphere MQ Workflow Web Client

In this section we cover the installation, configuration, and validation of WebSphere MQ Workflow Web Client. We also cover the application server preparation and the WebSphere MQ configuration.

The WebSphere MQ Workflow Web Client is a *servlet,* (a Java program running on a Web server, that enables a browser-based interface for the WebSphere MQ Workflow Runtime. When you deploy the Web Client in your application server, it allows a user with a Web browser with JavaScript™ support to access MQ Workflow process template lists, process instance lists, worklists, user settings, list settings, and object properties (input and output containers), including worklist control, and process monitoring.

Also, Web client enablement makes for easier WebSphere MQ Workflow client setups (only the server changes, no client updates necessary) and allows for a client concentrator machine to share WebSphere MQ workload with the server. Furthermore, the client concentrator setup is necessary for multiple WebSphere MQ Workflow Runtime servers to share a growing workload.

You can also link the WebSphere MQ Workflow Web client feature to the workflow server through WebSphere MQ Client connections. This opens a WebSphere MQ Client connection from the WebSphere Application Server to the workflow Runtime server for every client that needs workflow services.

The advantage of using a concentrator server is that it takes some of the WebSphere MQ workload from the workflow Runtime server. Instead of a client connection channel for every client that connects to the workflow system, there is now only one pair of sending/receiving channels between the workflow concentrator server and the workflow Runtime server.

Because WebSphere MQ Workflow inherently makes use of WebSphere MQ cluster technology, a client concentrator setup also does load-balancing and fail-over when needed.

Figure 4-1 on page 121 is a high-level graphic representation of all components that are involved in this setup.

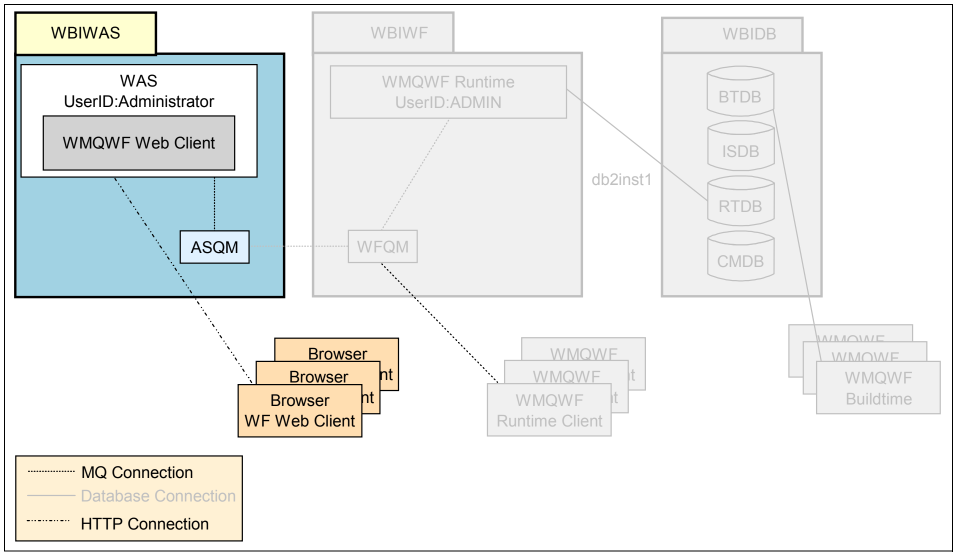


Figure 4-1 WebSphere MQ Workflow Web client components topology

4.1.1 Setting up the application server on Windows

Important: This redbook does not cover the installation and configuration of the application server and WebSphere MQ. For information about installation and configuration, refer to the redbook ??????

To set up a Windows based server, follow these steps:

1. Log on with an administrator ID on the server where WebSphere Application Server is installed.
2. Install WebSphere MQ server as described in the product documentation.

   It is important to use WebSphere MQ server on this platform because if you only use WebSphere MQ client, then every client that connects to the application server will also result in a WebSphere MQ client connection to the WebSphere MQ Workflow Runtime server. This, in turn, will result in severe scalability problems when many users try to work with the system.

   Note: A WebSphere MQ license for the application server is included in the WebSphere MQ Workflow license as long as that copy of WebSphere MQ is only used for WebSphere MQ Workflow.

3. Before we continue, we verify that the application server is running.

- If the application server is *not running*, select **Start** → **Programs** → **IBM WebSphere** → **Application Server v.5.1** → **Start the Server**. This starts the application server that hosts the administrative console and sample applications. You can use the same application server to host the Web client, or you can create a separate application server on the same host.
- To verify that the server is up and running, go to the command prompt and change directories to `<WAS_Home>\bin`. Enter:

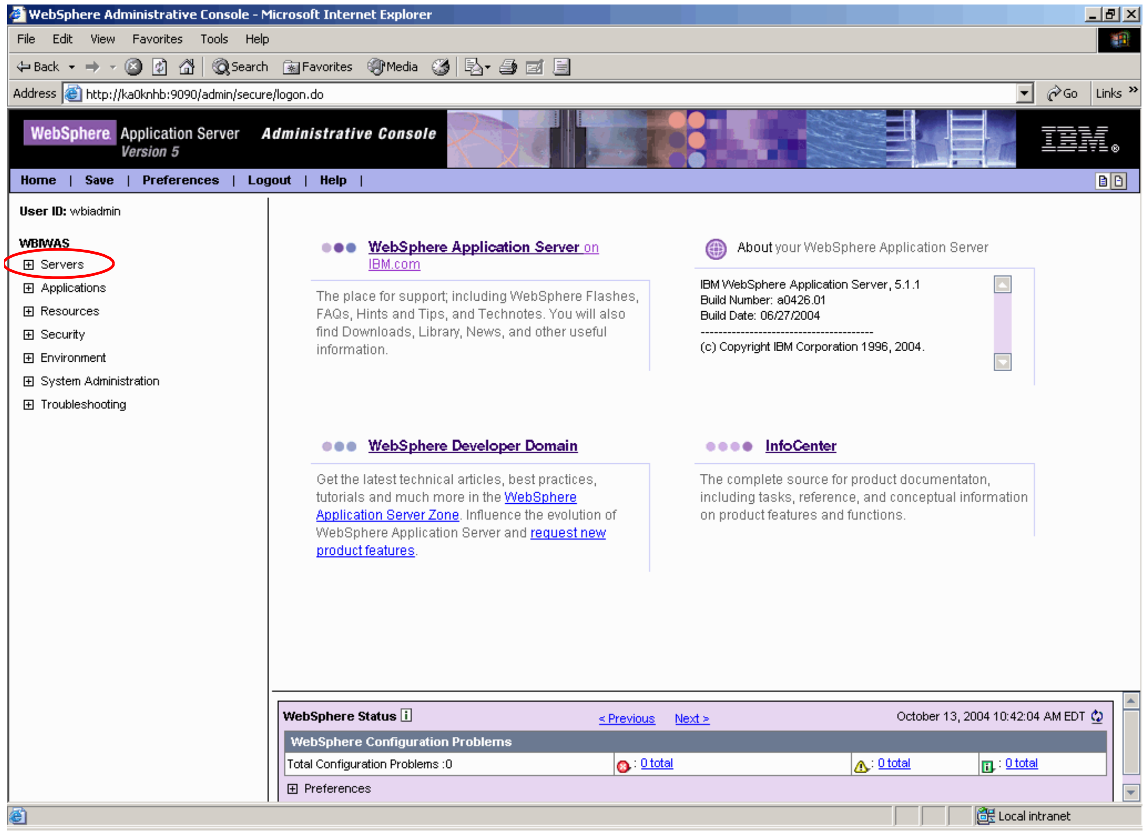
  ```
  serverStatus -all
  ```

- To use the administrative console to check the status, follow these steps:

  Select **Start** → **Programs** → **IBM WebSphere** → **Application Server v5.1** → **Administrative Console**.

The WebSphere Server - Administrative Console opens. Provide the administrative user ID, `wbiadmin` in our case, and click **OK**.

After login, the Administrative Console appears as is seen in Figure 4-2.


WebSphere Administrative Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://ka0knhb:9090/admin/secure/logon.do
Go Links
WebSphere Application Server Version 5 Administrative Console
IBM
Home | Save | Preferences | Logout | Help |
User ID: wbiadmin
WBIWAS
Servers
Applications
Resources
Security
Environment
System Administration
Troubleshooting
WebSphere Application Server on IBM.com

The place for support; including WebSphere Flashes, FAQs, Hints and Tips, and Technotes. You will also find Downloads, Library, News, and other useful information.

About your WebSphere Application Server

IBM WebSphere Application Server, 5.1.1
Build Number: a0426.01
Build Date: 06/27/2004

(c) Copyright IBM Corporation 1996, 2004.

WebSphere Developer Domain

Get the latest technical articles, best practices, tutorials and much more in the WebSphere Application Server Zone. Influence the evolution of WebSphere Application Server and request new product features.

InfoCenter

The complete source for product documentaton, including tasks, reference, and conceptual information on product features and functions.

WebSphere Status < Previous Next > October 13, 2004 10:42:04 AM EDT

WebSphere Configuration Problems

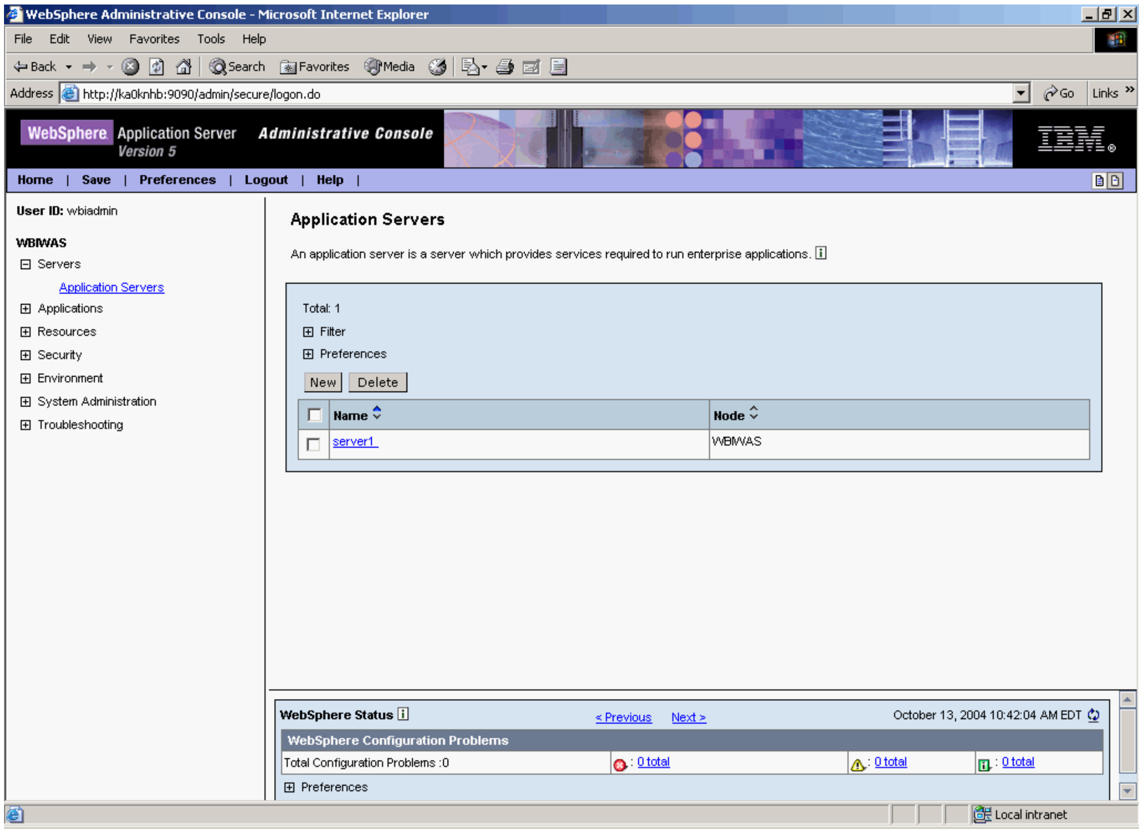
Total Configuration Problems :0 | : 0 total | : 0 total | : 0 total

Preferences

Figure 4-2 Administrative Console

4. Expand **Servers** in the upper left of the window (Figure 4-2) to see all servers.

5. Under Servers, click the **Application Servers** link. At this point, you can see the name of your application server in the right-hand pane. In our case, this is **server1**. See Figure 4-3.


WebSphere Administrative Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://ka0knhb:9090/admin/secure/logon.do
WebSphere Application Server Version 5 Administrative Console
Home | Save | Preferences | Logout | Help |
User ID: wbiadmin
WBIWAS
Servers
Application Servers
Applications
Resources
Security
Environment
System Administration
Troubleshooting

Application Servers

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

Total: 1

Filter

Preferences

New Delete

	Name	Node
☐	server1	WBIWAS

WebSphere Status < Previous Next > October 13, 2004 10:42:04 AM EDT

WebSphere Configuration Problems

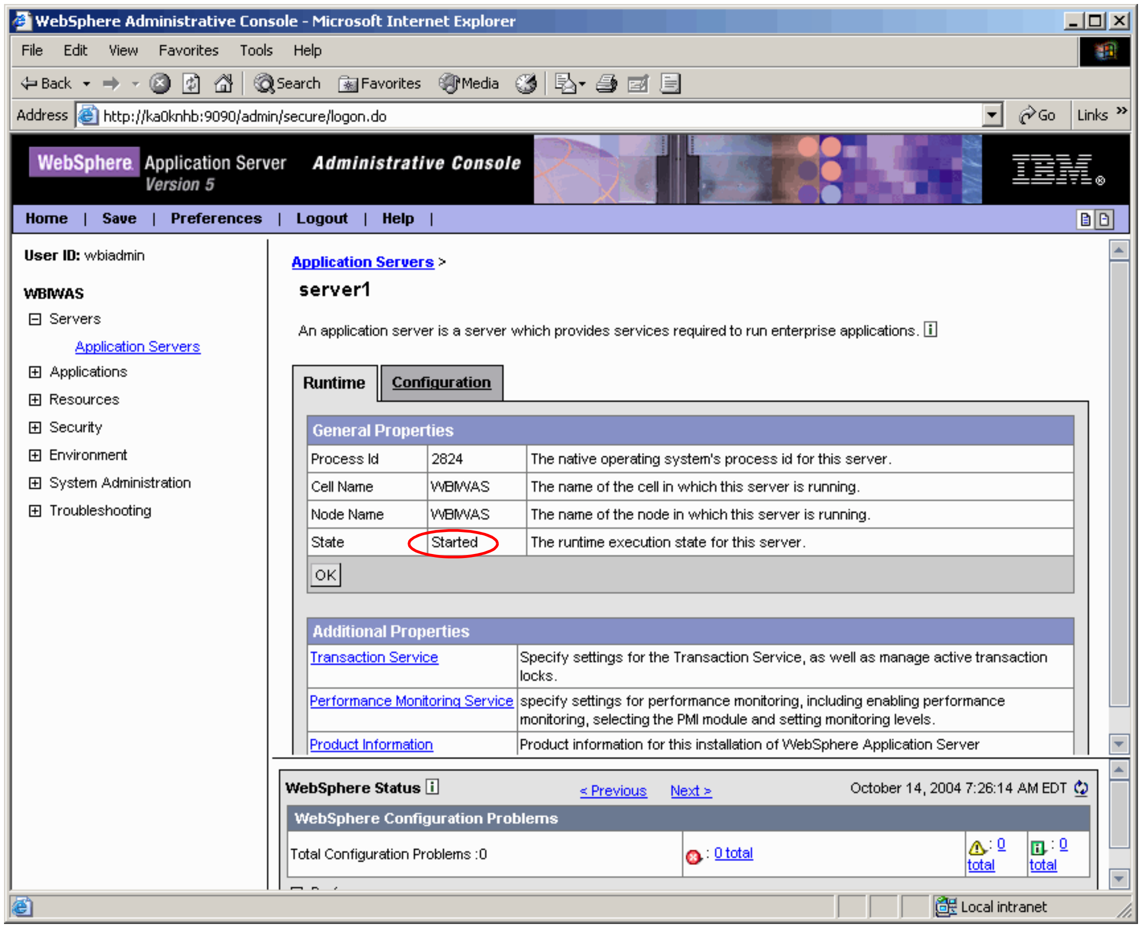
Total Configuration Problems :0	: 0 total	: 0 total	: 0 total

Preferences

Local intranet

Figure 4-3 Administrative Console - Application Servers list

6. Click the **server1** link and select the **Runtime** tab.



WebSphere Administrative Console - Microsoft Internet Explorer

Address http://ka0knhb:9090/admin/secure/logon.do

User ID: wbiadmin

WBIWAS

Application Servers > server1

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

Runtime | Configuration

General Properties

Process Id	2824	The native operating system's process id for this server.
Cell Name	WBIWAS	The name of the cell in which this server is running.
Node Name	WBIWAS	The name of the node in which this server is running.
State	Started	The runtime execution state for this server.

Additional Properties

Transaction Service	Specify settings for the Transaction Service, as well as manage active transaction locks.
Performance Monitoring Service	specify settings for performance monitoring, including enabling performance monitoring, selecting the PMI module and setting monitoring levels.
Product Information	Product information for this installation of WebSphere Application Server

WebSphere Status   October 14, 2004 7:26:14 AM EDT

WebSphere Configuration Problems

Total Configuration Problems :0

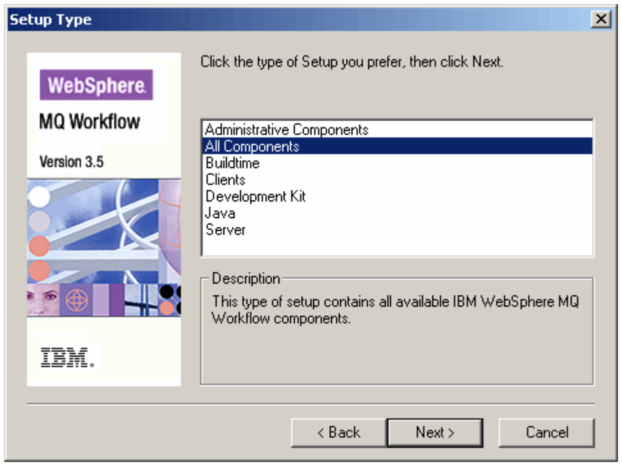
Figure 4-4   Application server state

As you can see in Figure 4-4, the application server is showing a `Started` state. We are now ready to begin the installation and configuration of the WebSphere MQ Workflow Web Client.

4.1.2 WebSphere MQ Workflow Web Client installation

This section details the installation of the WebSphere MQ Workflow Web Client.

1. Launch the WebSphere MQ Workflow setup, accept the license agreement, and select your installation language.
2. Click **Next** and select **All Components** (Figure 4-5).



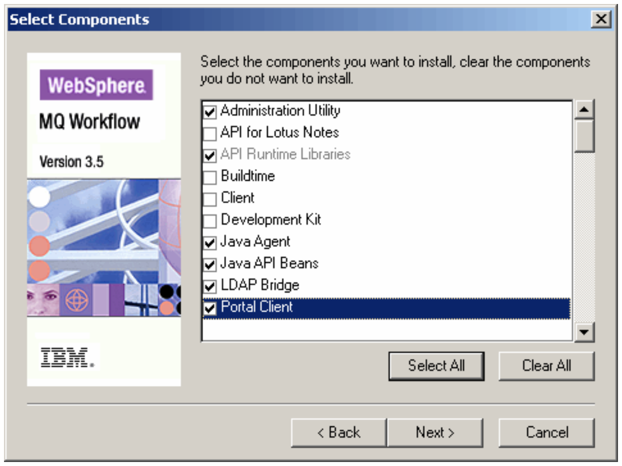
Click the type of Setup you prefer, then click Next.

Administrative Components
All Components
Buildtime
Clients
Development Kit
Java
Server

Description
This type of setup contains all available IBM WebSphere MQ Workflow components.

Figure 4-5 Setup type

3. On the Select Components screen (Figure 4-6), select **Administration Utility**, **API Runtime Libraries**, **Java Agent**, **Java API Beans**, **LDAP Bridge**, and **Portal Client**.



Select the components you want to install, clear the components you do not want to install.

- [✓] Administration Utility
- [ ] API for Lotus Notes
- [✓] API Runtime Libraries
- [ ] Buildtime
- [ ] Client
- [ ] Development Kit
- [✓] Java Agent
- [✓] Java API Beans
- [✓] LDAP Bridge
- [✓] Portal Client

Figure 4-6 Select Components

After a successful completion of the installation, the WebSphere MQ Workflow Configuration window opens (Figure 4-7). If it does not, select **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **IBM WebSphere MQ Workflow Configuration Utility**.

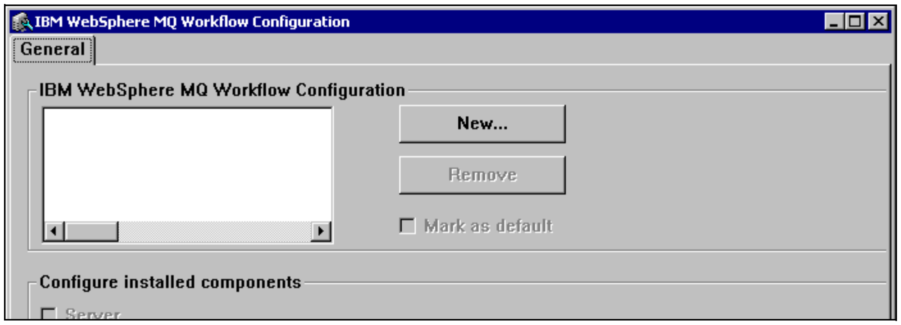
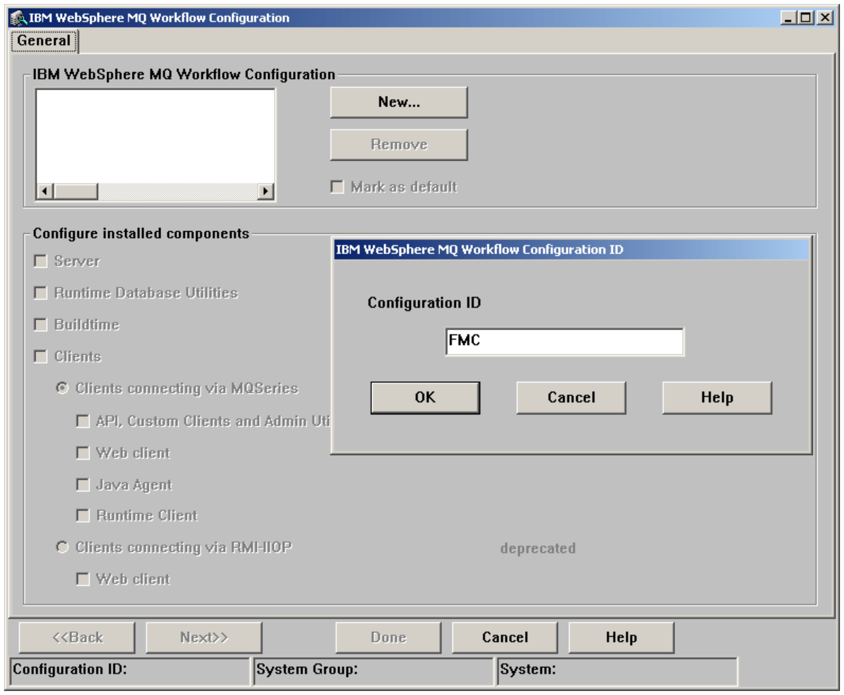

IBM WebSphere MQ Workflow Configuration
General
IBM WebSphere MQ Workflow Configuration
New...
Remove
Mark as default
Configure installed components
Server

Figure 4-7 WebSphere MQ Workflow Configuration

4.1.3 WebSphere MQ Workflow Web Client configuration

In this section, we cover the configuration needed to establish successfully a WebSphere MQ Workflow Web Client environment. As part of the configuration, we add an additional WebSphere MQ queue manager to the already established WebSphere MQ clustered queue manager environment on the Workflow server (WBIWF). This newly created queue manager will reside on the same server where the application server is running (WBIWAS), and will communicate with the repository queue manager on WBIWF. Because of the simplified management inherent in the WebSphere MQ queue manager clustering technology, we have to create only one pair of sender/receiver channels between the queue manager on the application server, and the WebSphere Workflow Runtime server.

1. On the **General** tab of the configuration utility, click **New** to create the configuration profile for the WebSphere MQ Workflow Web Client setup. Provide a name for the new configuration (Figure 4-8), and click **OK** to continue.


IBM WebSphere MQ Workflow Configuration
General
IBM WebSphere MQ Workflow Configuration
New...
Remove
Mark as default
Configure installed components
Server
Runtime Database Utilities
Buildtime
Clients
Clients connecting via MQSeries
API, Custom Clients and Admin Uti
Web client
Java Agent
Runtime Client
Clients connecting via RMI-IIOP
deprecated
Web client
IBM WebSphere MQ Workflow Configuration ID
OK
Cancel
Help
<<Back
Next>>
Done
Cancel
Help
System Group:
System:

Configuration ID

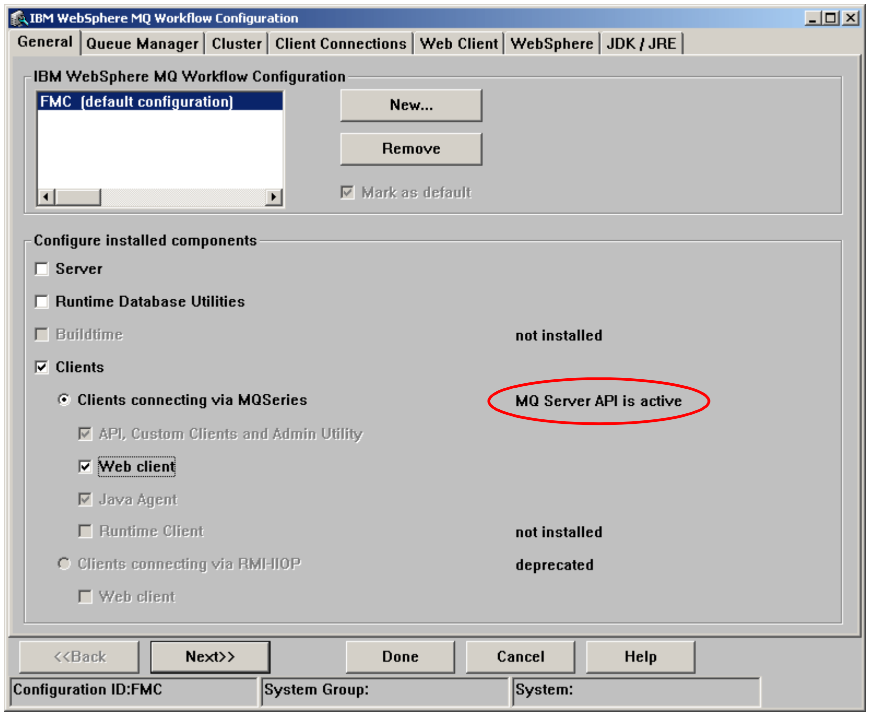
FMC

Configuration ID:

Figure 4-8 General and Configuration ID

2. Select the **Web client** check box and note that all of the necessary tabs are added to the top of the window (Figure 4-9).

Note: If the window does *not* show that the MQ Server API is the active WebSphere MQ API, then make sure that WebSphere MQ server is installed on the application server and that the path environment variable points to the WebSphere MQ Workflow server directory and not the client directory.


IBM WebSphere MQ Workflow Configuration
General | Queue Manager | Cluster | Client Connections | Web Client | WebSphere | JDK / JRE
New...
Remove
Done
Cancel
Help
<<Back
Next>>

IBM WebSphere MQ Workflow Configuration
FMC (default configuration)
[✓] Mark as default

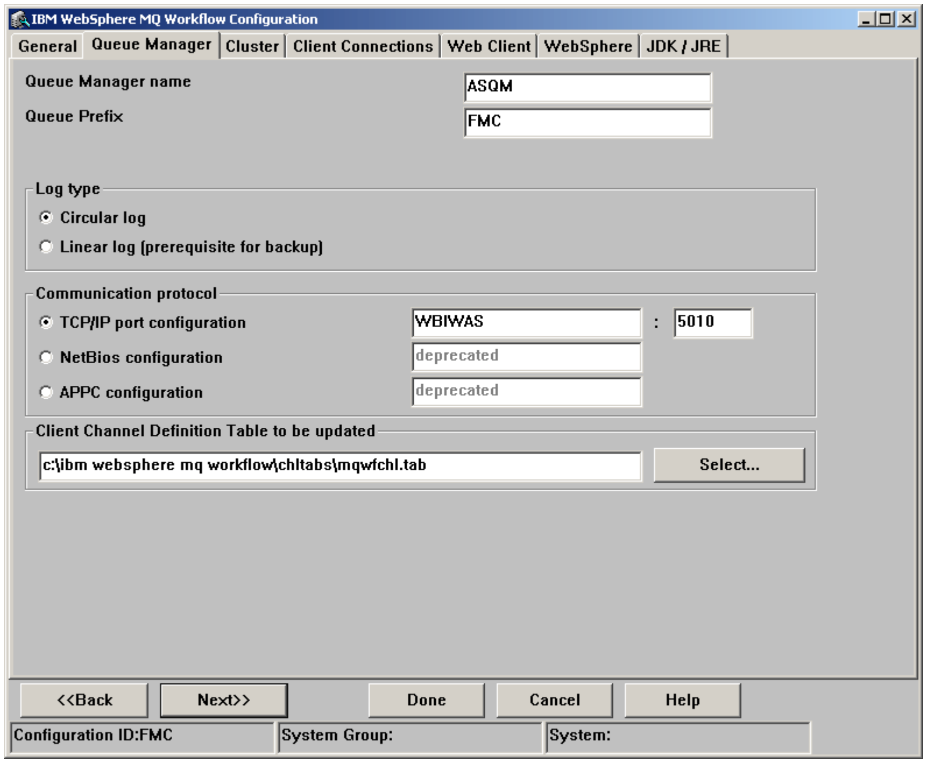
Configure installed components
[ ] Server
[ ] Runtime Database Utilities
[ ] Buildtime — not installed
[✓] Clients
(●) Clients connecting via MQSeries — MQ Server API is active
[✓] API, Custom Clients and Admin Utility
[✓] Web client
[✓] Java Agent
[ ] Runtime Client — not installed
( ) Clients connecting via RMI-IIOP — deprecated
[ ] Web client

Configuration ID:FMC | System Group: | System:

Figure 4-9 WebSphere MQ Workflow Web Client configuration: General tab

3. Select the **Queue Manager** tab and provide the name of the queue manager for the application server machine (see Table 2-2 on page 40). For TCP/IP port configuration, enter the host name of the WebSphere Application Server machine and the queue manager port, from the same table. (Figure 4-10 shows this screen.)

   Click **Next** to continue.



Queue Manager name: ASQM
Queue Prefix: FMC

Log type
- [✓ Circular log]
- Linear log (prerequisite for backup)

Communication protocol
- [✓ TCP/IP port configuration] WBIWAS : 5010
- NetBios configuration deprecated
- APPC configuration deprecated

Client Channel Definition Table to be updated: c:\ibm websphere mq workflow\chltabs\mqwfchl.tab

Configuration ID:FMC
System Group:
System:

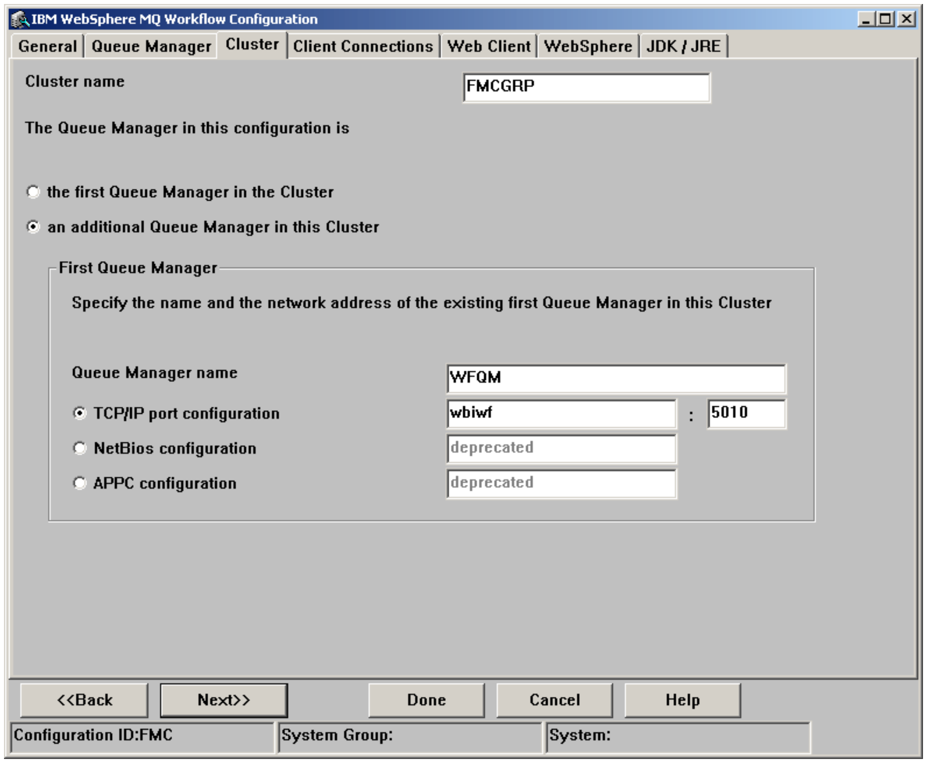
Figure 4-10 WebSphere MQ Workflow Web Client configuration: Queue Manager tab

4. Now we are on the Cluster tab (Figure 4-11). Because we already have another active queue manager in workflow queue manager cluster, we select the appropriate radio button to specify that the queue manager in this configuration is an additional Queue Manager in this Cluster. For the Cluster name, use the MQ cluster name (FMCGRP in our case) as planned in Table 2-1 on page 39.

   In the First Queue Manager portion of the screen, provide the Queue Manager name as planned in Table 2-1 on page 39. (See Queue Manager Name.) Use the WebSphere MQ Workflow Runtime machine's TCP/IP address or host name and Queue Manager TCP/IP Port information (see

TCP/IP address or host name and Queue Manager TCP/IP Port respectively in Table 2-1 on page 39) for the TCP/IP port configuration input fields.

Click **Next** to continue.



Cluster name: FMCGRP

The Queue Manager in this configuration is

( ) the first Queue Manager in the Cluster
[✓ an additional Queue Manager in this Cluster]

First Queue Manager

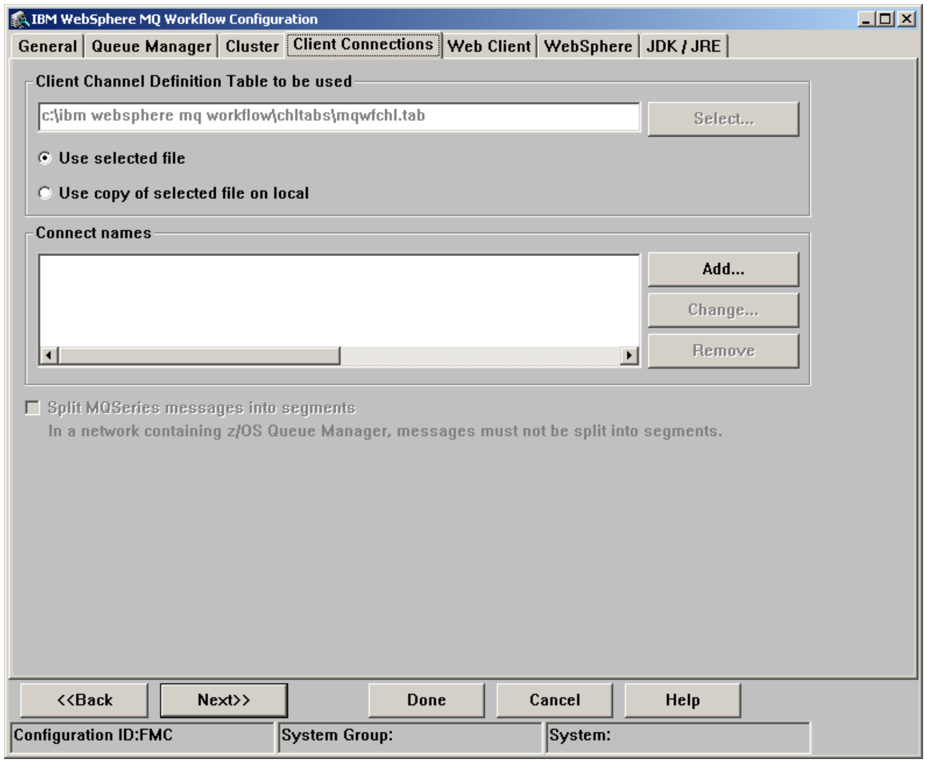
Specify the name and the network address of the existing first Queue Manager in this Cluster

Queue Manager name: WFQM
[✓ TCP/IP port configuration]: wbiwf : 5010
( ) NetBios configuration: deprecated
( ) APPC configuration: deprecated

Configuration ID:FMC | System Group: | System:

Figure 4-11 WebSphere MQ Workflow Web Client configuration: Cluster tab

5. On the **Client Connections** tab (Figure 4-12), in the Connect names portion of the screen, press the **Add** button to add information about the existing workflow server.



Client Channel Definition Table to be used

c:\ibm websphere mq workflow\chltabs\mqwfchl.tab

(●) Use selected file

( ) Use copy of selected file on local

Connect names

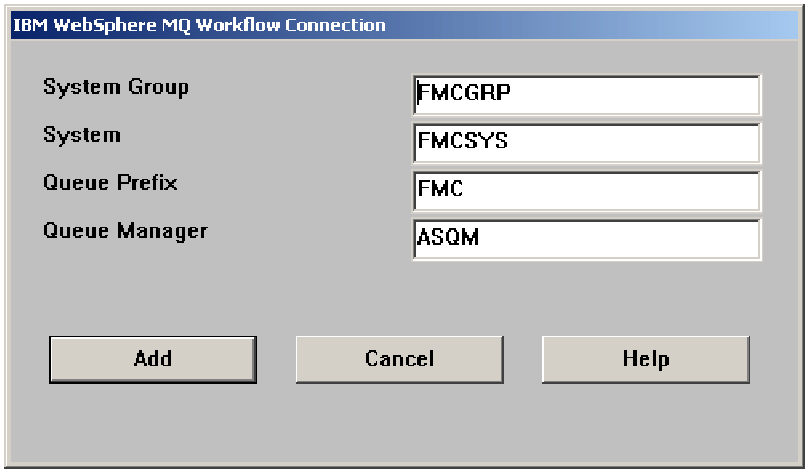
[ ] Split MQSeries messages into segments

In a network containing z/OS Queue Manager, messages must not be split into segments.

Configuration ID:FMC | System Group: | System:

Figure 4-12 WebSphere MQ Workflow Web Client configuration: Client Connections tab

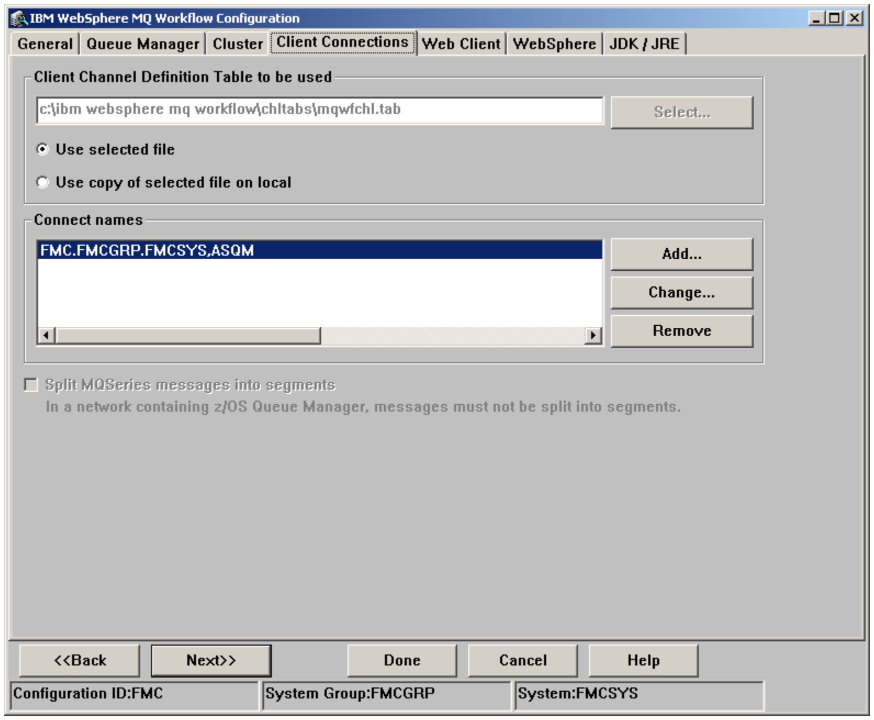
6. This opens the window shown in Figure 4-13 on page 133. Verify the system information and make sure that the Queue Manager field contains the Application Server Queue Manager Name as planned in Table 2-2 on page 40. Click **Add** to return to the Configuration window.


IBM WebSphere MQ Workflow Connection

System Group	FMCGRP
System	FMCSYS
Queue Prefix	FMC
Queue Manager	ASQM

Figure 4-13 WebSphere MQ Workflow Connection

As you can see in Figure 4-14 the Connect name is now specified.


IBM WebSphere MQ Workflow Configuration

Client Channel Definition Table to be used

c:\ibm websphere mq workflow\chltabs\mqwfchl.tab

(●) Use selected file

( ) Use copy of selected file on local

Connect names

FMC.FMCGRP.FMCSYS.ASQM

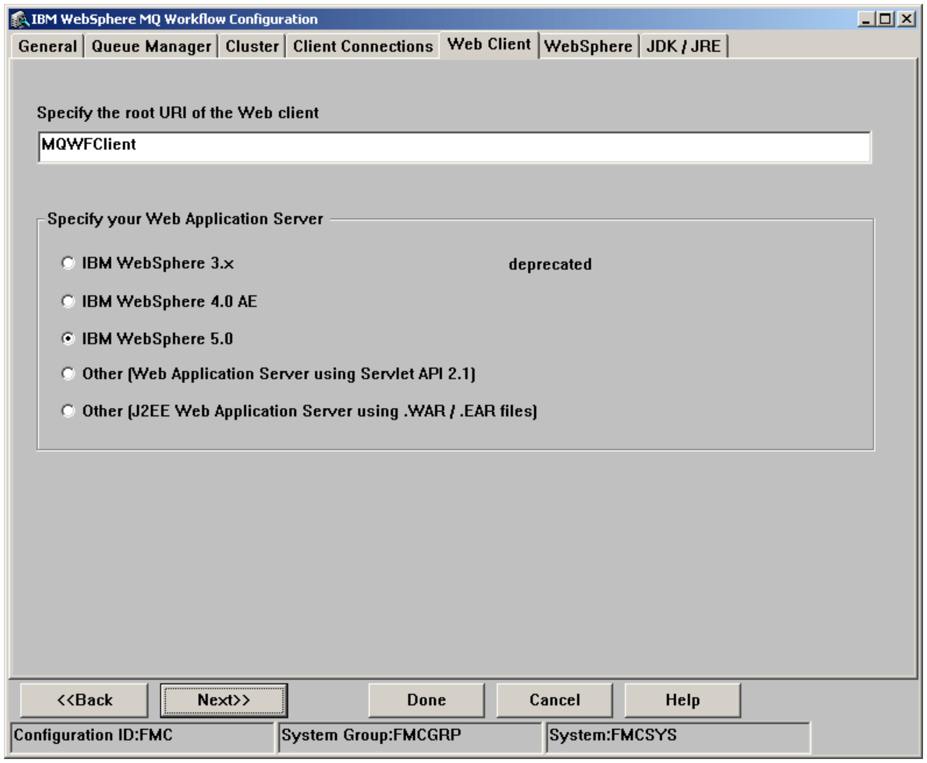
☐ Split MQSeries messages into segments

In a network containing z/OS Queue Manager, messages must not be split into segments.

Configuration ID:FMC | System Group:FMCGRP | System:FMCSYS

Figure 4-14 Client Connections tab with Connect names

7. Click the **Web Client** tab. Enter the Web Client URL from Table 2-2 on page 40 and ensure that the correct version of the application server is selected. Click **Next** to continue.



Specify the root URI of the Web client: MQWFClient

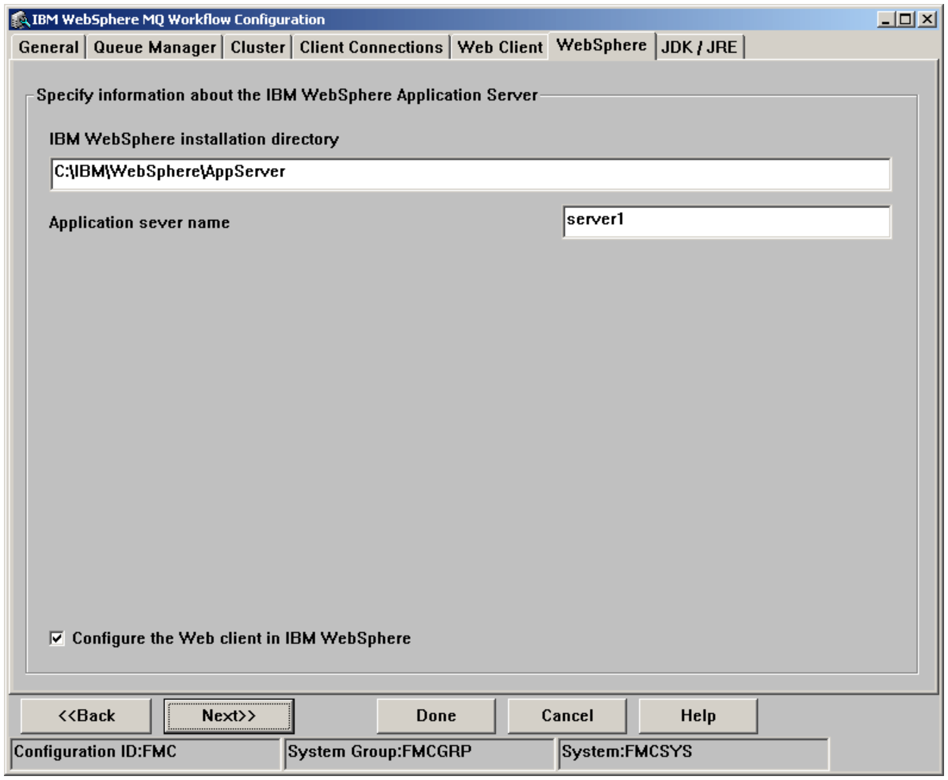
Specify your Web Application Server

- IBM WebSphere 3.x deprecated
- IBM WebSphere 4.0 AE
- [✓ IBM WebSphere 5.0]
- Other (Web Application Server using Servlet API 2.1)
- Other (J2EE Web Application Server using .WAR / .EAR files)

Configuration ID:FMC | System Group:FMCGRP | System:FMCSYS

Figure 4-15 WebSphere MQ Workflow Web Client configuration: Web Client tab

8. Click the **WebSphere** tab, review the values, and click **Next** to continue.


IBM WebSphere MQ Workflow Configuration

General | Queue Manager | Cluster | Client Connections | Web Client | WebSphere | JDK / JRE

Specify information about the IBM WebSphere Application Server

IBM WebSphere installation directory: C:\IBM\WebSphere\AppServer

Application sever name: server1

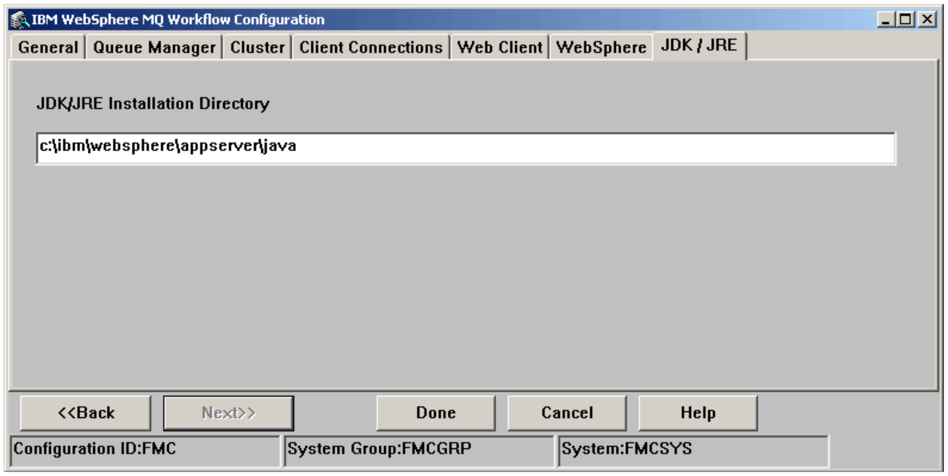
☑ Configure the Web client in IBM WebSphere

<<Back | Next>> | Done | Cancel | Help

Configuration ID:FMC | System Group:FMCGRP | System:FMCSYS

Figure 4-16 WebSphere MQ Workflow Web Client configuration: WebSphere tab

9. On the **JDK/JRE** tab (Figure 4-17), verify that the JDK/JRE Installation Directory is correct and click **Done.**



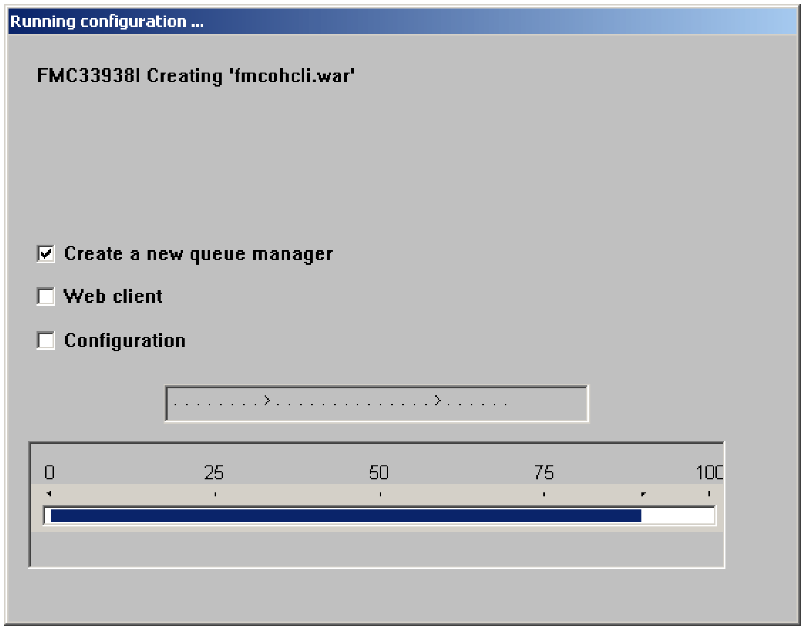
JDK/JRE Installation Directory

c:\ibm\websphere\appserver\java

Configuration ID:FMC
System Group:FMCGRP
System:FMCSYS

Figure 4-17 WebSphere MQ Workflow Web Client configuration: JDK/JRE tab

10. This will start the creation and configuration of all the defined components (Figure 4-18). During this process the configurator creates the queue manager and deploys the Web Client in the WebSphere Application Server.



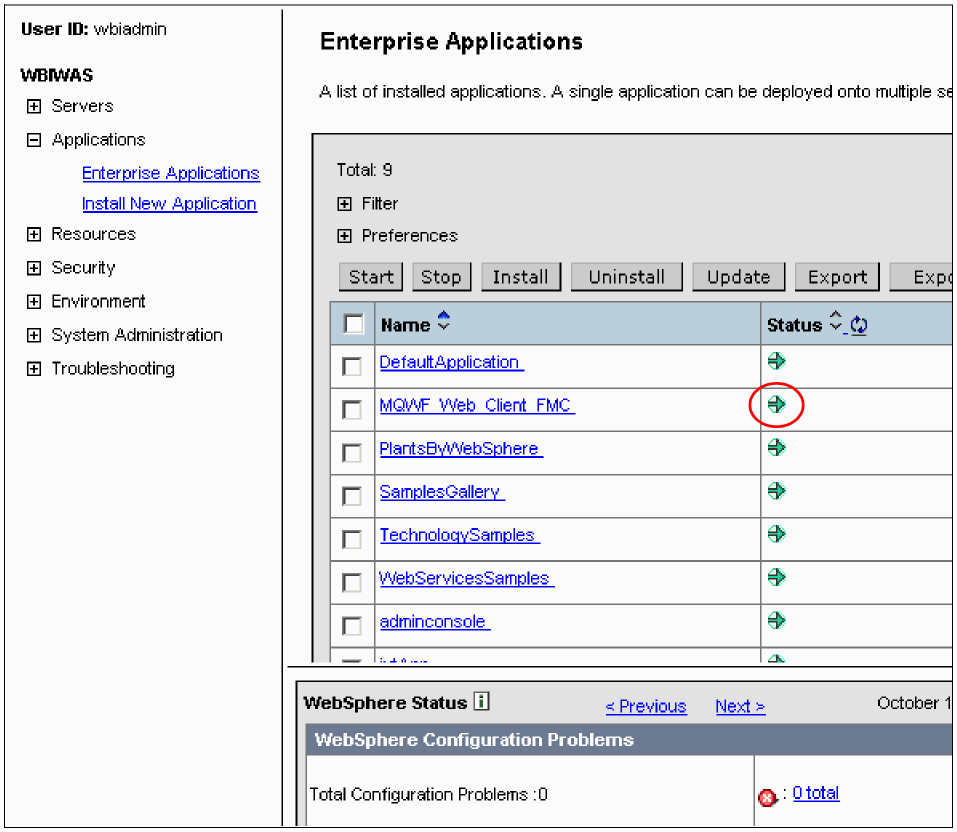
FMC33938I Creating 'fmcohcli.war'

- [x] Create a new queue manager
- [ ] Web client
- [ ] Configuration

Figure 4-18 Executing the configuration steps for the Web Client

11. After the configuration is complete, you will be notified that you should start the newly created enterprise application in the WebSphere Application Server and will receive a message about restarting the HTTP Server.

12.After restarting the HTTP Server, reopen the WebSphere Administrative Console. Expand **Applications** in the tree menu, and in the Enterprise Applications field, locate MQWF Web Client FMC. To the right of the application, you will see a status arrow (Figure 4-19). A red status arrow indicates that the application is stopped. A green status arrow indicates that the application is running.


User ID: wbiadmin

WBIWAS

- ⊞ Servers
- ⊟ Applications
  - Enterprise Applications
  - Install New Application
- ⊞ Resources
- ⊞ Security
- ⊞ Environment
- ⊞ System Administration
- ⊞ Troubleshooting

Enterprise Applications

A list of installed applications. A single application can be deployed onto multiple se

Total: 9

⊞ Filter

⊞ Preferences

☐	Name	Status
☐	DefaultApplication	➔
☐	MQWF_Web_Client_FMC	➔
☐	PlantsByWebSphere	➔
☐	SamplesGallery	➔
☐	TechnologySamples	➔
☐	WebServicesSamples	➔
☐	adminconsole	➔

WebSphere Status < Previous Next > October 1

WebSphere Configuration Problems

Total Configuration Problems :0 : 0 total

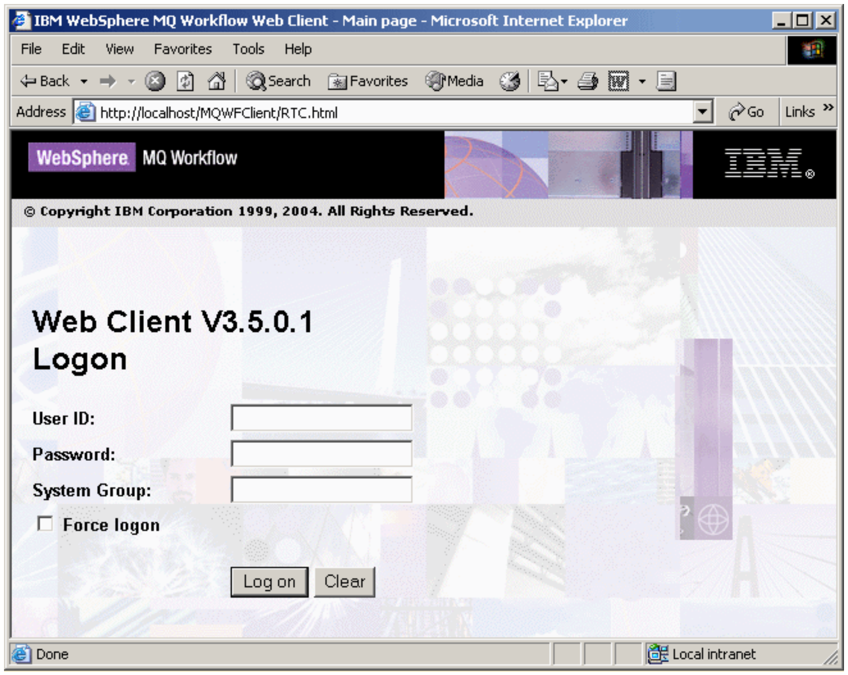
Figure 4-19 Status of Web Client enterprise application

If the arrow is red (stopped), check the box next to the **MQWF Web Client FMC** application and restart it.

4.1.4 WebSphere MQ Workflow Web Client validation

Before you use the Web client, several other tests can be performed to validate the configuration. Start the WebSphere MQ Explorer application and expand the tree structure **WebSphere MQ** → **Queue Managers** → **ASQM** → **Advanced** → **Channels**. You should see a cluster sender channel called `TO.WFQM.TCP` and a cluster receiver channel `TO.ASQM.TCP`. Both channels should have a channel status of `Running`. If this is not the case, try to start them manually. If they fail to start, use WebSphere MQ tools to determine the problem. Often, this is the result of a misspelling of a choice in the configuration tool.

If the channels are running, open a browser and go to the Web Client using the URL `http://`*`wbiwas`*`/MQWFClient/RTC.html`, where *`wbiwas`* is the host name of the server that is running WebSphere Application Server (Figure 4-20). Use the User ID `ADMIN` and password `password` to log on.


IBM WebSphere MQ Workflow Web Client - Main page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address
Go
Links
WebSphere MQ Workflow
IBM
Done
Local intranet

http://localhost/MQWFClient/RTC.html

© Copyright IBM Corporation 1999, 2004. All Rights Reserved.

Web Client V3.5.0.1
Logon

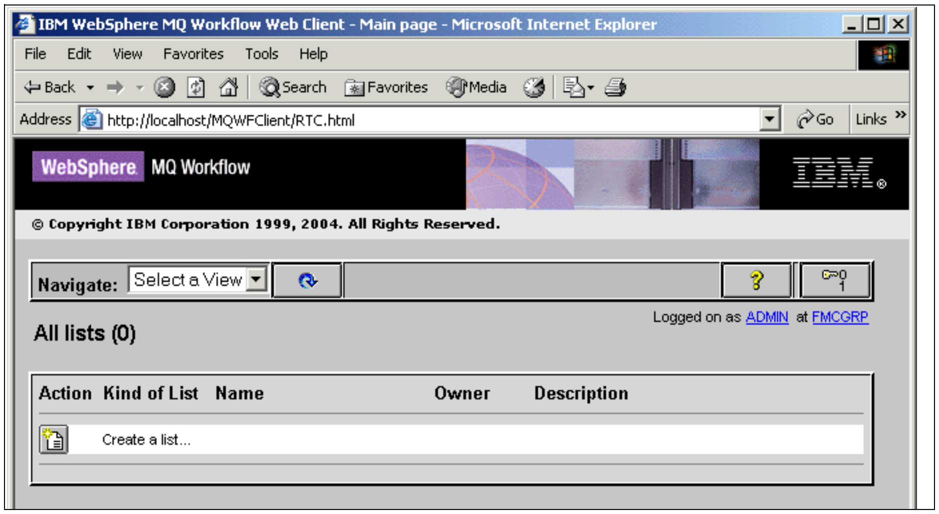
User ID:
Password:
System Group:
☐ Force logon

Log on | Clear

Figure 4-20 WebSphere MQ Workflow Web Client logon

Note: Figure 4-20 indicates that we have installed FixPak 1 for the Web Client (i.e. Web Client V3.5.0.1). This FixPak is installed on the workflow server, and also on the application server. You should install the desired FixPak before creating the Web Client configuration.

When the logon is successful, this opens the uncustomized Web Client interface shown in Figure 4-21.


IBM WebSphere MQ Workflow Web Client - Main page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/MQWFClient/RTC.html
Go Links
WebSphere MQ Workflow
IBM

© Copyright IBM Corporation 1999, 2004. All Rights Reserved.

Navigate: Select a View

Logged on as ADMIN at FMCGRP

All lists (0)

Action	Kind of List	Name	Owner	Description
	Create a list...			

Figure 4-21 Successful logon via Web client

4.2 Implementing a development client

This section describes the implementation of a development client for each of the runtime servers.

4.2.1 WebSphere MQ Workflow development environment

In Chapter 3, “Implementing the runtime components” on page 45, we install all WebSphere MQ Workflow components. However, there are instances where you might only want certain Workflow components. Hardware cost, team structure, and user roles and responsibilities are just a few examples that could drive certain installation decisions. Considering that a typical team structure might consist of developers, administrators, and project and team managers, certain access restrictions can apply. Thus, a development team might need an environment to build and share common Workflow objects. Keeping this in mind, we look at installing and configuring a WebSphere Workflow development environment.

WebSphere MQ Workflow Buildtime is a graphical tool that enables you to model the business processes. The Buildtime component is responsible for depicting your business activities, adding the staff to support such tasks, and providing the programs and network infrastructure that to support the people. In addition, flow of control, information between activities, and modeling are kept in this database.

WebSphere MQ Workflow Buildtime is implemented as a DB2 client application, although you could also use a Microsoft Jet Engine database. However, a typical development environment consists of more than one developer, and usually they share common objects, so a shared DB2 database is often used instead of an individual Jet Engine database. This implies that the DB2 client software is required before we can use the Buildtime tool.

In “Implementing the runtime components” on page 45, we set up a Buildtime database while we were creating the Runtime database. Because of this, we only have to set up a DB2 client connection to the existing Buildtime database on each development machine.

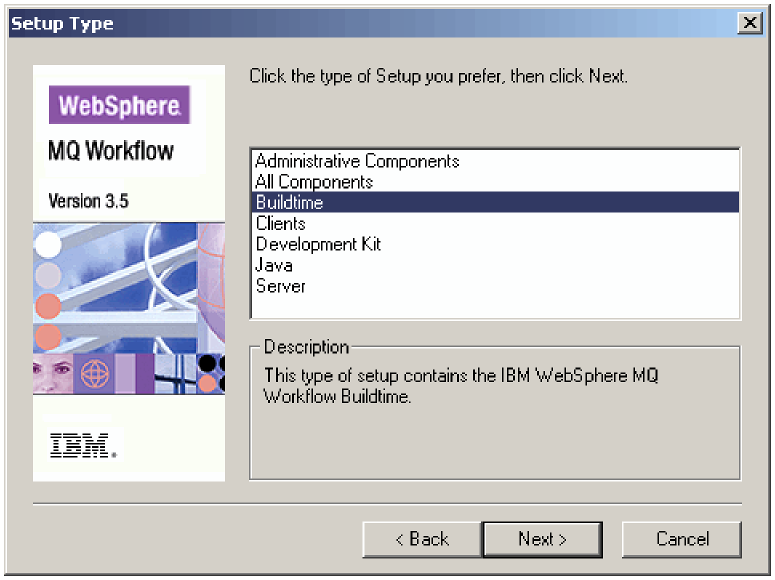
The install and configuration procedures are as follows:

1. Before we can set up the DB2 client connection from the development machine to the database server (WBIDB), we must update the local DB2 catalogs with the remote system information. This is done using the following commands. At the db2cmd prompt execute:

   ```
   catalog tcpip node WBIDB remote <machine name> server 50000 remote_instance
   WBIDB system <server name> ostype win

   catalog database FMCBTDB AS FMCBT at node WBIDB
   ```

   In our example, `WBIDB` is the host name of the database server, `FMCBTDB` is the name of the database on the database server, and `FMCBT` is the local database name.

2. Run the Workflow installation program and select **Buildtime** as the Setup type, as shown in Figure 4-22, and click **Next.**



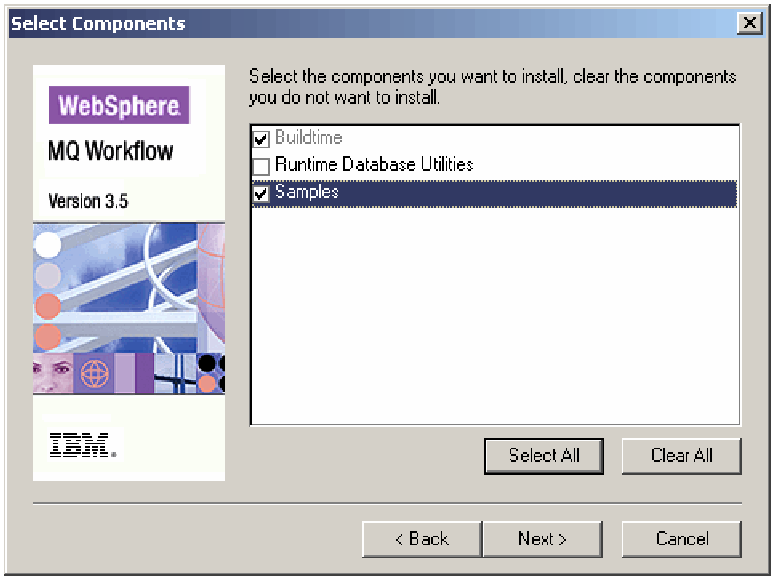
Click the type of Setup you prefer, then click Next.

Administrative Components
All Components
Buildtime
Clients
Development Kit
Java
Server

Description
This type of setup contains the IBM WebSphere MQ Workflow Buildtime.

Figure 4-22 Installation Type for WebSphere MQ Workflow Buildtime

3. For this setup type, there are two selectable components (Figure 4-23); the Buildtime component is preselected. If the developer needs access to the Runtime database (for example, to import the workflow model) select the **Runtime** Database Utilities component as well. Similarly, select the **Samples**, if you want those. Select **Next**.



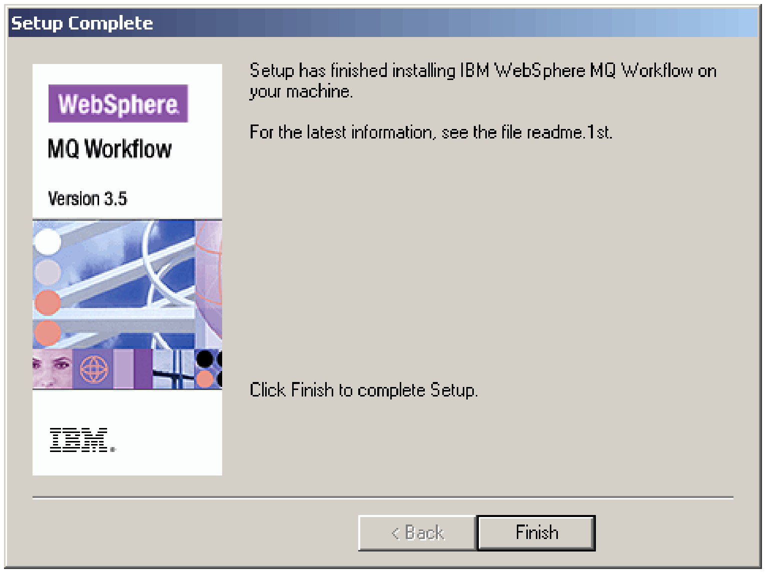
Select the components you want to install, clear the components you do not want to install.

- [✓] Buildtime
- [ ] Runtime Database Utilities
- [✓] Samples

Figure 4-23 WebSphere MQ Workflow Buildtime - select components for installation

4. As with the product installation, you have the option to select a Program Folder where setup will install add program icons. And, of course, you can review or change previously selected install settings. Click **Next**, when you are finished.

5. After the completion of the install you should receive a '`Setup Complete`' message (Figure 4-24). Click **Finished**.


Setup Complete

Setup has finished installing IBM WebSphere MQ Workflow on your machine.

For the latest information, see the file readme.1st.

Click Finish to complete Setup.

Figure 4-24 WebSphere MQ Workflow Buildtime - Setup Complete

6. After the installation is finished, reinstall the FixPak for WebSphere MQ Workflow that was also installed for the Runtime component.

When the product installation is complete (including installing the FixPak), use the configuration utility to setup a profile for Buildtime.

1. To start the configuration utility, select **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **IBM WebSphere MQ Workflow Configuration Utility.** Figure 4-25 opens.

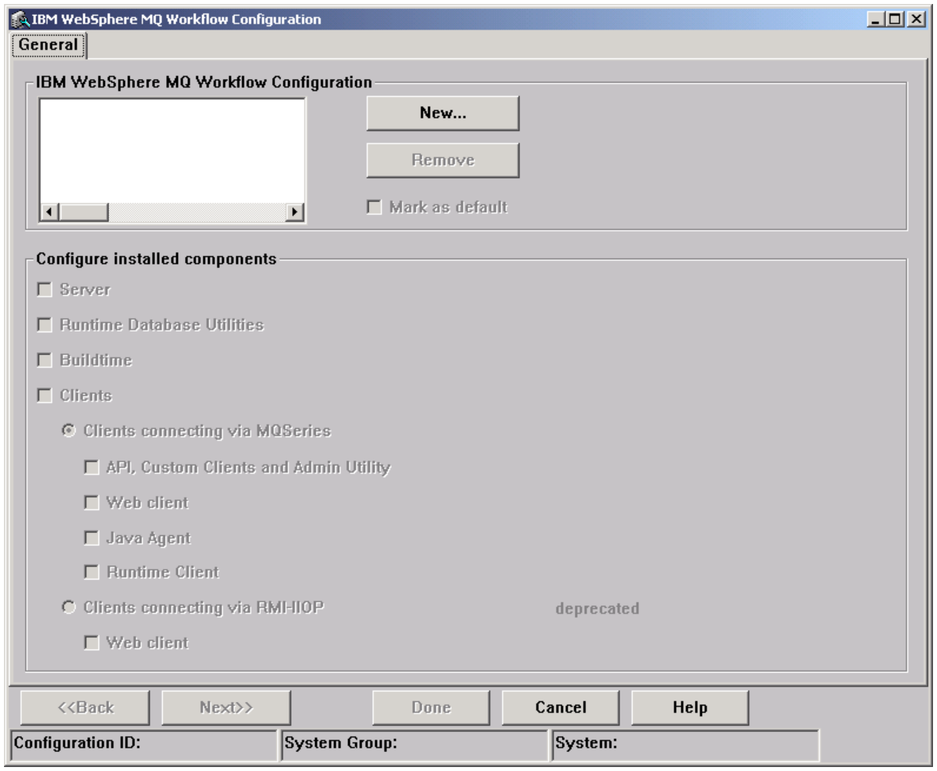
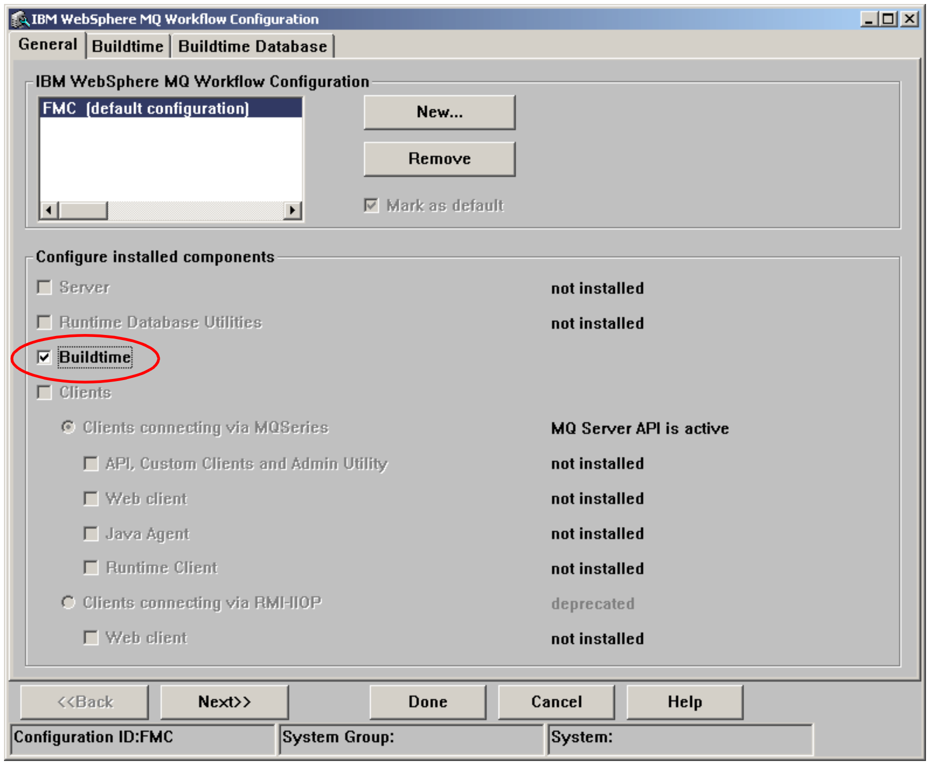

IBM WebSphere MQ Workflow Configuration
General
IBM WebSphere MQ Workflow Configuration
New...
Remove
Mark as default
Configure installed components
Server
Runtime Database Utilities
Buildtime
Clients
Clients connecting via MQSeries
API, Custom Clients and Admin Utility
Web client
Java Agent
Runtime Client
Clients connecting via RMI-IIOP
deprecated
Web client
<<Back
Next>>
Done
Cancel
Help
Configuration ID:
System Group:
System:

Figure 4-25 General tab

2. After entering the Configuration ID (**FMC**), select **Buildtime** as the installed component to configure (Figure 4-26).


IBM WebSphere MQ Workflow Configuration
General
Buildtime
Buildtime Database
IBM WebSphere MQ Workflow Configuration
New...
Remove
Configure installed components
<<Back
Next>>
Done
Cancel
Help

FMC (default configuration)

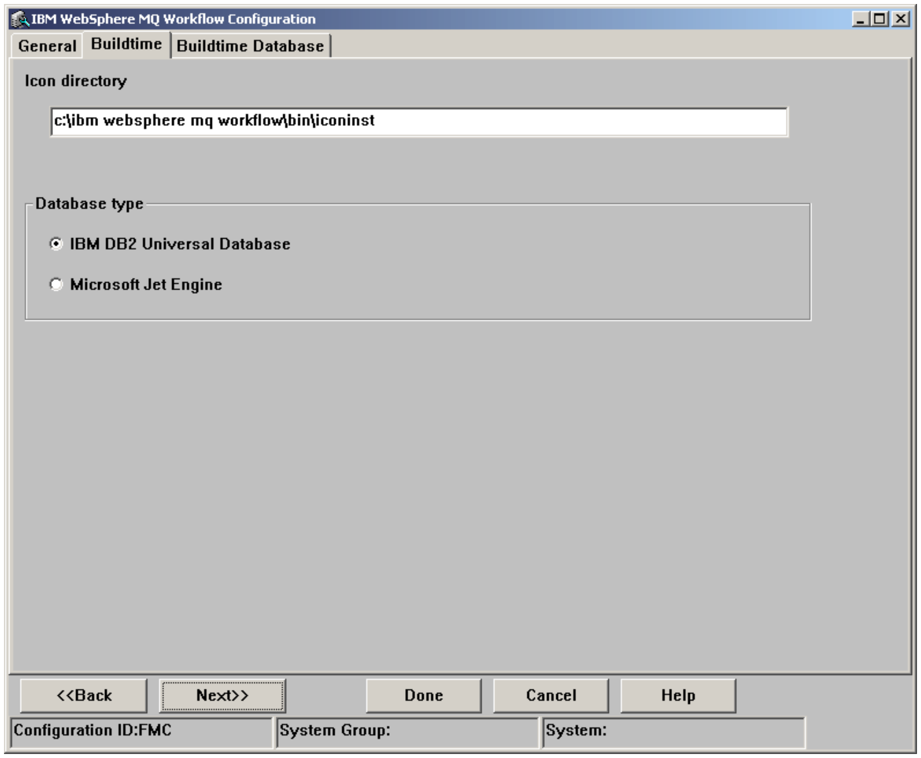
[✓] Mark as default

☐ Server	not installed
☐ Runtime Database Utilities	not installed
[✓] Buildtime	
☐ Clients	
● Clients connecting via MQSeries	MQ Server API is active
☐ API, Custom Clients and Admin Utility	not installed
☐ Web client	not installed
☐ Java Agent	not installed
☐ Runtime Client	not installed
○ Clients connecting via RMI-IIOP	deprecated
☐ Web client	not installed

Configuration ID:FMC
System Group:
System:

Figure 4-26 Buildtime - configure installed components

3. Click **Next** to go to the Buildtime tab. The Buildtime tab is used to specify the database type. We will be connecting to our existing DB2 Buildtime database. So, verify that IBM DB2 Universal Database is selected (Figure 4-27).



Icon directory: c:\ibm websphere mq workflow\bin\iconinst

Database type

(●) IBM DB2 Universal Database

( ) Microsoft Jet Engine

Configuration ID:FMC | System Group: | System:

Figure 4-27 Buildtime tab

4. Select the **Buildtime Database** tab. First we set the DB2 connection properties that will be used to connect to the database server. To do this, click the **DB2 Connect parameters** button (which can be seen in Figure 4-28). Provide a user ID with database administrator authority and it's password.

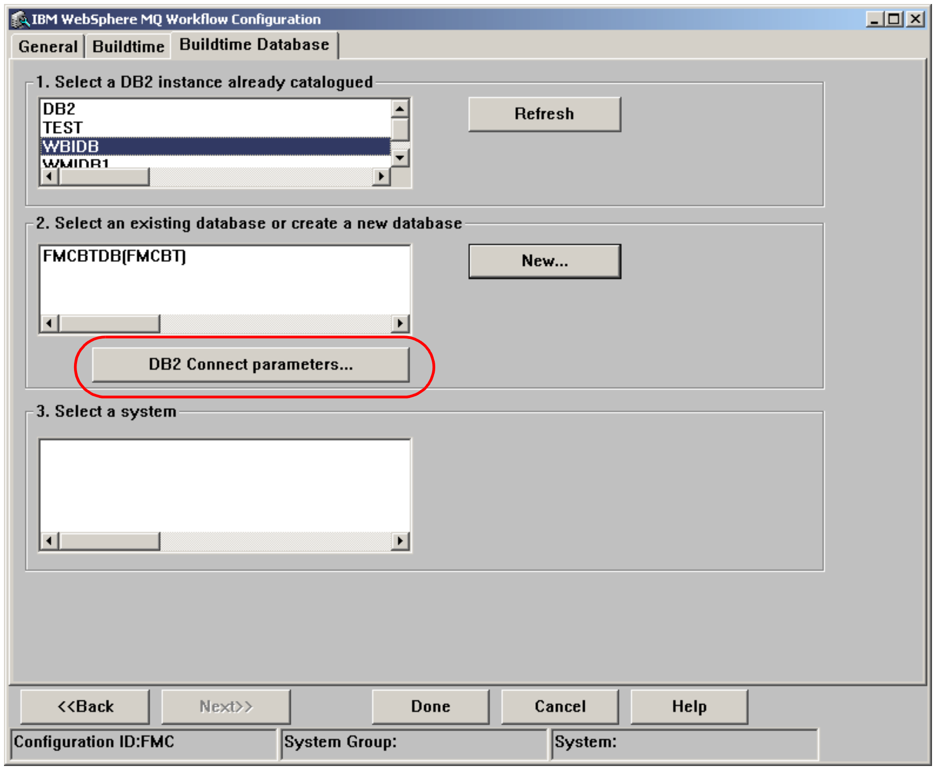


Figure 4-28 Buildtime Database tab

5. Select the catalogued instance **WBIDB**. The existing database appears in list box 2, select an existing database or create a new database, as shown in Figure 4-29. Select the database, and systems appear in the bottom list box, Select a system. Select the workflow system and click **Done**.

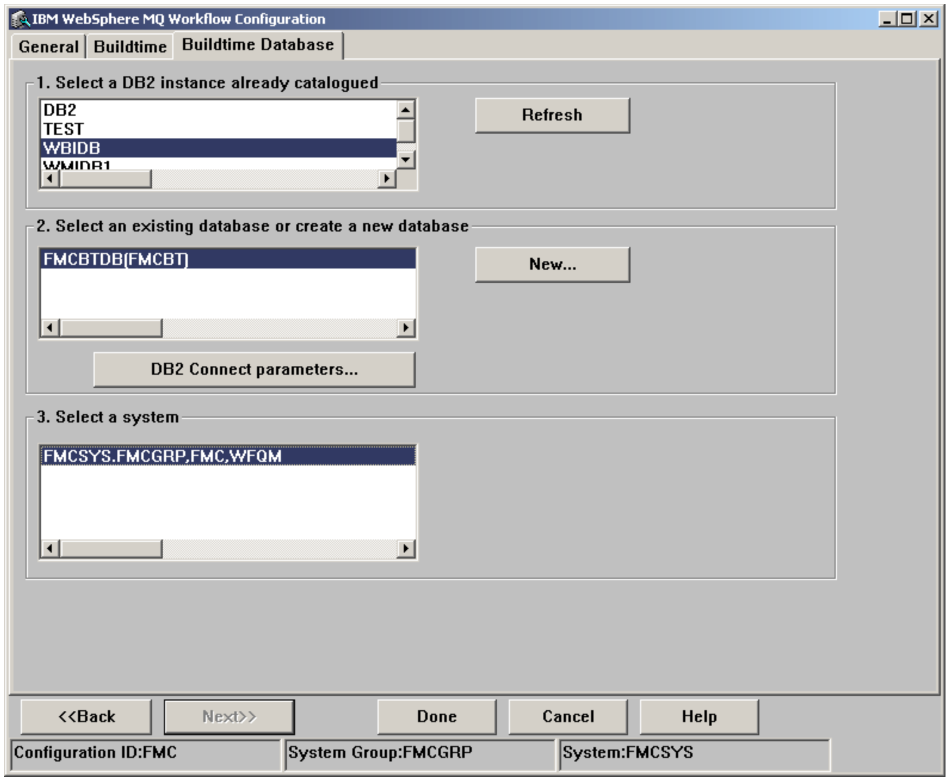


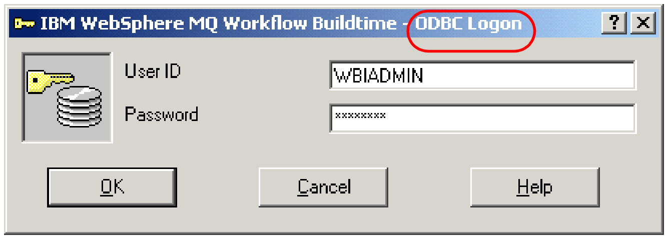
Figure 4-29 Configuring access to an existing Buildtime database

In our example, WBIDB is the previously established instance, FMCBTDB(FMCBT) is the existing database, and our workflow system is FMCSYS,FMCGRP,FMC,WFQM.

We are now ready to start the Buildtime Tool. To start the Buildtime tool:

1. Select **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **WebSphere MQ Workflow Buildtime -** ***FMC***, where *FMC* is the name of the configuration profile.

2. When Buildtime is launched, it might present two logon windows. The first window is the database logon. Enter a database user ID and password (Figure 4-30). Click **OK**.

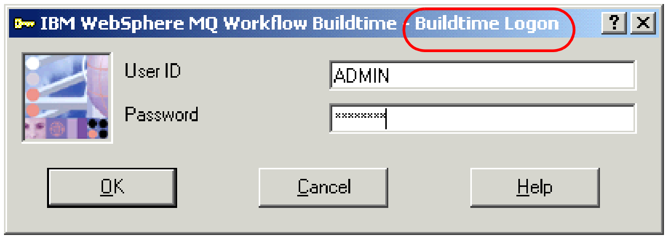

IBM WebSphere MQ Workflow Buildtime - ODBC Logon

User ID: WBIADMIN

Password: ********

Figure 4-30 Database logon window in Buildtime

3. Following a successful database logon, a second window appears requesting a workflow logon. Provide the user ID `ADMIN` (or another user ID defined in the workflow database) and its password (Figure 4-31). Click **OK**.


IBM WebSphere MQ Workflow Buildtime - Buildtime Logon

User ID: ADMIN

Password: ********

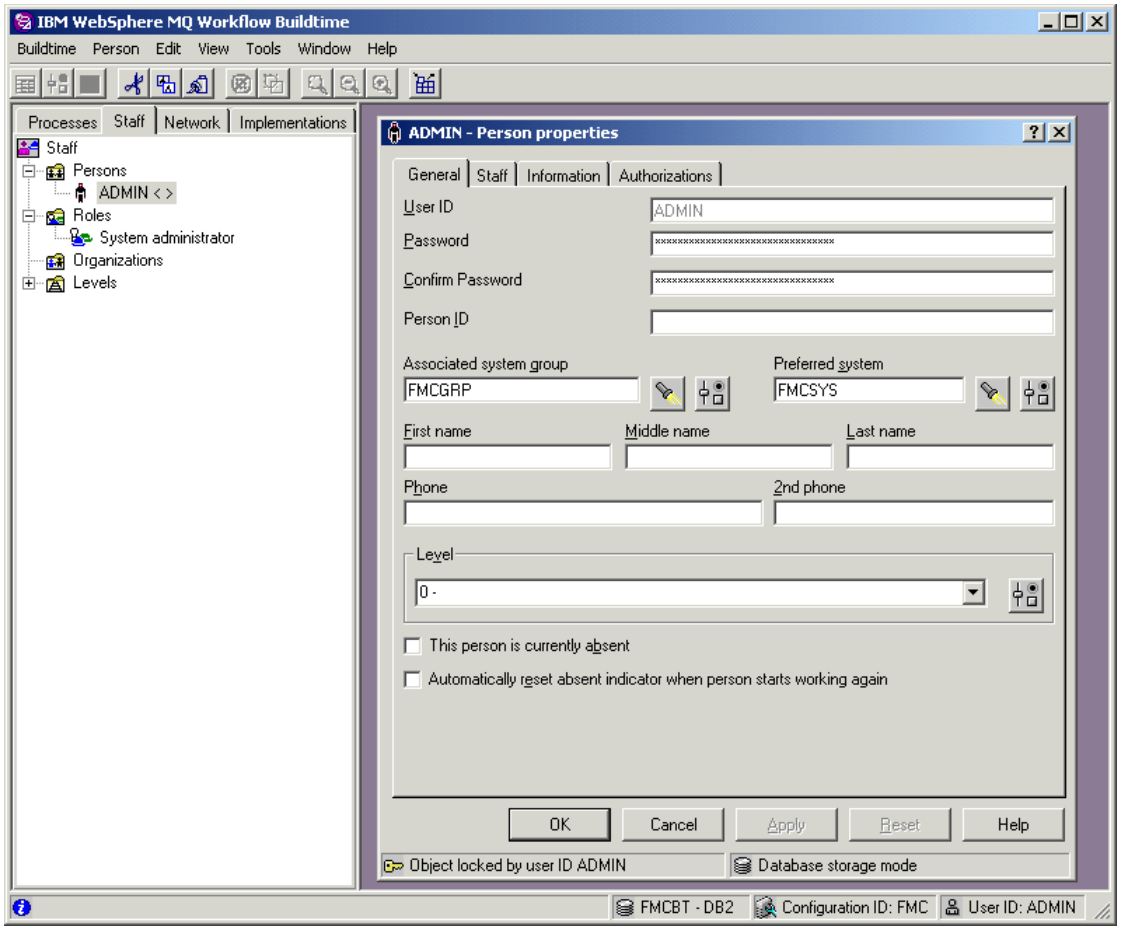
Figure 4-31 Workflow logon window in Buildtime

   The second logon window appears in the exact same position as the first window did. This can cause some confusion because you might think that you have made an error in the first logon window. To know which user ID to provide, check the window heading as circled in Figure 4-31.

   To simplify the start-up of Buildtime, make sure that the Windows logon ID that is used to run Buildtime is a valid and authorized database user ID.

4. When both logon steps are successful, a warning message appears about changing the password of the system administrator ID ADMIN to a nonstandard password. When the tool is launched, select the **Staff** tab and expand the **Persons** folder. Double-click the **user ADMIN** to open the properties, which includes password. Note that changing the password here only affects Buildtime. To update the Runtime password as well, you must export the changes from Buildtime and import them into Runtime.

Note: WebSphere MQ Workflow keeps its own repository of users and user IDs, but bridges exist to support, for example, LDAP-based user repositories.


IBM WebSphere MQ Workflow Buildtime
Buildtime Person Edit View Tools Window Help
Processes Staff Network Implementations
Staff
Persons
ADMIN < >
Roles
System administrator
Organizations
Levels
ADMIN - Person properties
General Staff Information Authorizations
OK Cancel Apply Reset Help

User ID: ADMIN
Password: ********************************
Confirm Password: ********************************
Person ID:
Associated system group: FMCGRP
Preferred system: FMCSYS
First name:
Middle name:
Last name:
Phone:
2nd phone:
Level: 0 -
[ ] This person is currently absent
[ ] Automatically reset absent indicator when person starts working again
Object locked by user ID ADMIN
Database storage mode
FMCBT - DB2 | Configuration ID: FMC | User ID: ADMIN

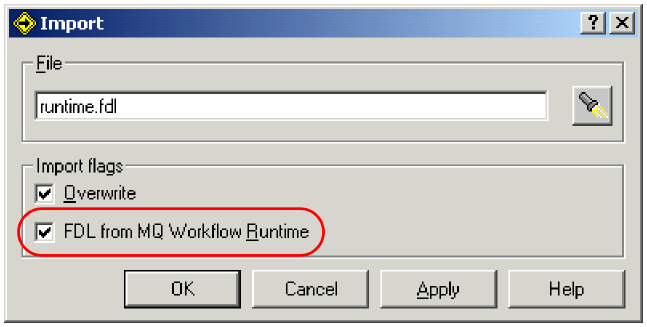
Figure 4-32 Using Buildtime to change password

Tip: Before changing anything in Buildtime, import the runtime configuration into Buildtime so that it matches the runtime setup. This is not required, but it helps keep things consistent.

To import the runtime definitions into Buildtime, first export them from the runtime environment. This command exports the definitions and stores them in the runtime.fdl file:

```
fmcibie -u admin -p password -e runtime.fdl -y FMC
```

5. To import the exported runtime definitions into Buildtime, in the Buildtime tool, select **Buildtime** → **Import** and provide the name of the runtime export file. Make sure to select the option that this file is **FDL from WebSphere MQ Workflow Runtime**, as shown in Figure 4-33.



Import

File: runtime.fdl

Import flags

[✓] Overwrite

[✓] FDL from MQ Workflow Runtime

Figure 4-33 Import runtime definitions in Buildtime

You can now make any changes in Buildtime, export them to an FDL file and import them into the runtime environment. These steps are explained in detail in Chapter 6, “Implementing a process model in WebSphere MQ Workflow” on page 201.

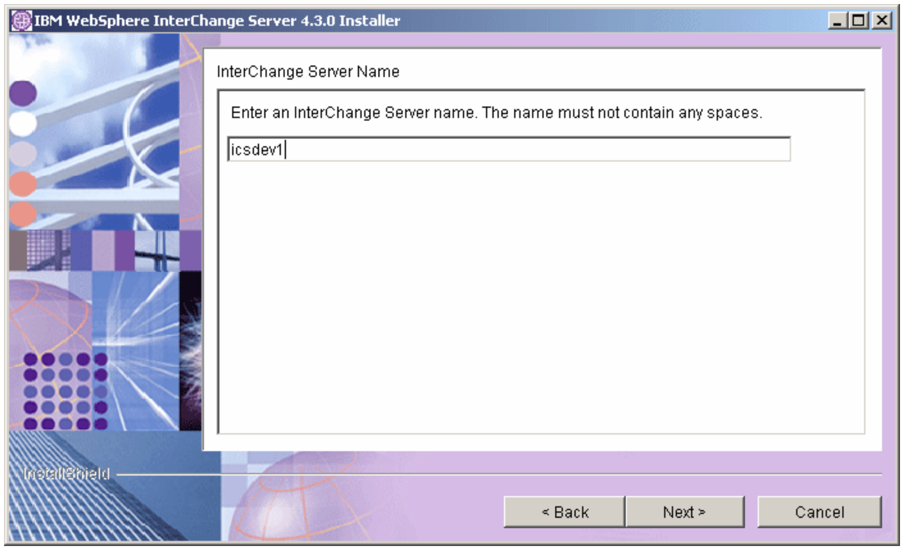
4.2.2 WebSphere InterChange Server development environment

In this section, we show how to start and configure the WebSphere InterChange Server System Manager to work with the InterChange Server that we installed and configured in 3.2.3, “Configuration of the InterChange Server” on page 73.

Also, we want to create a test environment for the InterChange Server developer. This test environment consists of a specially configured InterChange Server instance that is registered in the System Manager as a local test server.

To allow for a true test and development environment, it is common to install a complete InterChange Server platform, including a local DB2 V8 server and a local WebSphere MQ server.

1. Install the base software required by the InterChange Server as defined in the latest WebSphere InterChange Server installation manuals. At the time of this writing, that included IBM JDK 1.4.2, WebSphere MQ Server, and IBM DB2 UDB.
2. When installing WebSphere InterChange Server for development purposes, you select most of the same options as you would for a runtime machine. You need both the development and administration toolsets and the InterChange Server itself as well as any collaborations for your development.
3. During the installation, you are asked to provide the name of the InterChange Server. Choose a name that is unique in the local network.



InterChange Server Name

Enter an InterChange Server name. The name must not contain any spaces.

icsdev1

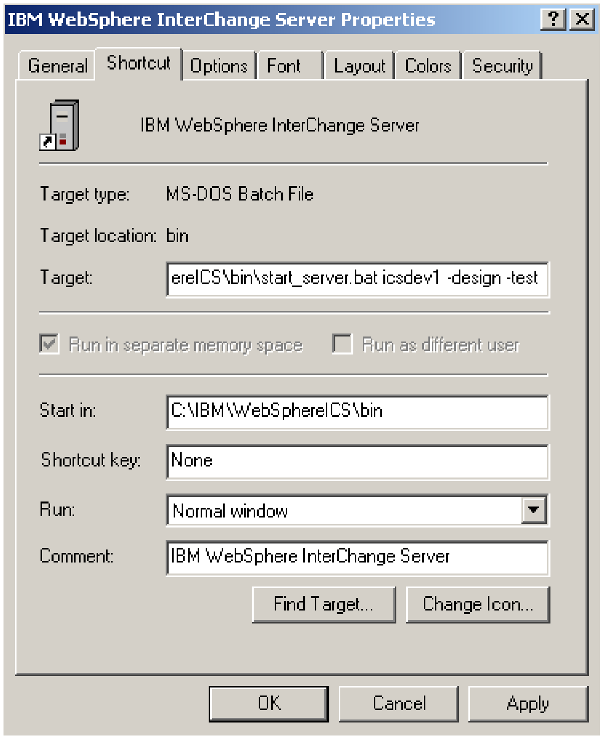
Figure 4-34 Set name for development InterChange Server instance

4. The configuration of a local development server is similar to the creation of a regular server:
   - Create and configure the repository database. (See installation manual.)
   - Create and configure the queue manager.
   - Use the configuration wizard of the InterChange Server.
5. If your development task consists of developing application-specific business objects, developing transformation maps, configuring the WebSphere Business Integration Adapters, or implementing pre-built WebSphere

Business Integration Collaborations, then you will need to install those packages after installing the InterChange Server.

Note: Before installing WebSphere Business Integration Adapters, WebSphere InterChange Server v4.3 now requires a separate installation of the WebSphere Business Integration Adapter Framework. Please review the adapter installation manuals for more details.

6. To be able to run an InterChange Server in test mode, you must add a start-up parameter to the shortcut that launches the InterChange Server. Open the **Properties** of the IBM InterChange Server shortcut and add the `-design -test` option at the end of the command line. (See the Target field in Figure 4-35.)



IBM WebSphere InterChange Server

Target type: MS-DOS Batch File

Target location: bin

Target: erelCS\bin\start_server.bat icsdev1 -design -test

[✓] Run in separate memory space   [ ] Run as different user

Start in: C:\IBM\WebSphereICS\bin

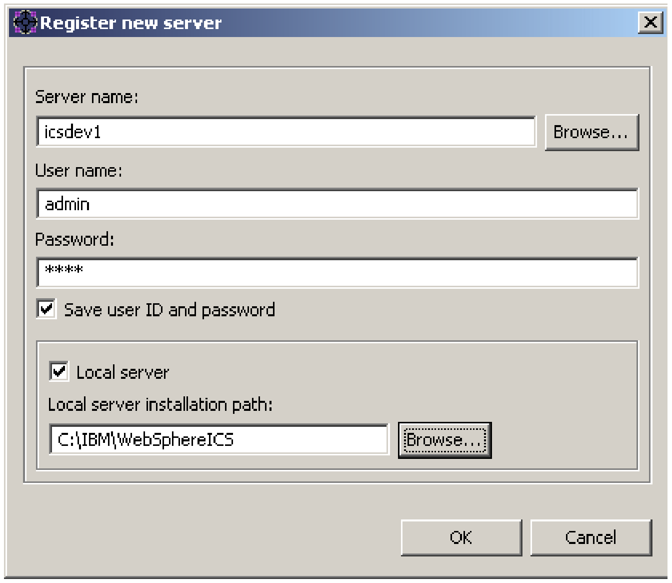
Shortcut key: None

Run: Normal window

Comment: IBM WebSphere InterChange Server

Figure 4-35 Running the InterChange Server in design mode

7. When the InterChange Server has completed its first startup, you can start the System Manager and register the new design server. This time, during server registration, make sure to set the **Local Server** option (Figure 4-36).



Server name: icsdev1

User name: admin

Password: ****

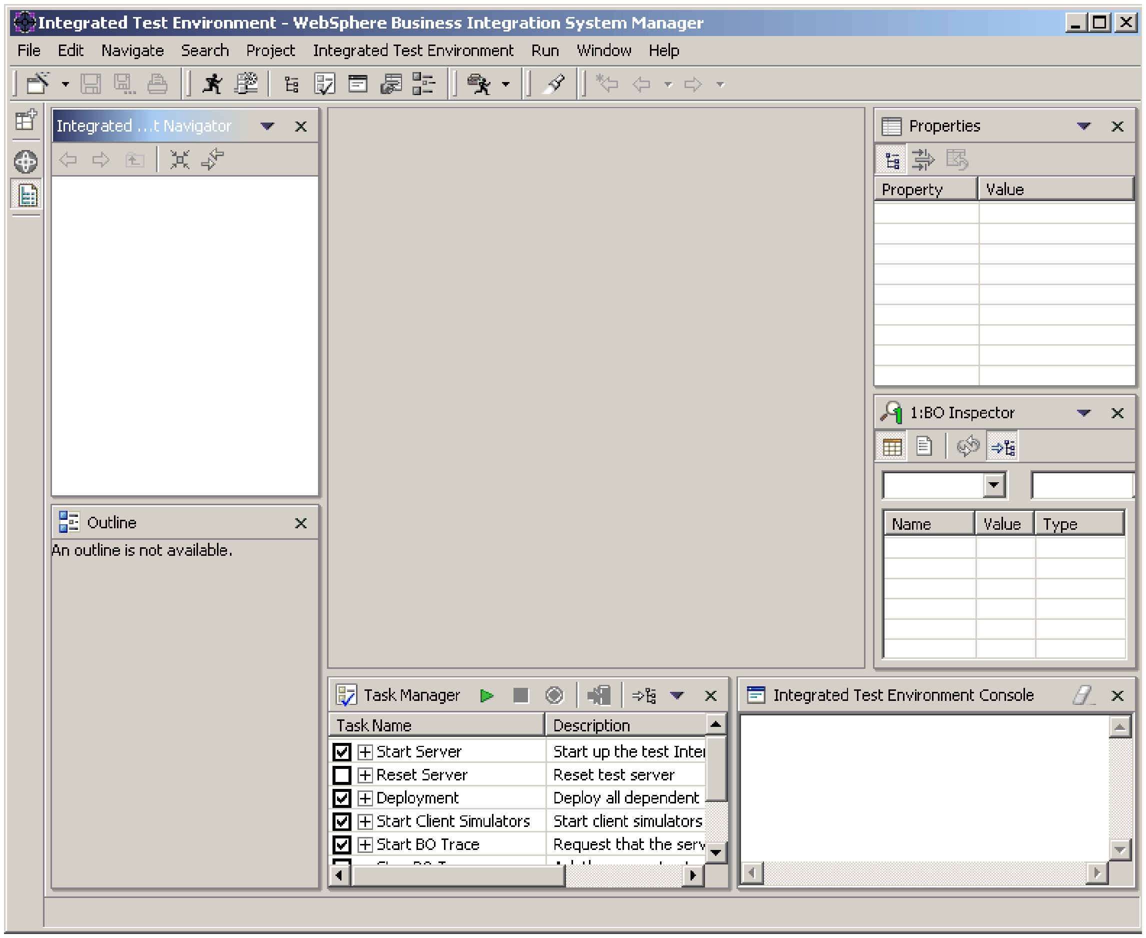
[✓] Save user ID and password

[✓] Local server

Local server installation path: C:\IBM\WebSphereICS

Figure 4-36 Register InterChange Server as a Test Server

8. When connected, open the perspective Integrated Test Environment (**Window** → **Open Perspective**). The interface should look similar to Figure 4-37 on page 156.


Integrated Test Environment - WebSphere Business Integration System Manager
File Edit Navigate Search Project Integrated Test Environment Run Window Help
Integrated ...t Navigator
Properties
Property Value
1:BO Inspector
Name Value Type
Outline
Task Manager
Integrated Test Environment Console

An outline is not available.

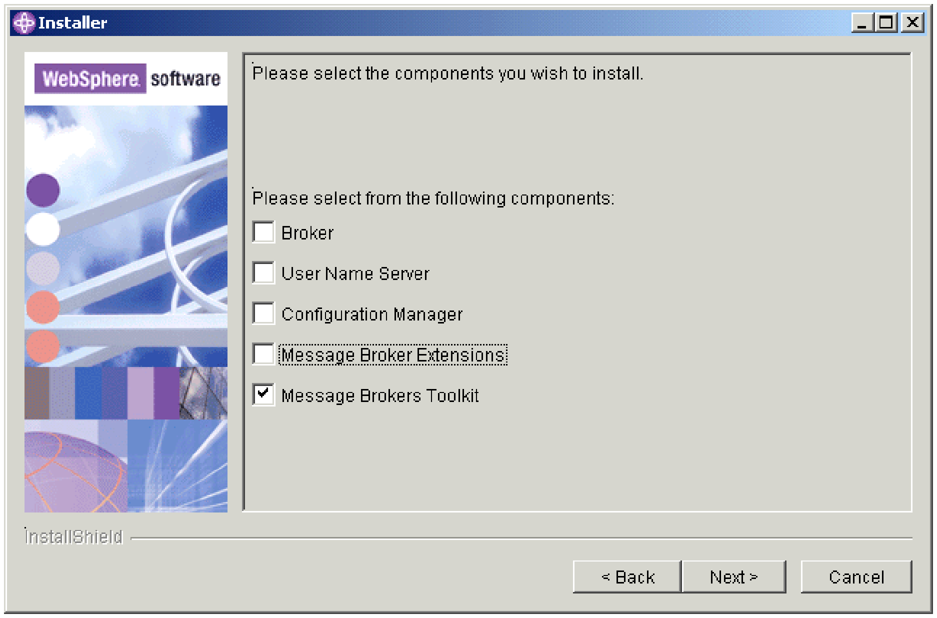
Task Name	Description
☑ Start Server	Start up the test Inte
☐ Reset Server	Reset test server
☑ Deployment	Deploy all dependent
☑ Start Client Simulators	Start client simulators
☑ Start BO Trace	Request that the serv

Figure 4-37 Integrated Test Environment

4.2.3 WebSphere BI Message Broker development environment

Installation of the Broker Toolkit for development purposes has no major prerequisites. The toolkit does require Microsoft Data Access Components (MDAC), which is provided on the supplemental CD.

When installing the product, select the installation option **Custom** and choose to install only the **Message Brokers Toolkit** component. (Figure 4-38)



Please select the components you wish to install.

Please select from the following components:

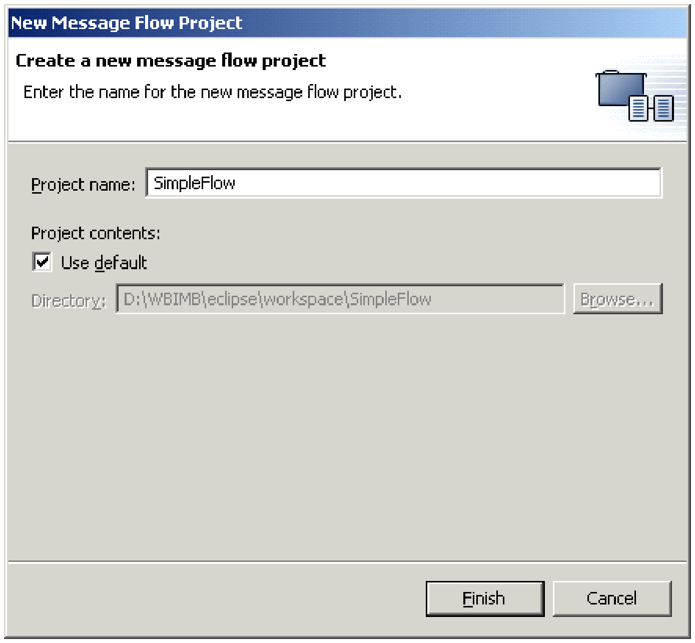
- [ ] Broker
- [ ] User Name Server
- [ ] Configuration Manager
- [ ] Message Broker Extensions
- [✓] Message Brokers Toolkit

Figure 4-38 Select components to install

In general, as a developer, you would like to have an environment that enables you to debug a solution. Facilities within the Message Brokers Toolkit exist. One of those facilities is a test server. However, before we can create a test server, we need to have a deployable component, such as a message flow.

To create a message flow, follow these steps:

1. We first create a message flow project. Select **File** → **New** → **Message flow project**. Give it a name such as `SimpleFlow` (Figure 4-39).


New Message Flow Project

Create a new message flow project

Enter the name for the new message flow project.

Project name: SimpleFlow

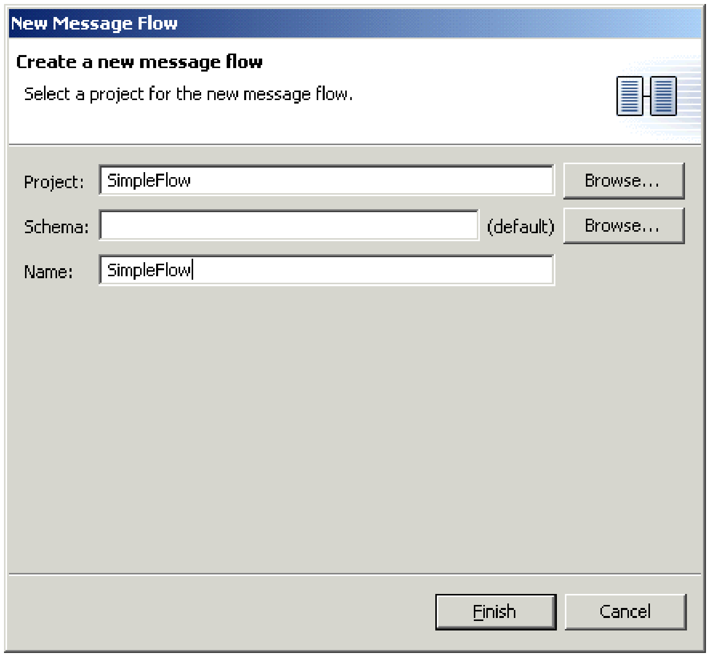
Project contents:

[✓] Use default

Directory: D:\WBIMB\eclipse\workspace\SimpleFlow

Figure 4-39 Create message flow project

2. Within this message flow project, create a message flow by selecting **File** → **New** → **Message flow**. Provide a name for this flow (Figure 4-40).


New Message Flow
Create a new message flow
Select a project for the new message flow.
Browse...
Browse...
Finish
Cancel

Project: SimpleFlow
Schema: (default)
Name: SimpleFlow

Figure 4-40 Create a message flow

After the message flow editor is opened, we can actually build the flow. For our purpose of setting up a test server, the flow can be simple: an MQInput node and MQOutput node that are connected to each other will do fine. This flow simply copies messages from one queue to another.

1. Open the properties of the MQInput node and select the category **Basic**. Provide a queue name for input.
2. Change to the **Default** tab and set the message domain to **XML**.
3. Open the properties of the MQOutput node and select the category **Basic**. Provide a queue name for output.
4. Connect from the output terminal of the MQInput node to the input terminal of the MQOutput node.
5. Save the message flow.
6. Log on to the broker machine and define the queues that are named in the MQOutput and MQInput nodes to the queue manager of the broker. See Figure 4-41 on page 160.

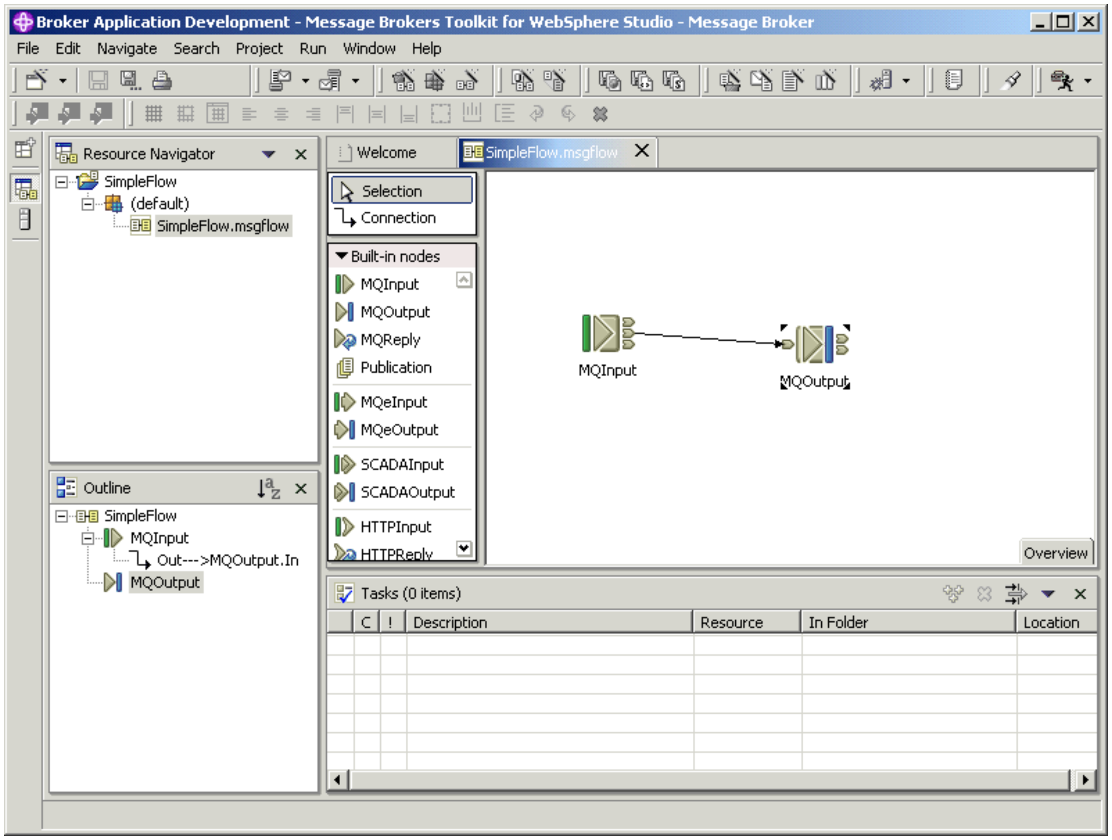
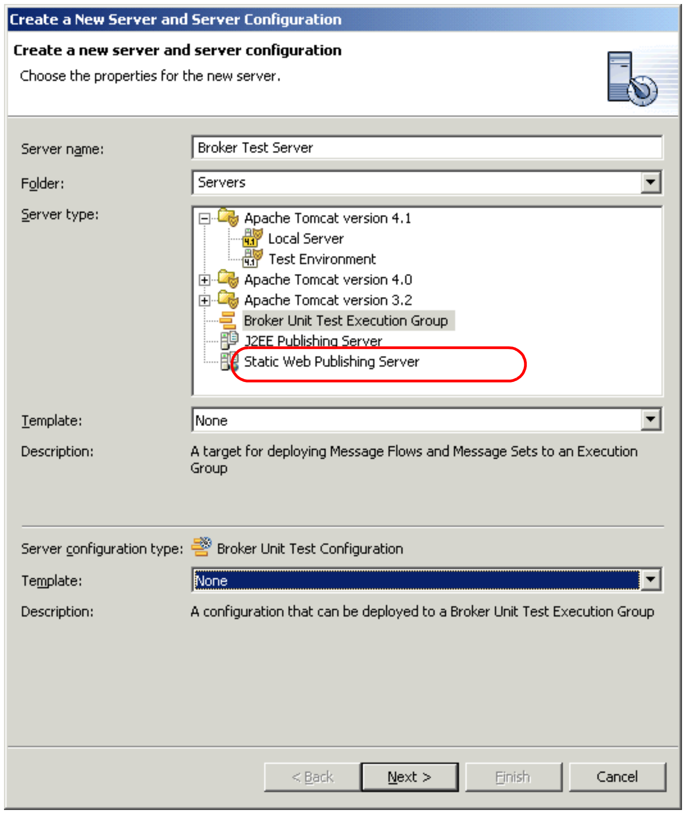


Figure 4-41 Simple message flow

7. Switch to the Server perspective by selecting **Window** → **Open Perspective** → **Other**. Select **Server** from the presented list.

8. Select **File** → **New** → **Server and Server Configuration** to create a test server. Provide a name for the server and select the correct type, which is **Broker Unit Test Execution Group**. (Figure 4-42) 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: Broker Test Server

Folder: Servers

Server type:
- Apache Tomcat version 4.1
  - Local Server
  - Test Environment
- Apache Tomcat version 4.0
- Apache Tomcat version 3.2
- Broker Unit Test Execution Group
- J2EE Publishing Server
- Static Web Publishing Server

Template: None

Description: A target for deploying Message Flows and Message Sets to an Execution Group

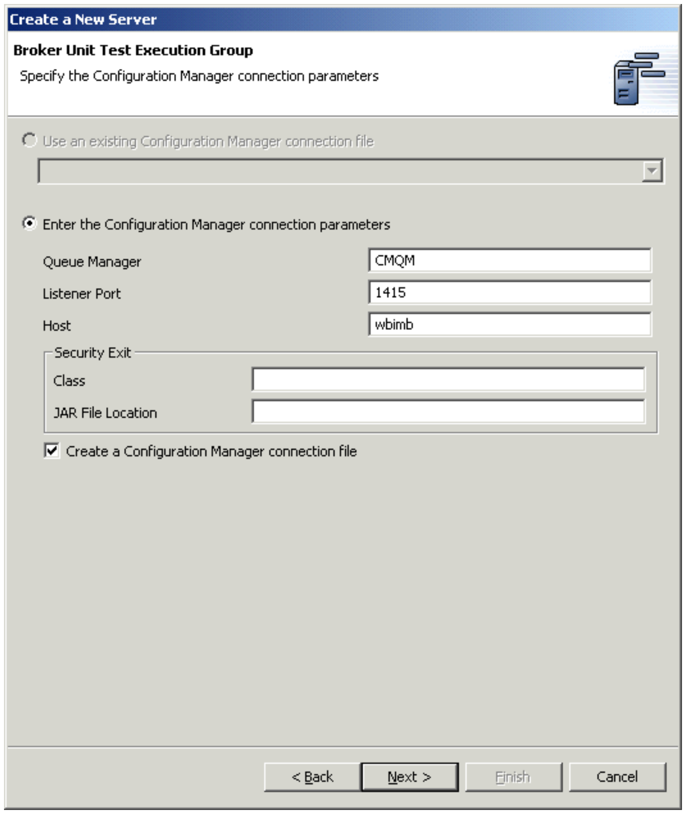
Server configuration type: Broker Unit Test Configuration

Template: None

Description: A configuration that can be deployed to a Broker Unit Test Execution Group

Figure 4-42 Create new server and server configuration

9. The Broker Unit Text Execution Group page (Figure 4-43) is used to provide connection information for the configuration manager. Enter the queue manager name (`CMQM`), the host name (`wbimb`), and the port number for TCP/IP (`1415`). Click **Next**.



Broker Unit Test Execution Group

Specify the Configuration Manager connection parameters

○ Use an existing Configuration Manager connection file

● Enter the Configuration Manager connection parameters

Queue Manager	CMQM
Listener Port	1415
Host	wbimb

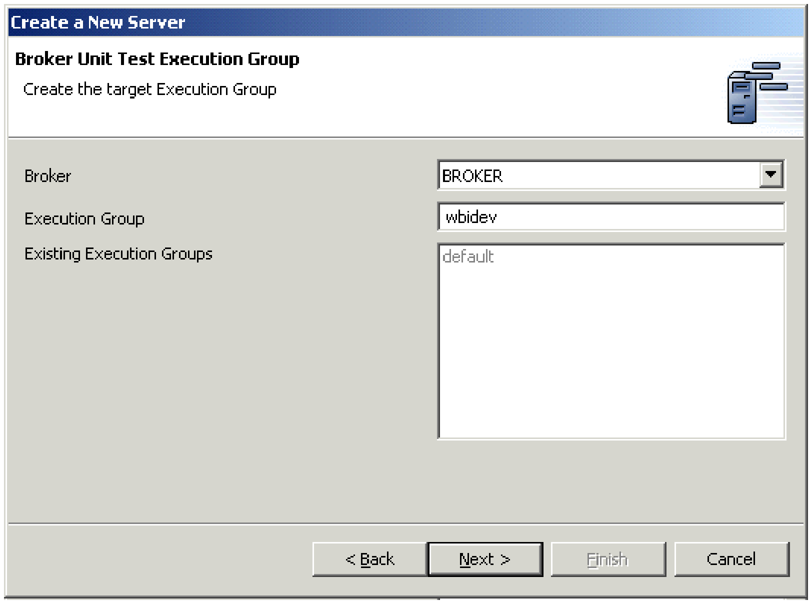
Security Exit

Class	
JAR File Location	

☑ Create a Configuration Manager connection file

Figure 4-43 Provide information about the configuration manager

10.At this point, the toolkit retrieves broker topology information from the configuration manager. The next window (Figure 4-44) enables you to select the existing broker to which you want to add a test execution group. By default, the new execution group is named after the user ID with which you are logged on. The name of the execution group has to be unique. Click **Next**.



Create a New Server

Broker Unit Test Execution Group

Create the target Execution Group

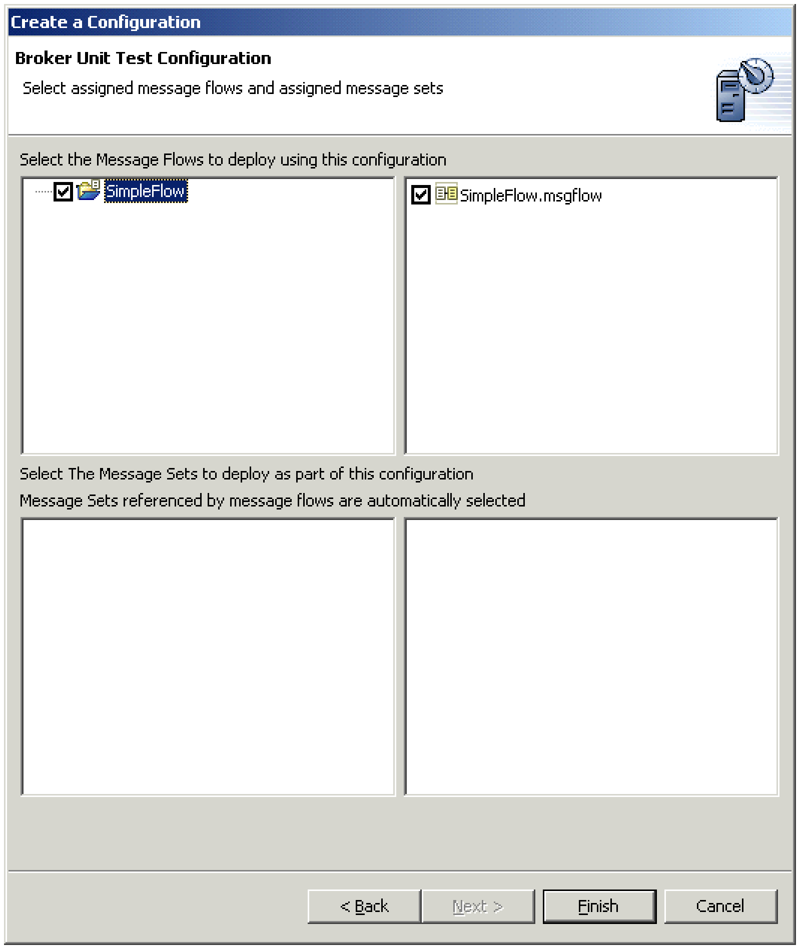
Broker: BROKER

Execution Group: wbidev

Existing Execution Groups: default

Figure 4-44 Select broker and name the execution group

11. The final step is to select solution components to run in the test server. As of now, we have only one message flow to add. Select it and click **Finish**. (Figure 4-45)



Create a Configuration

Broker Unit Test Configuration

Select assigned message flows and assigned message sets

Select the Message Flows to deploy using this configuration

☑ SimpleFlow

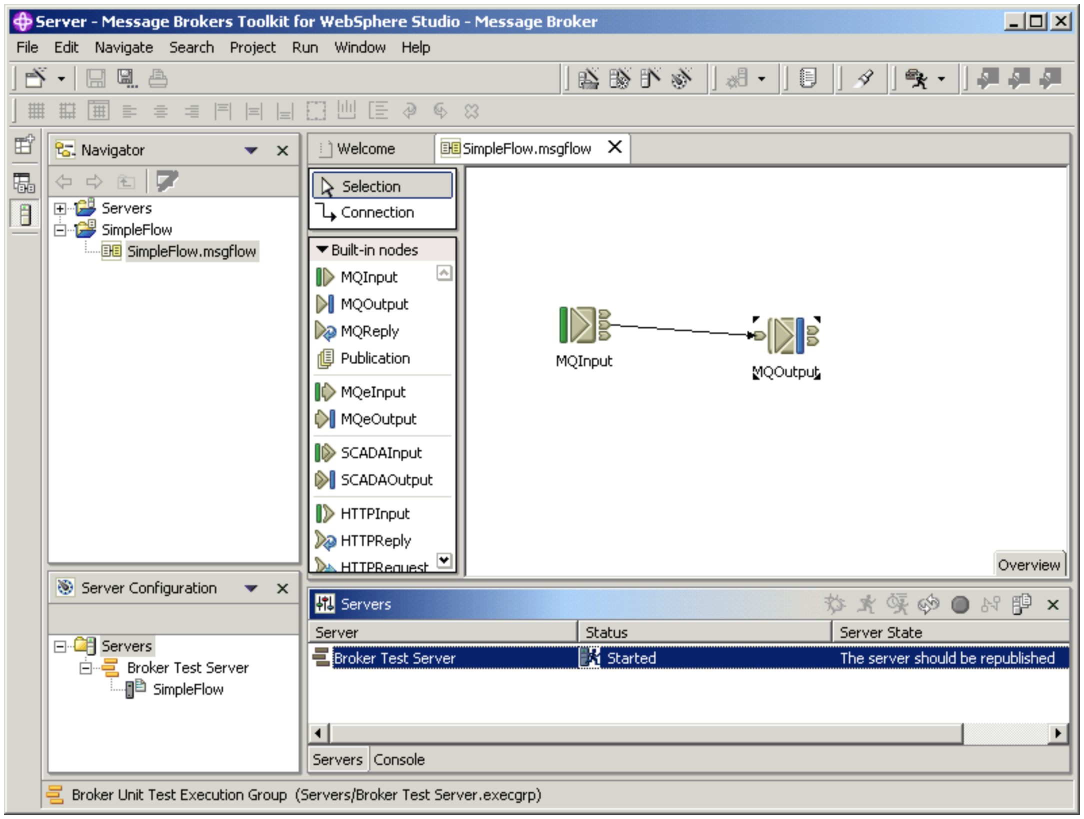
☑ SimpleFlow.msgflow

Select The Message Sets to deploy as part of this configuration

Message Sets referenced by message flows are automatically selected

Figure 4-45 Select message flow to run in test server

The test server is now created (Figure 4-46 on page 165). From now on, for any given project, we can simply select **Run on Server** or **Debug on Server**, making it easier to debug a flow.



Server	Status	Server State
Broker Test Server	Started	The server should be republished

Broker Unit Test Execution Group (Servers/Broker Test Server.execgrp)

Figure 4-46 Server perspective in the Brokers Toolkit

4.3 Implementing a management client

This section discusses the implementation of a management client for each of our runtime environments. From a functional perspective, we want to provide an environment that enables the administrator to deploy solutions to the runtime, monitor the runtime, and possibly alter the runtime to correct any error condition that may occur.

4.3.1 WebSphere MQ Workflow management client

As a workflow system administrator, you need access to the following tools:

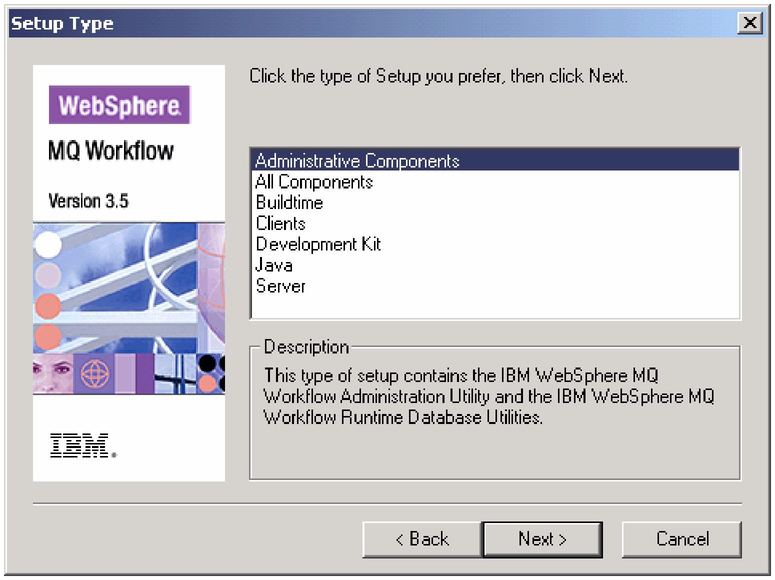
- The WebSphere MQ Workflow administration utility (command interface) which enables you to stop and start components, view logs, and query status and configuration information.
- A Web browser based client (Web Client) for interacting with the Runtime environment which allows the administrator to monitor and work with process instances.
- Database utilities that are part of the workflow product to manage the workflow database.

To be able to use the administration utility, the administrator's machine must have at minimum the WebSphere MQ Client software, which can be downloaded for free from the Internet. To use the database utilities, DB2 client software also must be installed. The database server should be catalogued as a remote instance on the client machine.

To install the relevant WebSphere MQ Workflow components, run the same installation program as you did for the runtime server.

1. Launch the WebSphere MQ Workflow setup, accept the license agreement, and select the install language.

2. Select **Administrative Components** as the setup type (Figure 4-47). Click **Next**.



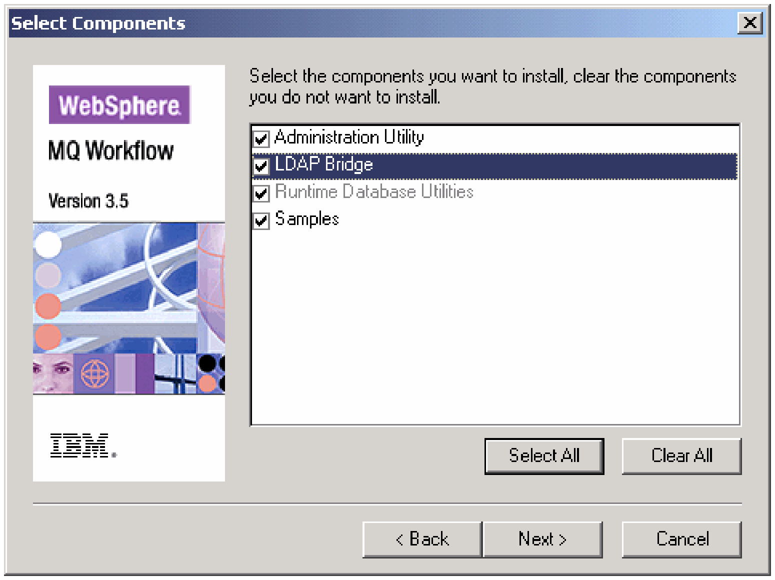
Click the type of Setup you prefer, then click Next.

Administrative Components
All Components
Buildtime
Clients
Development Kit
Java
Server

Description
This type of setup contains the IBM WebSphere MQ Workflow Administration Utility and the IBM WebSphere MQ Workflow Runtime Database Utilities.

Figure 4-47 Select components to install

3. In the next window (Figure 4-48), we choose to install all of the available product components. (Runtime Database Utilities is automatically selected when LDAP is selected.)



Select the components you want to install, clear the components you do not want to install.

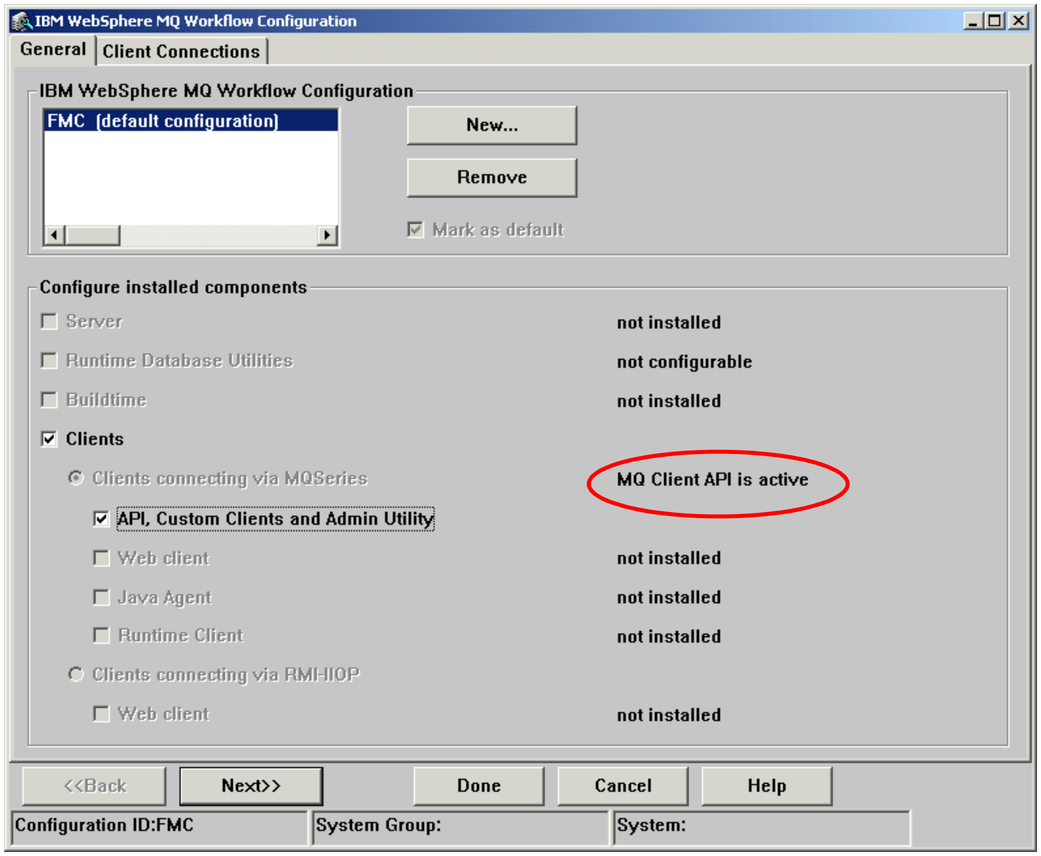
- ✓ Administration Utility
- ✓ LDAP Bridge
- ✓ Runtime Database Utilities
- ✓ Samples

Figure 4-48 Select product components to install

When the installation is complete, the configuration utility is started.

4. Click the **New** button and provide a name for the configuration profile. Check **Clients** and **API, Custom Clients and Admin Utility**, as shown in Figure 4-49 on page 169.

Note: The administration utility can use the MQ Client and MQ Server API. If you have the full WebSphere MQ product on your system, the configuration utility will automatically configure the administration utility to use the MQ Server API to define and configure a queue manager for you. On Windows platforms, you cannot force the administration utility or any other workflow client component to use the MQ Client API. On UNIX® platforms, you can set the environment variable MQConnectionType to MQClient so that the MQ Client API will be used.


IBM WebSphere MQ Workflow Configuration

General | Client Connections

IBM WebSphere MQ Workflow Configuration

FMC (default configuration)

☑ Mark as default

Configure installed components

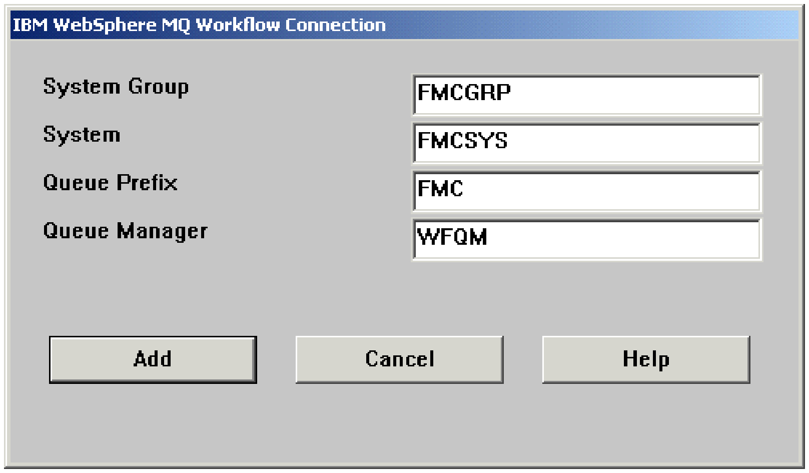
☐ Server	not installed
☐ Runtime Database Utilities	not configurable
☐ Buildtime	not installed
☑ Clients	
● Clients connecting via MQSeries	MQ Client API is active
☑ API, Custom Clients and Admin Utility	
☐ Web client	not installed
☐ Java Agent	not installed
☐ Runtime Client	not installed
○ Clients connecting via RMI-IIOP	
☐ Web client	not installed

Configuration ID:FMC | System Group: | System:

Figure 4-49 Select components to configure

5. Click the **Client Connections** tab and enter the name and location of the client channel connection file that is stored and maintained on the server. Copy this file from the server to the workstation and provide the name and folder to the configuration utility.

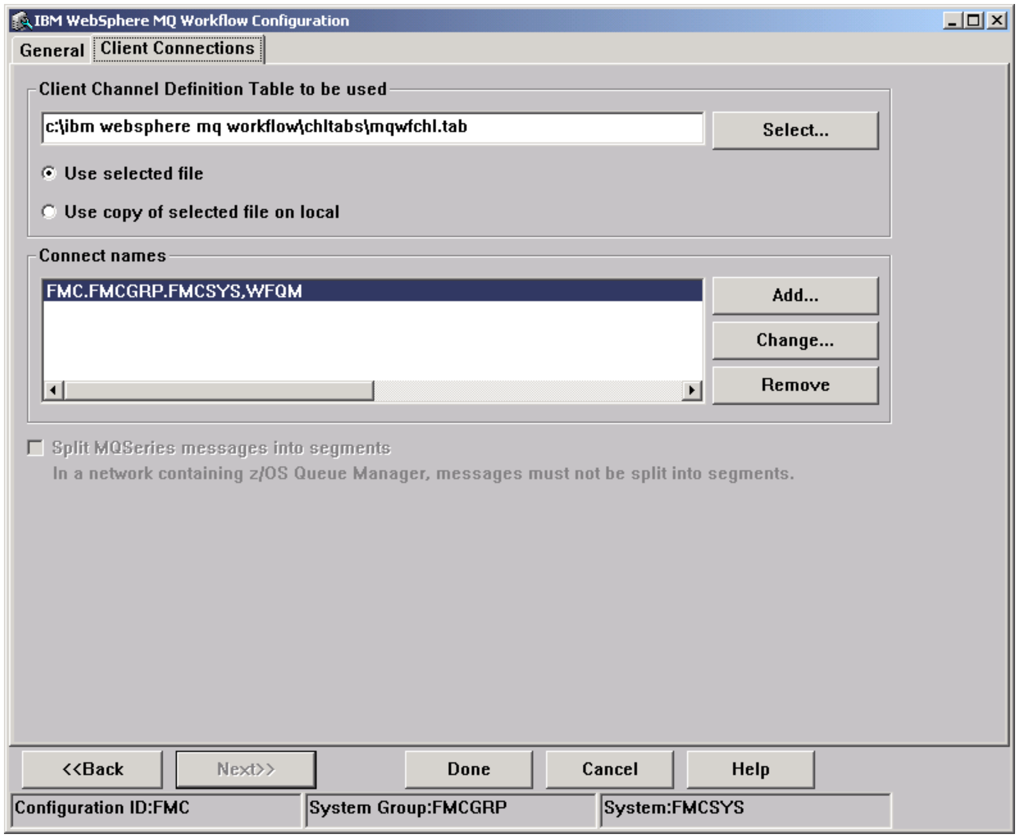
6. To provide information about the workflow system that you want to administer, click the **Add** button and provide the System Group, System (name), Queue Prefix, and Queue Manager, as shown in Figure 4-50.


IBM WebSphere MQ Workflow Connection

System Group	FMCGRP
System	FMCSYS
Queue Prefix	FMC
Queue Manager	WFQM

Figure 4-50   Providing connection information

You can add more connections to the configuration if you have more than one workflow server or workflow system group. Figure 4-51 shows the completed Client Connections tab.



Client Channel Definition Table to be used

c:\ibm websphere mq workflow\chltabs\mqwfchl.tab

(●) Use selected file

( ) Use copy of selected file on local

Connect names

FMC.FMCGRP.FMCSYS,WFQM

[ ] Split MQSeries messages into segments

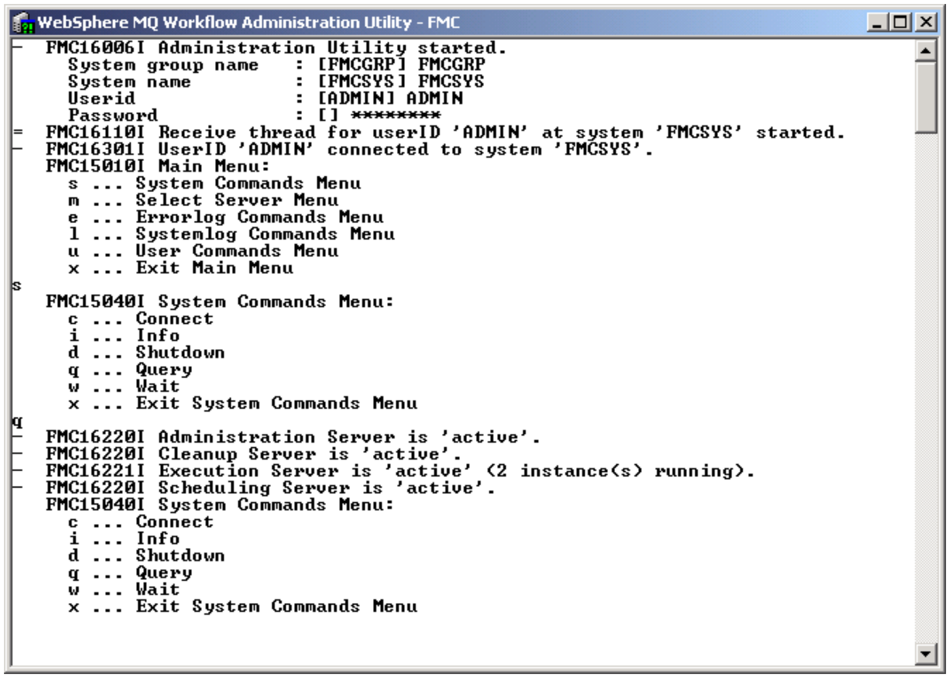
In a network containing z/OS Queue Manager, messages must not be split into segments.

Configuration ID:FMC | System Group:FMCGRP | System:FMCSYS

Figure 4-51 Client configuration complete

To verify that the configuration is working on the client system, select **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **WebSphere MQ Workflow Administration Utility - FMC**.

When the utility starts, provide the password of the ADMIN user ID, which is normally the word `password`. Figure 4-52 shows a sequence of commands that result in providing the status of the workflow server on the remote machine. Using this utility the administrator can also consult error and system logs.



```
- FMC16006I Administration Utility started.
    System group name    : [FMCGRP] FMCGRP
    System name          : [FMCSYS] FMCSYS
    Userid               : [ADMIN] ADMIN
    Password             : [] ********
= FMC16110I Receive thread for userID 'ADMIN' at system 'FMCSYS' started.
- FMC16301I UserID 'ADMIN' connected to system 'FMCSYS'.
  FMC15010I Main Menu:
    s ... System Commands Menu
    m ... Select Server Menu
    e ... Errorlog Commands Menu
    l ... Systemlog Commands Menu
    u ... User Commands Menu
    x ... Exit Main Menu
s
  FMC15040I System Commands Menu:
    c ... Connect
    i ... Info
    d ... Shutdown
    q ... Query
    w ... Wait
    x ... Exit System Commands Menu
q
- FMC16220I Administration Server is 'active'.
- FMC16220I Cleanup Server is 'active'.
- FMC16221I Execution Server is 'active' (2 instance(s) running).
- FMC16220I Scheduling Server is 'active'.
  FMC15040I System Commands Menu:
    c ... Connect
    i ... Info
    d ... Shutdown
    q ... Query
    w ... Wait
    x ... Exit System Commands Menu
```

Figure 4-52 WebSphere MQ Workflow Administration Utility

4.3.2 InterChange Server management client

Management tools for the InterChange Server can be classified in two categories:

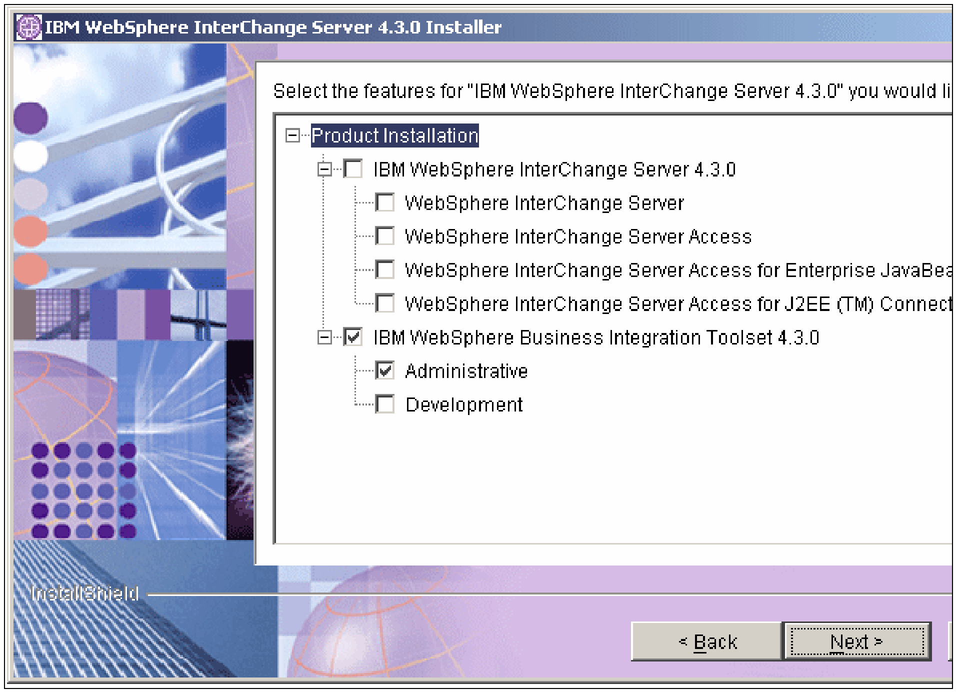
- Standalone GUI tools, such as the System Manager and the Log Viewer
- Web-based tools, such as the System Monitor

In this section we step through the implementation of these two categories of tools and describe their function. The actions are described from the perspective of an administrator that has requirements to install the administration tools on a separate windows workstation.

Implementing the System Manager

The System Manager and other ICS related administration tools do not require any other software, such as WebSphere MQ. The System Manager interacts with the InterChange Server via the name server or object request broker (ORB).

1. Start the installation program for the InterChange Server and select only the **Administrative** toolset, as shown in Figure 4-53.

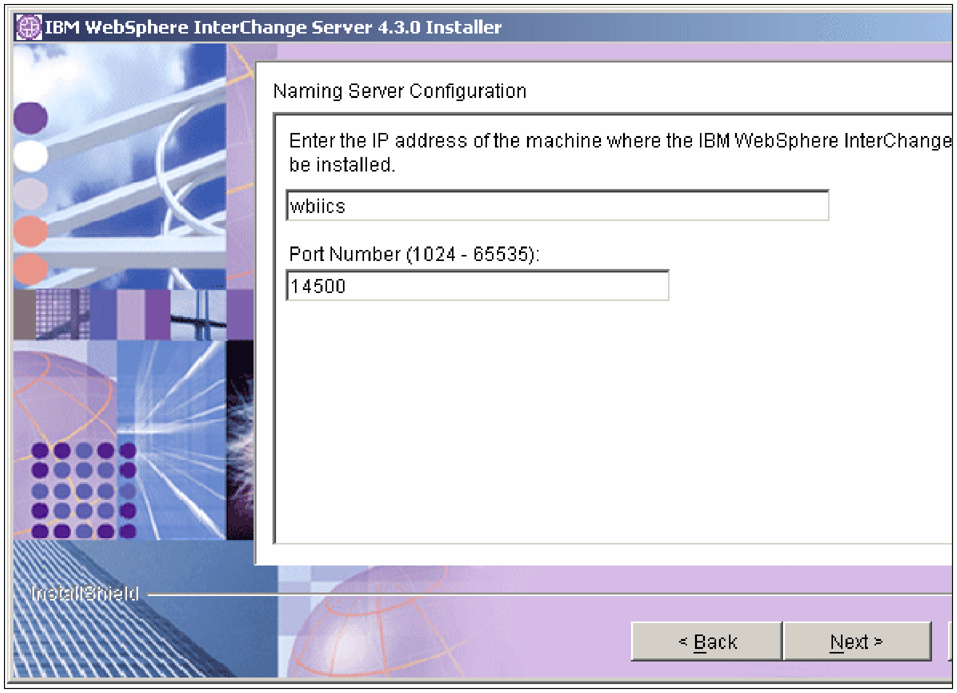


Select the features for "IBM WebSphere InterChange Server 4.3.0" you would li

- Product Installation
  - [ ] IBM WebSphere InterChange Server 4.3.0
    - [ ] WebSphere InterChange Server
    - [ ] WebSphere InterChange Server Access
    - [ ] WebSphere InterChange Server Access for Enterprise JavaBea
    - [ ] WebSphere InterChange Server Access for J2EE (TM) Connect
  - [✓] IBM WebSphere Business Integration Toolset 4.3.0
    - [✓] Administrative
    - [ ] Development

Figure 4-53 Select InterChange Server components to install

2. Enter the host name and port number of the ORB, which is `14500` by default. The host name for our environment is `wbiics`.



IBM WebSphere InterChange Server 4.3.0 Installer

Naming Server Configuration

Enter the IP address of the machine where the IBM WebSphere InterChange
be installed.

wbiics

Port Number (1024 - 65535):

14500

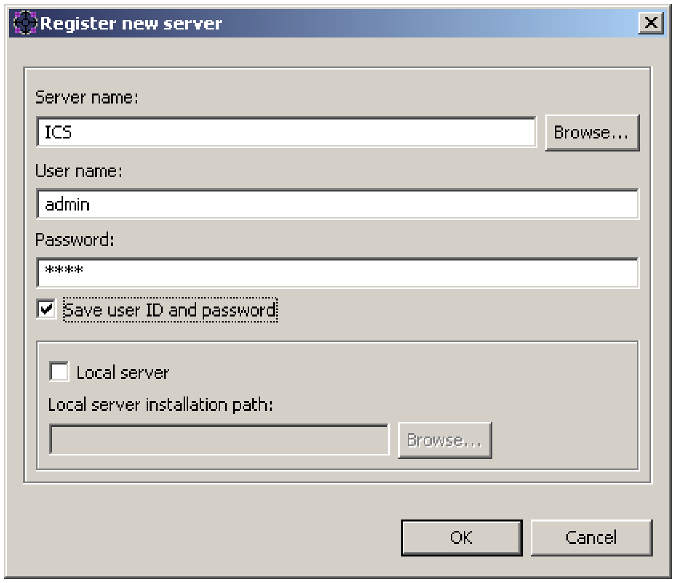
Figure 4-54 Provide information about the ORB

3. Because the System Manager will communicate with the InterChange Server through the ORB, it is important to configure the System Manager startup to point to the ORB host machine where the InterChange Server is registered. Locate the `CWSharedEnv.bat` file and modify the ORB_HOST variable to be the host name of the system where the ORB is implemented. (See Example 4-1 on page 175)

Example 4-1 Modifying CWSharedEnv.bat

```
REM Licensed Materials - Property of IBM
REM 5724-C10, 5724-E30, 5724-I78
REM (C) Copyright IBM Corporation 1997, 2004. All Rights Reserved
REM US Government Users Restricted Rights- Use, duplication or disclosure
REM restricted by GSA ADP Schedule Contract with IBM Corp.

goto setenv

:setenv

set ORB_PORT=14500
set ORB_HOST=wbiics
set ORB_PROPERTY=-DORBNamingProvider=CosNaming
-Dorg.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
-Dorg.omg.CORBA.ORBInitialPort=%ORB_PORT%
-Dorg.omg.CORBA.ORBInitialHost=%ORB_HOST% -Dcom.ibm.CORBA.Debug.Output=nul

set JRE_HOME="%CROSSWORLDS%"\jre
set JRE_LIB=%JRE_HOME%\lib
set JRE_BIN=%JRE_HOME%\bin
set CWJAVA="%CROSSWORLDS%\jre\bin\java"
set CWJAVAW="%CROSSWORLDS%\jre\bin\javaw"
REM set jre part of java.ext.dirs
set JRE_EXT_DIRS=%JRE_LIB%\ext

set PATH=%JRE_BIN%;"%CROSSWORLDS%"\bin;%PATH%
```

4. When the installation is finished, start System Manager and switch to the System Manager perspective. Register the existing InterChange Server `wbiics` within this System Manager the same way as in the previous chapter. See Figure 4-55 on page 176.



Server name: ICS

User name: admin

Password: ****

[✓] Save user ID and password

[ ] Local server

Local server installation path:

Figure 4-55 Register existing server

5. The System Manager connects to the InterChange Server and retrieves information about deployed components from its repository. See Figure 4-56 on page 177. We can now use this instance of the System Manager to inspect runtime statistics, to administer role based access, configure data security, to start and stop components, and to deploy new components.

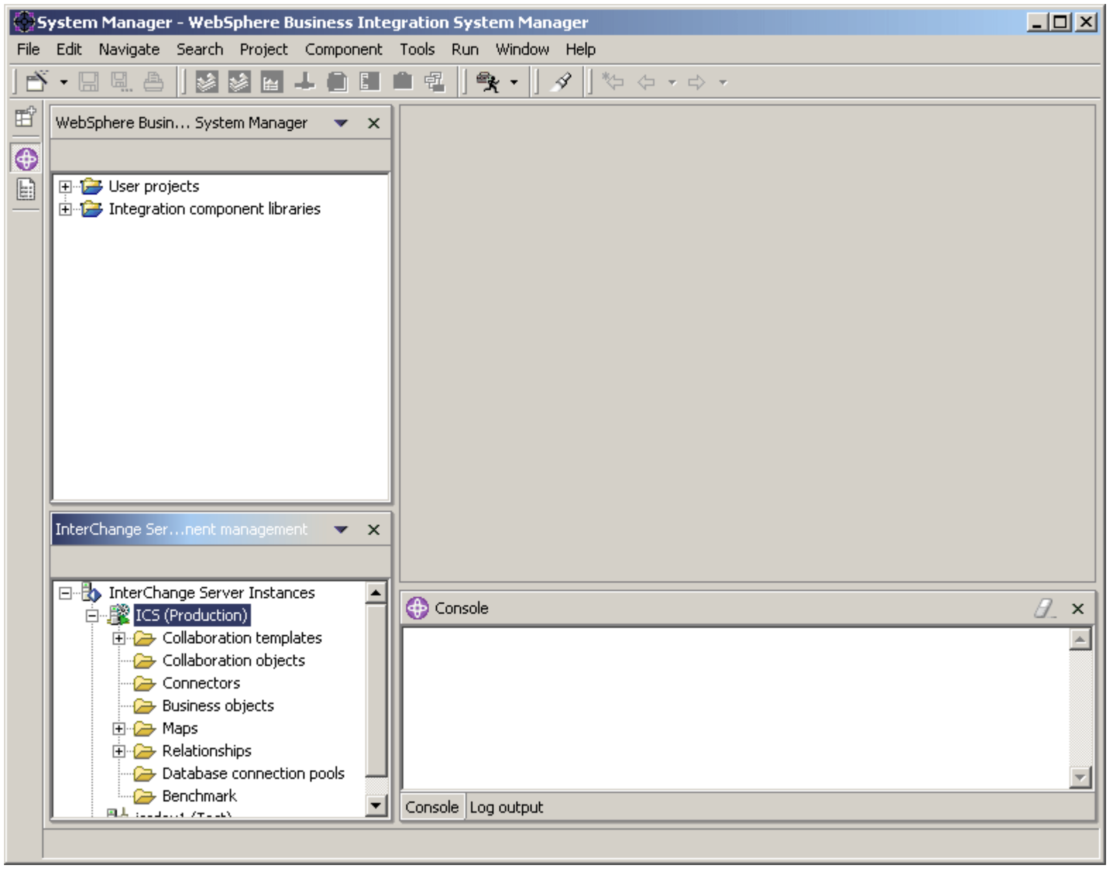

System Manager - WebSphere Business Integration System Manager
File Edit Navigate Search Project Component Tools Run Window Help
WebSphere Busin... System Manager
User projects
Integration component libraries
InterChange Ser...nent management
InterChange Server Instances
ICS (Production)
Collaboration templates
Collaboration objects
Connectors
Business objects
Maps
Relationships
Database connection pools
Benchmark
Console
Console
Log output

Figure 4-56 System Manager connected to remote InterChange Server

Implementing the System Monitor

The *System Monitor* is a Web-based application that enables an InterChange Server administrator to perform operational tasks, interpret historical data, and analyze real-time server statistics. Similar information and options are available as part of the Component Management View in System Manager. However, the System Monitor Web application has the added advantage that the InterChange Server administrator does not have to have a machine with System Manager installed.

Implementing System Monitor on a machine that has WebSphere Application Server and InterChange Server installed is relatively simple. When WebSphere Application Server is installed, the monitoring application is installed automatically during the installation of the InterChange Server.

However, in the environment that was created for this redbook, this was not the case. We had a dedicated system for use by WebSphere Application Server and a dedicated system for use by the InterChange Server.

What must be installed on WebSphere Application Server to deploy the Web-based System Monitor? The installation options for InterChange Server (shown in Figure 4-53 on page 173) do not list the System Monitor application explicitly; it is included in the Administrative Toolset section. Thus, an easy answer to our question is - install this subset on the machine that hosts WebSphere Application Server.

However, this answer might not be acceptable. For example, if WebSphere Application Server runs on a different platform than the InterChange Server, you might not have installation media. If WebSphere Application Server is running on Linux®, it might not even be possible to install any component of the InterChange Server.

For those situations in which it is not possible to use the automated installation program, you can use the following steps.

1. Copy the installation folder **WBSM** in the InterChange Server installation folder from the InterChange Server machine to the WebSphere Application Server machine.

   Follow a similar folder structure on the WebSphere Application Server machine where the WBSM folder is copied in a WebSphereICS folder on the WebSphere Application Server machine.

2. To install the Web-based application, copy two command files, CWSharedEnv.bat and CWDashboard.bat from the bin directory of InterChange Server. Note that these command files do little more than setting the right environment variables and invoking the WebSphere Application Server utility wsadmin. The command file CWSharedEnv.bat is called by CWDashboard.bat.

3. To ensure that the command file CWSharedEnv.bat creates an environment that is correct on the WebSphere Application Server machine, update ORB_HOST to point to the host name of the InterChange Server instead of local host.

4. While WebSphere Application Server is not running, execute the following command:

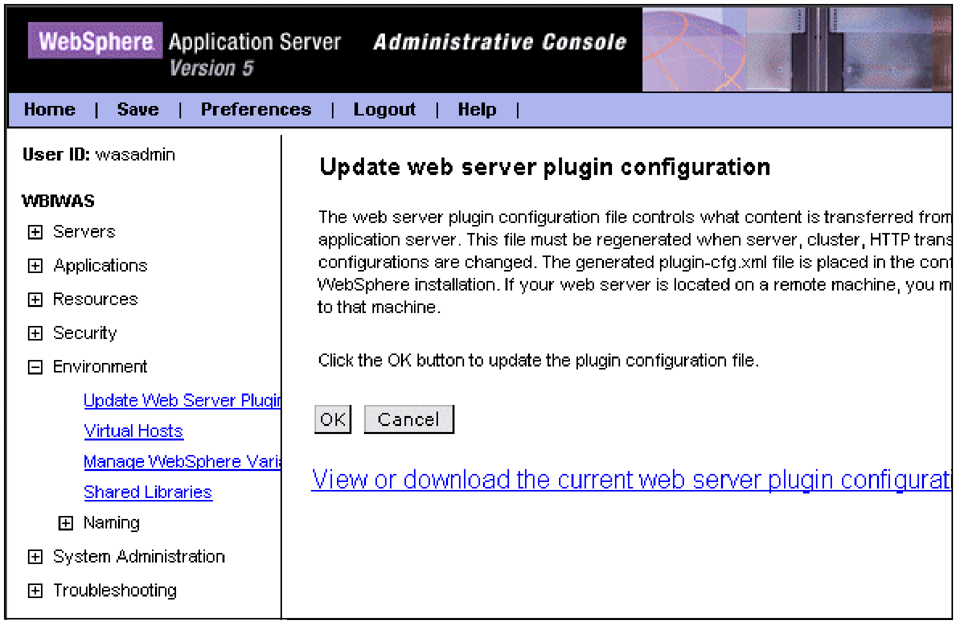
   ```
   D:\WebSphereICS\bin>cwdashboard.bat d:\websphere\appserver
   wbiwas.itso.ral.ibm.com D:\WebSphereICS D:\SQLLIB\java
   ```

   The first parameter tells where WebSphere Application Server is installed. The second parameter is the fully-qualified host name. `D:\WebSphereICS` is the folder on the WebSphere Application Server machine where the WBSM folder was copied. The last parameter points to the folder where the JDBC driver for DB2 is stored.

5. When this command completes successfully, you can start the administrative server with the command:

```
startServer server1
```

6. Open a browser and point it to the administrative console of WebSphere Application Server. Update the Web server plugin configuration file. In the menu in the left-hand pane, select **Environment** → **Update Web Server Plugin**. Click **OK** to confirm your request (Figure 4-57).



Update web server plugin configuration

The web server plugin configuration file controls what content is transferred from application server. This file must be regenerated when server, cluster, HTTP trans configurations are changed. The generated plugin-cfg.xml file is placed in the con WebSphere installation. If your web server is located on a remote machine, you m to that machine.

Click the OK button to update the plugin configuration file.

View or download the current web server plugin configurat

Figure 4-57 Update the web server plugin configuration file

7. Start the application server that hosts the System Monitor Web application:

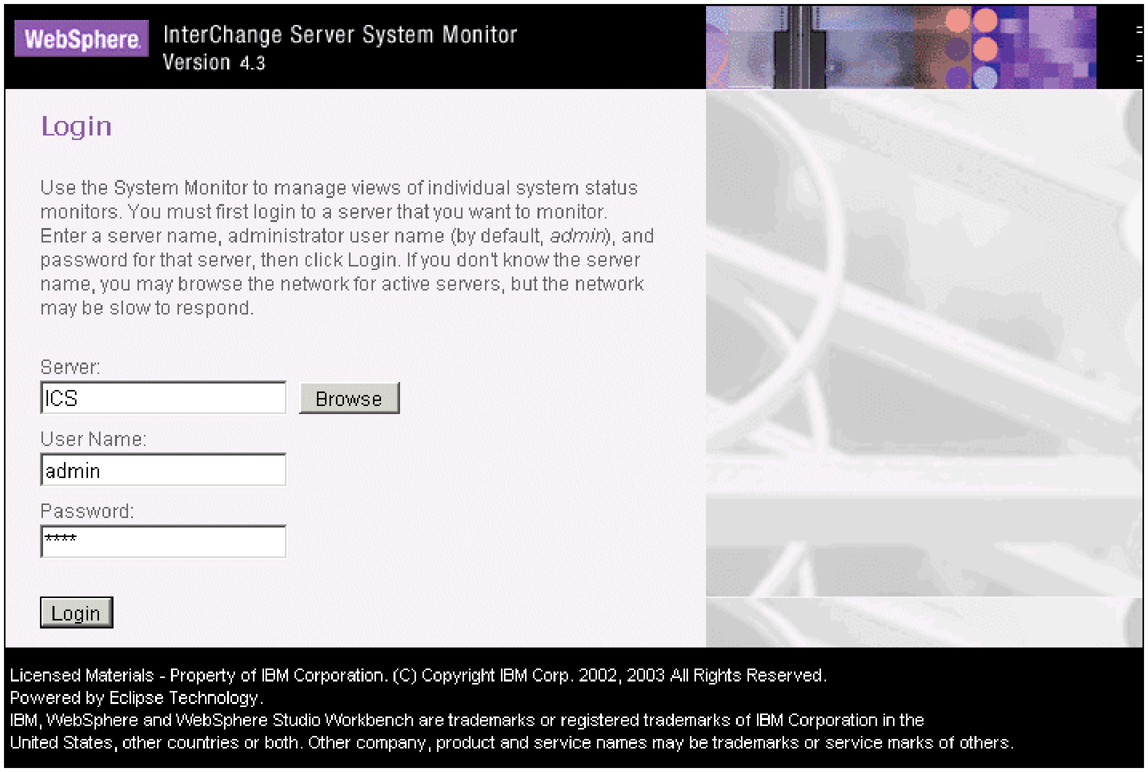
```
startServer ICSMonitor
```

Note that the `ICSMonitor` parameter is case-sensitive.

The browser that is used to work with the System Monitor should have the SVG Viewer plugin installed. This plugin is used by certain features of the System Monitor to visualize graphical data. This plugin is freely available from the Adobe Web site at:

`http://www.adobe.com/svg/viewer/install/main.html`

8. Point your browser to `http://`*`wbiwas`*`/ICSMonitor` (where *`wbiwas`* is the host name of the WebSphere Application Server). A logon page opens, as shown in Figure 4-58.



Login

Use the System Monitor to manage views of individual system status monitors. You must first login to a server that you want to monitor. Enter a server name, administrator user name (by default, *admin*), and password for that server, then click Login. If you don't know the server name, you may browse the network for active servers, but the network may be slow to respond.

Server: ICS

User Name: admin

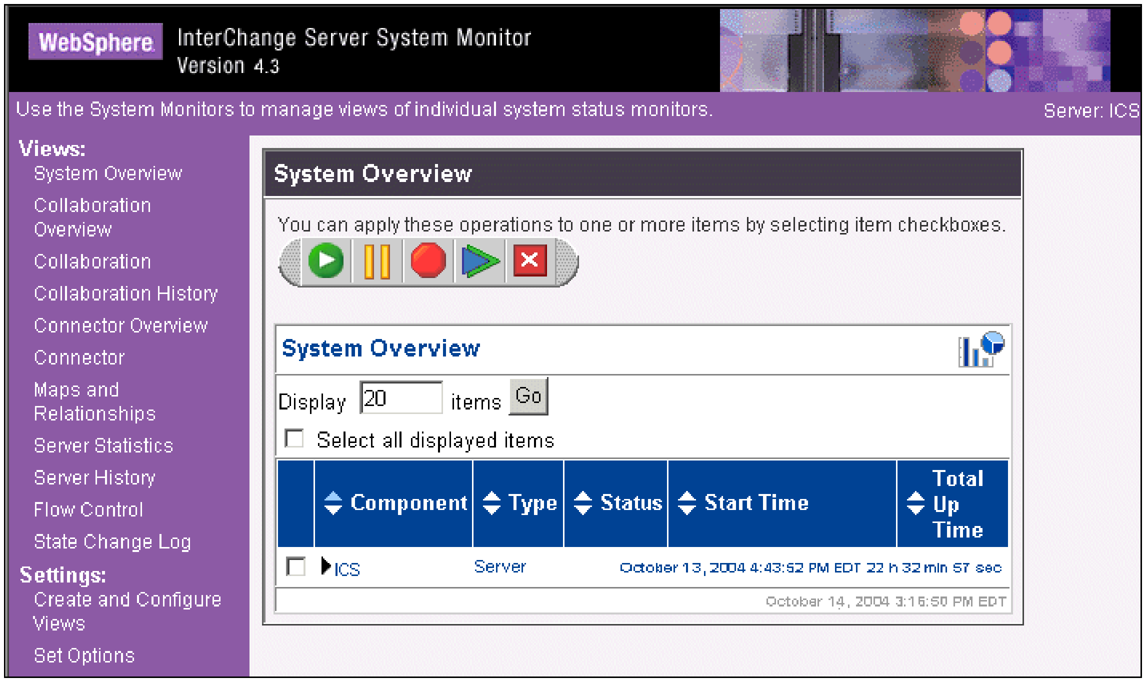
Password: ****

Licensed Materials - Property of IBM Corporation. (C) Copyright IBM Corp. 2002, 2003 All Rights Reserved.
Powered by Eclipse Technology.
IBM, WebSphere and WebSphere Studio Workbench are trademarks or registered trademarks of IBM Corporation in the United States, other countries or both. Other company, product and service names may be trademarks or service marks of others.

Figure 4-58 Login page of System Monitor

9. Enter the name of the InterChange Server (`ICS` in our example), the user ID (`admin`), and its password (the word `null`, by default).

When the login is successful, the System Overview page is shown. (Figure 4-59 on page 181)


WebSphere InterChange Server System Monitor Version 4.3

Use the System Monitors to manage views of individual system status monitors. Server: ICS

Views:
- System Overview
- Collaboration Overview
- Collaboration
- Collaboration History
- Connector Overview
- Connector
- Maps and Relationships
- Server Statistics
- Server History
- Flow Control
- State Change Log

Settings:
- Create and Configure Views
- Set Options

System Overview

You can apply these operations to one or more items by selecting item checkboxes.

System Overview

Display 20 items Go

☐ Select all displayed items

	Component	Type	Status	Start Time	Total Up Time
☐	ICS	Server		October 13, 2004 4:43:52 PM EDT	22 h 32 min 57 sec

October 14, 2004 3:16:50 PM EDT

Figure 4-59 System Overview page

10. Select a menu option that uses the SVG Viewer plugin, such as **Server Statistics**. If you have to install this plugin, accept its license when it is invoked for the first time by the browser.

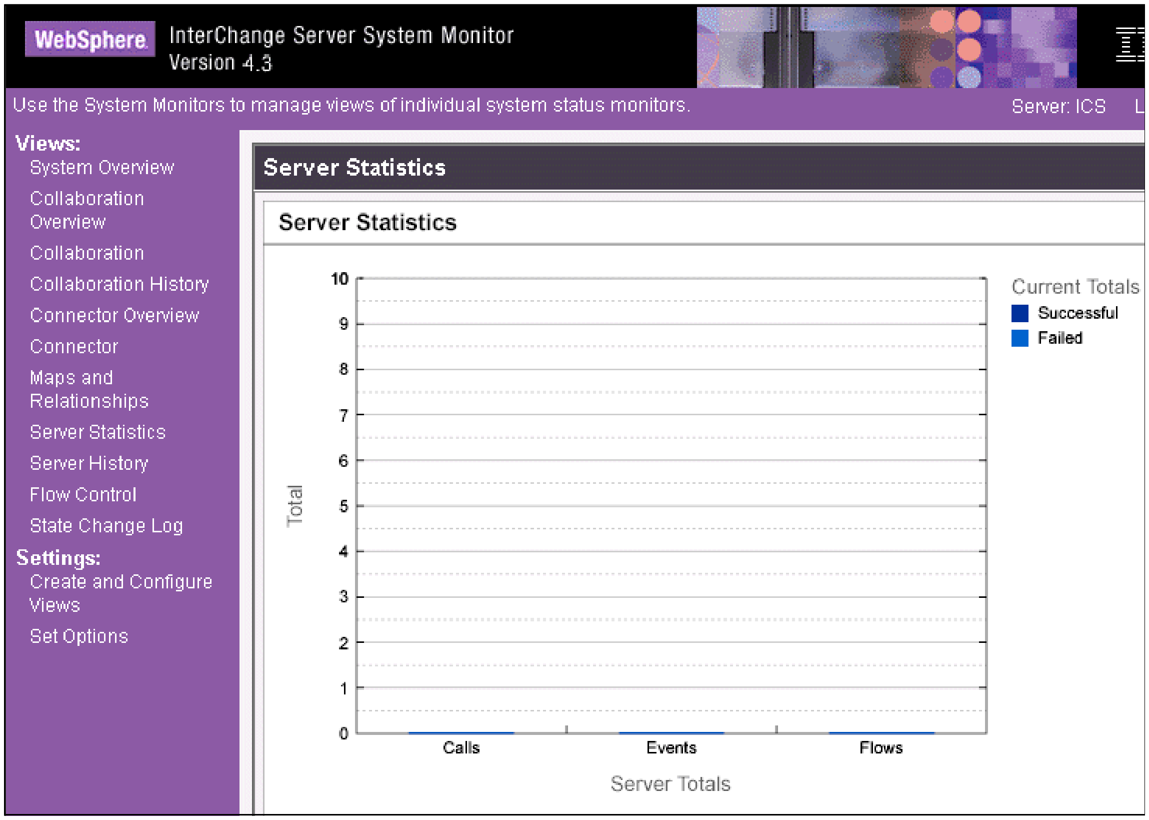


Figure 4-60 Server statistics

Figure 4-60 shows statistics only about Calls, Events, and Flows. However, this page also contains statistics about database connections and usage and MQ queue depth.

4.3.3 WebSphere Message Broker management client

As a system administrator for a broker domain, you use the Message Brokers Toolkit (Broker Toolkit) to perform the following tasks:

- Create and manage the broker domain:
  - Add and remove brokers.
  - Add and remove execution groups.
- Assemble solution components into broker archives and deploy them to brokers.
- Manage topics and subscriptions.

- Start and stop deployed components.
- Start and stop broker and configuration manager.
- Initiate runtime problem determination:
  - Trace a solution
  - Inspect the Windows Event Viewer, which is used by the broker and by WebSphere MQ to report errors

These tasks can be performed by either the Broker Toolkit or standard Windows utilities.

Setting up the Broker Toolkit for management purposes

The Broker Toolkit uses WebSphere MQ communication techniques to interact with the configuration manager on the remote machine. The configuration manager also distributes requests (a deployment, for example) to the target broker through WebSphere MQ. As a result, the management client must have at least WebSphere MQ Client software. This software is freely available from the following Web site:

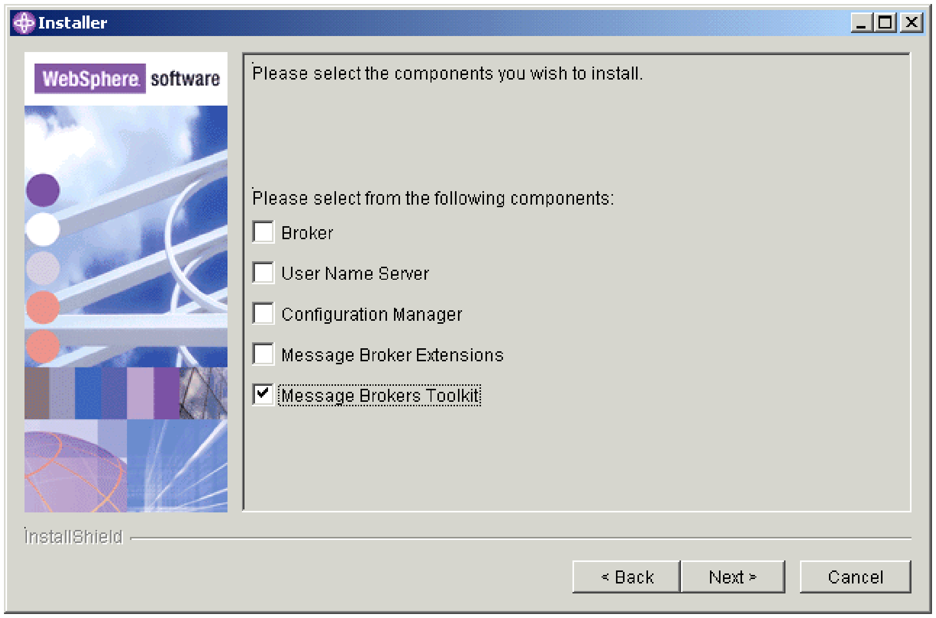
`http://www-306.ibm.com/software/integration/support/supportpacs/`

The Broker Toolkit also requires that Microsoft Data Access Components (MDAC) is available on the machine. MDAC is distributed with the WebSphere Business Integration Message Broker product.

From a software perspective, there is not much difference between a development client and a management client for WebSphere Business Integration Message Broker. However, while a developer does not have to interact with the configuration manager to develop components, an administrator must interact with the configuration manager to do her job. A developer interacts with the configuration manager when testing solutions by using the unit test component. The developer can control the use of the services that are offered by the configuration manager by making user IDs members of certain groups. The configuration manager checks the user ID that is logged on to the client side. This user ID is contained in the Broker Toolkit WebSphere MQ message sent to the configuration manager.

Besides making the user ID of the broker administrator a member of the WebSphere Business Integration Message Broker user groups, you should consider adding WebSphere MQ authorizations to ensure that the WebSphere MQ client can connect to the configuration manager’s queue manager, and that the WebSphere MQ client can read and write messages to the required queues.

To install the Broker Toolkit, go through the same steps as when the full product was installed. Select a **Custom** setup and in the next window select only the **Message Brokers Toolkit** component, as shown in Figure 4-61.



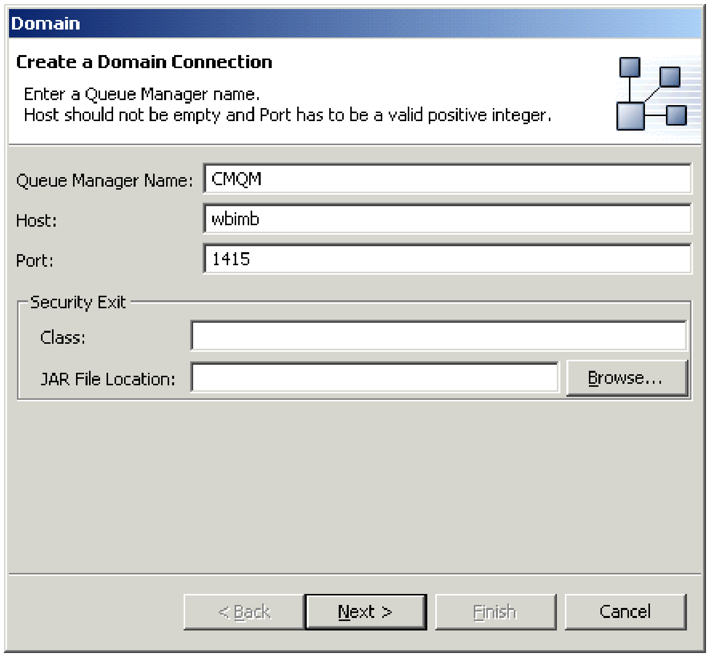
Please select the components you wish to install.

Please select from the following components:

- [ ] Broker
- [ ] User Name Server
- [ ] Configuration Manager
- [ ] Message Broker Extensions
- [✓] Message Brokers Toolkit

Figure 4-61 Install Broker Toolkit component only

When the installation is complete, create a connection file that contains information about how to interact with the configuration manager. Start the toolkit and switch to the Broker Administration perspective. Locate the Domains view and select **New** → **Domain**. This starts the wizard to create a connection file. In the first step, enter the name of the queue manager that is used by the configuration manager, the host name, and port number. Click **Next**. (Figure 4-62 on page 185)



Create a Domain Connection

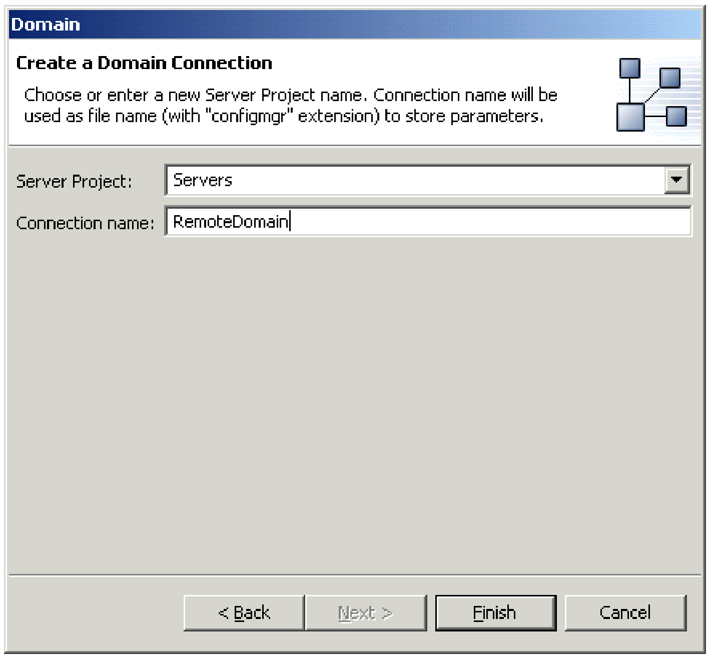
Enter a Queue Manager name.
Host should not be empty and Port has to be a valid positive integer.

Queue Manager Name: CMQM
Host: wbimb
Port: 1415

Security Exit
Class:
JAR File Location:

Figure 4-62 Provide connection information

In the next step, enter a local folder as the name of the server project and a name for the connection file. This file holds the information entered in the previous step. Click **Finish**. (Figure 4-63 on page 186)


Domain

Create a Domain Connection

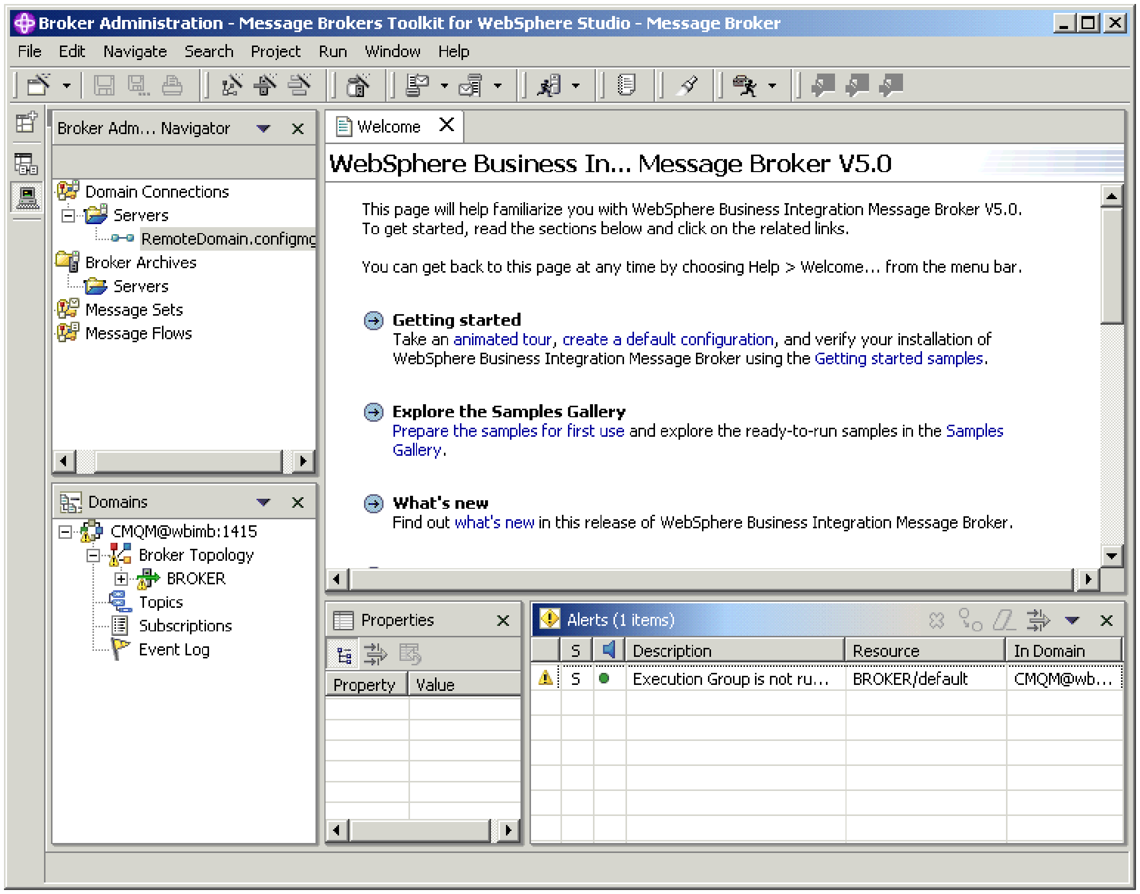
Choose or enter a new Server Project name. Connection name will be used as file name (with "configmgr" extension) to store parameters.

Server Project: Servers

Connection name: RemoteDomain

Figure 4-63 Provide Server Project folder and Connection name file

The Broker Toolkit accesses the configuration manager and retrieves the broker topology from it. The returned information is used to populate the Domains view. Contrasting Figure 4-64 on page 187 with Figure 3-44 on page 98, you see that this time the view is populated. There is no need to register the broker again.



WebSphere Business In... Message Broker V5.0

This page will help familiarize you with WebSphere Business Integration Message Broker V5.0. To get started, read the sections below and click on the related links.

You can get back to this page at any time by choosing Help > Welcome... from the menu bar.

Getting started
Take an animated tour, create a default configuration, and verify your installation of WebSphere Business Integration Message Broker using the Getting started samples.

Explore the Samples Gallery
Prepare the samples for first use and explore the ready-to-run samples in the Samples Gallery.

What's new
Find out what's new in this release of WebSphere Business Integration Message Broker.

	S		Description	Resource	In Domain
⚠	S	●	Execution Group is not ru...	BROKER/default	CMQM@wb...

Figure 4-64 Management client connected to configuration manager

We use this management client in a later chapter when we deploy our first solution to the broker.

Using Windows system tools

While the Broker Toolkit can perform many management tasks for the brokers that are deployed in your domain, not everything is possible through this interface. To stop and start the broker or the configuration manager, you must have access to operating system tools.

Open the Computer Management application by right-clicking **My Computer** icon on the Windows Desktop and selecting **Manage**. Right-click **Computer Management** and select **Connect to another computer** (Figure 4-65 on page 188). Depending how your Windows security environment is configured, you might be prompted to provide a user ID and password.

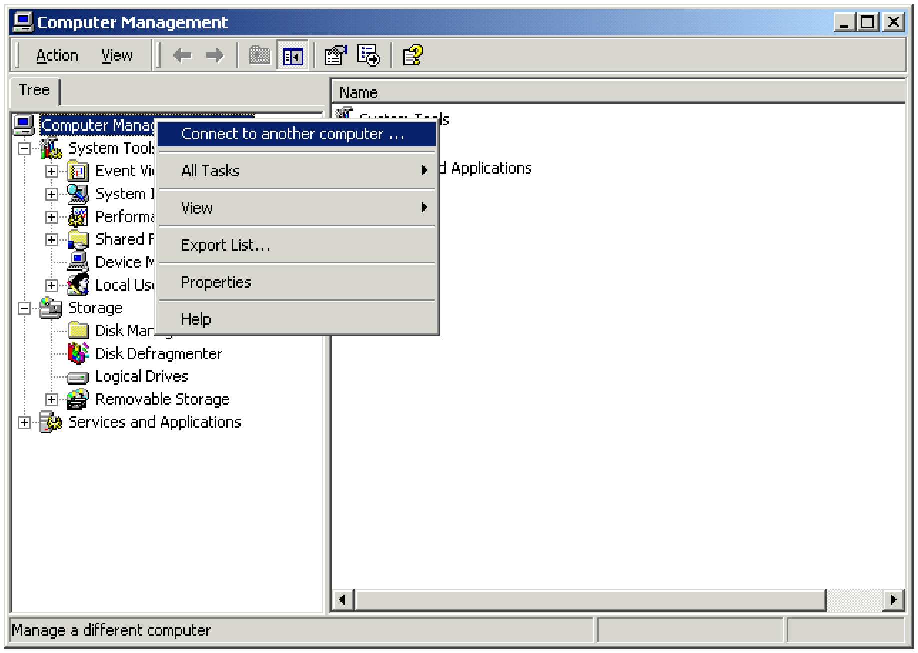
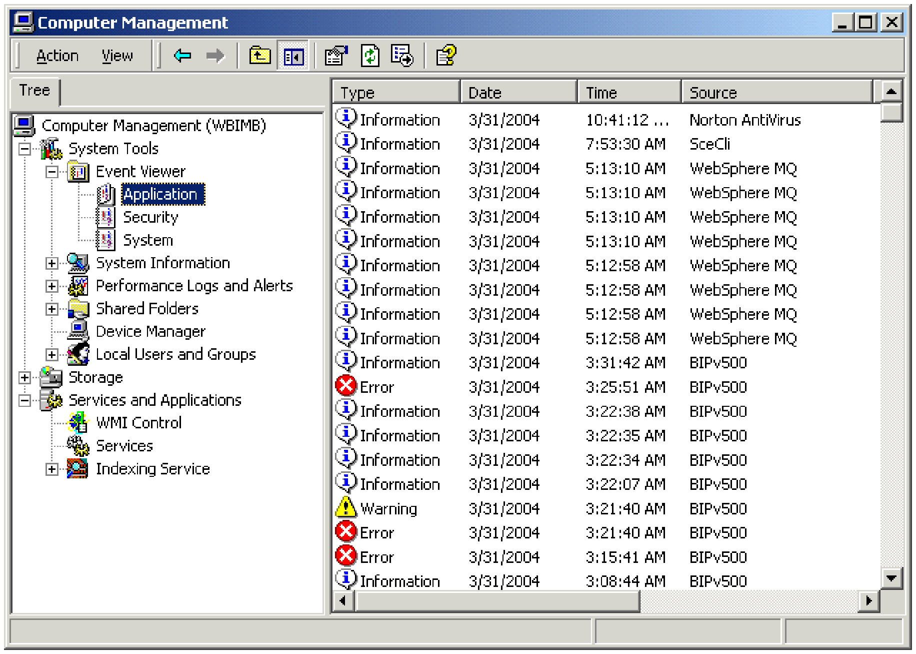


Figure 4-65 Connect to another computer

When connected, you can open the Event Viewer on the remote machine (Figure 4-66 on page 189) to inspect runtime messages from WebSphere Business Integration Message Broker. You also have access to the Windows Services through this interface. This enables you to stop and start the WebSphere Business Integration Message Broker components.



Type	Date	Time	Source
Information	3/31/2004	10:41:12 ...	Norton AntiVirus
Information	3/31/2004	7:53:30 AM	SceCli
Information	3/31/2004	5:13:10 AM	WebSphere MQ
Information	3/31/2004	5:13:10 AM	WebSphere MQ
Information	3/31/2004	5:13:10 AM	WebSphere MQ
Information	3/31/2004	5:13:10 AM	WebSphere MQ
Information	3/31/2004	5:12:58 AM	WebSphere MQ
Information	3/31/2004	5:12:58 AM	WebSphere MQ
Information	3/31/2004	5:12:58 AM	WebSphere MQ
Information	3/31/2004	5:12:58 AM	WebSphere MQ
Information	3/31/2004	3:31:42 AM	BIPv500
Error	3/31/2004	3:25:51 AM	BIPv500
Information	3/31/2004	3:22:38 AM	BIPv500
Information	3/31/2004	3:22:35 AM	BIPv500
Information	3/31/2004	3:22:34 AM	BIPv500
Information	3/31/2004	3:22:07 AM	BIPv500
Warning	3/31/2004	3:21:40 AM	BIPv500
Error	3/31/2004	3:21:40 AM	BIPv500
Error	3/31/2004	3:15:41 AM	BIPv500
Information	3/31/2004	3:08:44 AM	BIPv500

Figure 4-66 Event Viewer of remote broker machine

4.4 Summary

This chapter provided information about creating a runtime client environment, a development client environment, and a management client environment.

The full power of having dedicated machines for each task or role becomes apparent when we build, test, deploy, and manage a business integration solution in the second part of this book.

Part 2

Implementing business integration solution components

5

Application scenario and solution design

This chapter describes a business scenario and solution design that is deployed within the business integration infrastructure discussed in Part 1, “Implementing a BI solution framework” on page 1.

5.1 Business scenario

Within the business integration infrastructure that is described in the first part of this book, we implement an integration solution. The scenario that we use is a pared-down version of a real-life scenario that is described in another redbook, *Business Integration Management using WebSphere BI Modeler and Monitor A Real World Case Study*, SG24-7024. You can find the book at this Web site:

`http://www.redbooks.ibm.com/abstracts/sg247024.html?Open`

That redbook describes full details of the business process and uses WebSphere Business Integration Modeler and Monitor to model, simulate, and measure the business process. The business process that is described in that redbook contains many human interaction points and a few application integration points. In our book, we have simplified the process model to demonstrate two integration points and a limited amount of human interaction.

The following drawings illustrate, from a high-level to the most detailed view, all the business process elements we are working on for this case study. We start first with the high-level view of the main elements in Figure 5-1 on page 195.

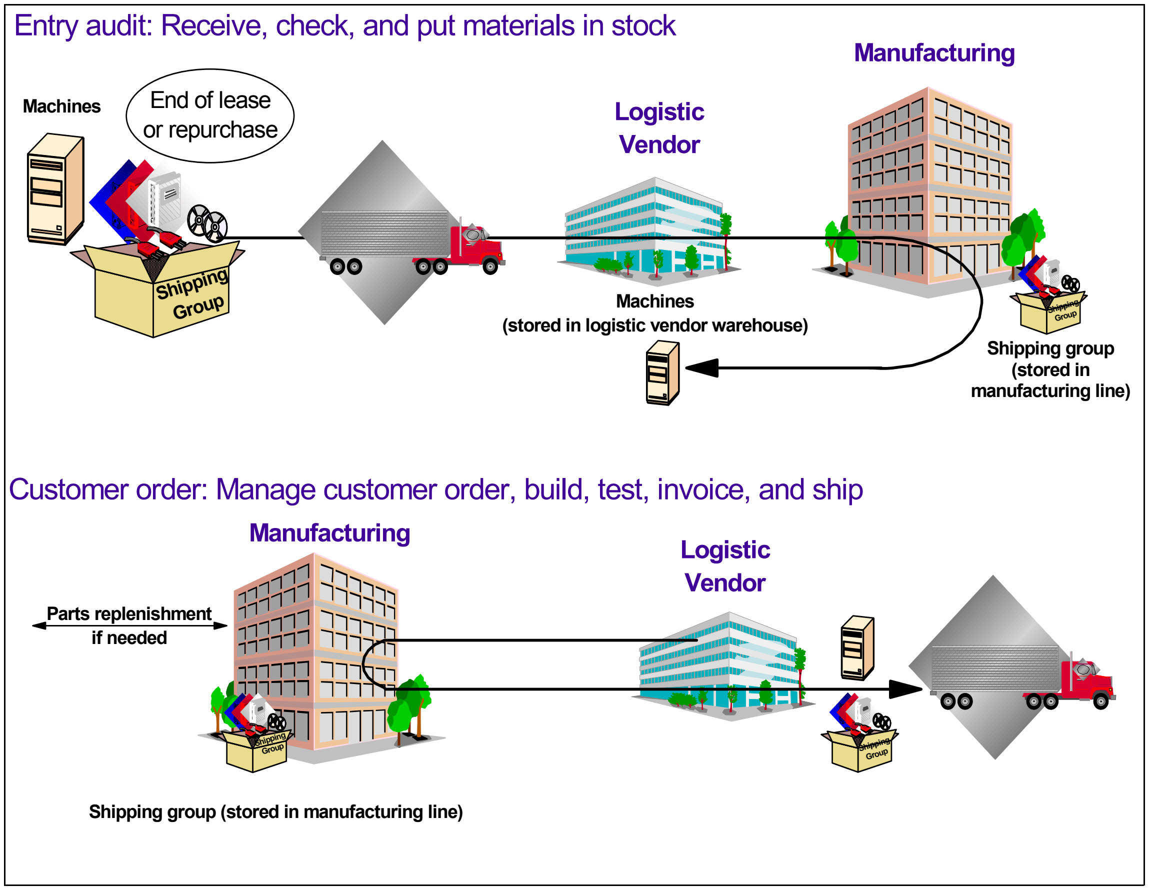

Entry audit: Receive, check, and put materials in stock
Machines
End of lease or repurchase
Shipping Group
Logistic Vendor
Manufacturing
Machines (stored in logistic vendor warehouse)
Shipping group (stored in manufacturing line)
Customer order: Manage customer order, build, test, invoice, and ship
Manufacturing
Logistic Vendor
Parts replenishment if needed
Shipping group (stored in manufacturing line)

Figure 5-1 High-level view of the real case business process

Figure 5-2 on page 196 shows the business process as a flow of main functions. We have numbered the processes and some of the functions, for reference.

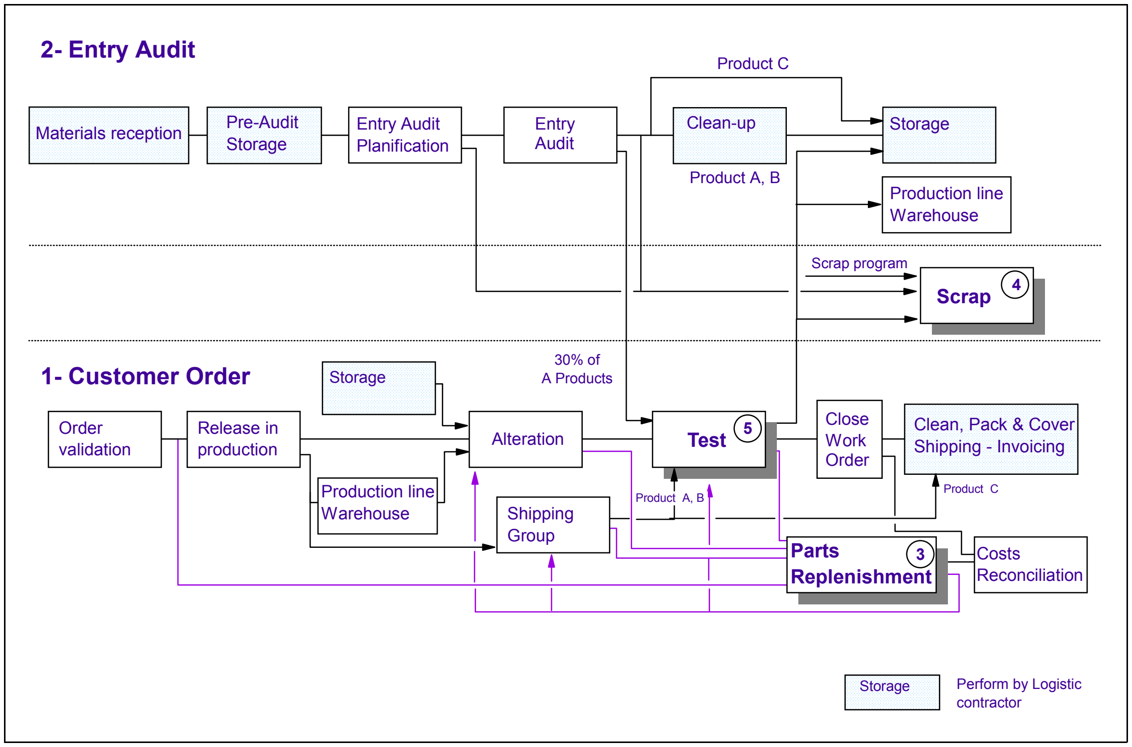


Figure 5-2 Chain of functions overview

5.1.1 Customer order process

The customer order business process starts when an order arrives from an SAP system. This is the business process trigger. Orders are validated in terms of machine configuration, and in-stock supply is checked before releasing the order to production. If the order is valid and supply is available in used stock, the production line receives machines and feature codes (memory, power supplies, channel cards, and so forth) and configures the order. Alteration performs hardware operations by adding or removing feature codes. Feature codes that have been removed are stored in the production line warehouse.

In parallel, the production line warehouse produces the shipping group, which is dependent on the order configuration. The machine is tested, and the order is closed and sent out to the logistics vendor, which performs finalization (clean, pack, and cover) if it is a product C, and, for all product types, sends it to shipping and invoicing activities. The business process ends when the SAP sales system is updated after shipping and invoicing.

Orders can be made on feature codes only when the customer requires a configuration upgrade of a machine that he already owns.

If the used inventory does not have all of the parts to build the machine, the alteration and shipping group detects missing parts, or if a test detects defective parts, parts can be replenished from an external source according to price and replenishment time rules. On parts arrival, the order is completed and can proceed to the end of the business process.

5.1.2 Entry audit process

Entry audit, the first subprocess in this business process, consists of storing materials in warehouses and recording them in the inventory system.

Materials are received in the logistic vendor's warehouse and stored until manufacturing calls for the machine to check the configuration. At this time, if disposition instructions are available for the machine, meaning that only some feature codes are interesting to keep, the machine is delivered to test. The feature codes with disposition instructions are tested, removed from the machine, and stored in the production line warehouse. This is the case for 30% of product A machines. This enables the customer order process to respond quickly to an order on feature codes only, especially if they are already tested. In this case, all that is needed is to pack the tested feature codes, as they are ready to be shipped without having any other operations performed. The rest of the machine, and any defective parts that were detected during test operations, are sent directly to scrap.

When materials' configuration has been checked, they are stored in two different locations:

- In the logistic vendor warehouse for machines (large-dimension)
- In the production line warehouse for feature codes and shipping group

5.1.3 Subprocesses

Three subprocesses are identified in the business process. They are labeled 3 (parts replenishment), 4 (scrap), and 5 (test) in Figure 5-2 on page 196.

Parts replenishment

This is critical to manufacturing fulfillment of customer orders. It must be very responsive (taking less than 0.5 day) to be able to assess order feasibility. Parts can be replenished from up to four different locations, which can be internal to the plant or even in other countries. Although shipment tracking of these parts in completing customer orders is stopped while waiting for them, it is key with respect to the manufacturing cycle time-committed target. This is where the business integration that is associated with workflow management was perceived to be necessary in order to monitor and closely manage the completeness of the business process.

Scrap

This program is common to the customer order process and the entry audit process. It is triggered by the entry audit process after advanced testing of feature codes and by the customer order process when parts are found to be defective during test operations. The customer order process also has its scrap program based on yearly financial decisions and inventory analysis. Scrap operations are approved by different people in different organizations. Scrap operations is an excellent prospect for part of an implementation of a business integration management system with a workflow to track the scrap progress and keep archives for audit purposes.

Test

Test is common to the customer order process and the entry audit process. Capacity conflicts can occur at this step, as seen during the reengineering study. A close monitoring of this process helps production-line management to deal with solving capacity issues.

5.2 What we implemented

Instead of implementing the whole business process in this book, we create a reduced business process designed to demonstrate the use of WebSphere Business Integration Server in three situations:

- Human workflow integration
- Process integration
- Message-based integration

If you are is interested in modeling, simulating, and business monitoring of a business process with real-life complexity, consult the redbook *Business Integration Management using WebSphere BI Modeler and Monitor A Real World Case Study*, SG24-7024.

Our implemented the business process involves the creation of a new order, the saving of that order information into a master order database, aggregating part of the information from multiple suppliers, manually approving the final order, and acting upon that final approval to prepare for billing and shipment.

Figure 5-3 on page 200 shows the simplified model as it is implemented in WebSphere MQ Workflow. The first integration point involves a 'hand-shake' between WebSphere MQ Workflow and InterChange Server. This integration point occurs at the Create Sales Order node in the process model. Through the use of WebSphere MQ Workflow Web Client, a new process instance is created (the actual sales order). Included in the new process instance are the following details about the sales order:

- OrderDate
- CustomerNumber
- ExpectedDeliveryDate
- NumberOfParts
- OrderDetail (PartNumber, Quantity)

After this information is entered, WebSphere MQ Workflow generates and sends an XML message to an input queue (MQWF.INPUT) residing on the InterChange Server queue manager. From there, the Interchange Server retrieves the message and uses the Sales Order Collaboration to control the inserting of the order and order line items data into a database. A new order number is automatically created in the database and returned to the collaboration, which in turn sends the information back to the controlling workflow through the output queue (FMC.FMCGRP.EXE.XML) where it is retrieved with the MQ Workflow Server.

The second integration point involves a *hand-shake,* the exchange of credentials, between WebSphere MQ Workflow and Message Broker. This integration point occurs at the Order Parts node in the process model. Without any manual intervention, the Message Broker retrieves the XML message sent from the Workflow server (ORDER.INPUT). The WebSphere Business Integration Message Broker behaves as a supplier simulator by inserting the following items in the message:

- Supplier
- Order Reference
- Expected Shipment, Reception, and Registration dates
- Case number
- Invoice Price

Next, the data is routed back to the Workflow server through an alias queue (FMC.FMCGRP.EXE.XML). When Workflow acquires the response message from the Message Broker, human intervention is required to approve the order. The order approval is based on whether the unit price per part is too high or is appropriate and, in the process, validating whether or not it is cost effective to fulfill the sales order. Through the Workflow Web Client, you are required to insert a response in the Authorization field of the activity instance (**1** to authorize the transaction) and to fill in the corresponding order status (`Active` or `Canceled`). Based on your response, the Workflow directs the message to the Confirm Order or Delete Sales Order node. Both the Confirm Order activity and the Delete

Sales Order activity send an asynchronous message to the InterChange Server to update the status of the order appropriately.

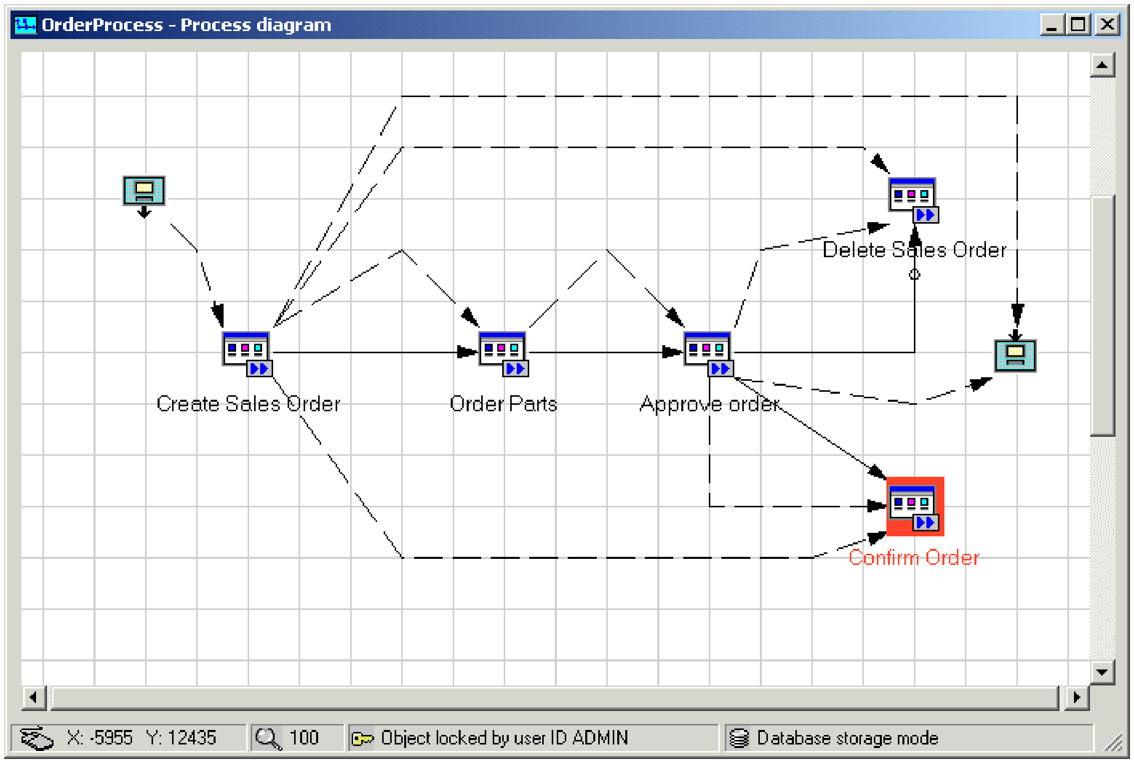


Figure 5-3 Customer order process

6

Implementing a process model in WebSphere MQ Workflow

To accommodate a long-running business process with human interactions, the decision was made to implement the business process in WebSphere MQ Workflow.

This chapter describes how to model, deploy, and test the workflow.

6.1 Overview

To model a business process for use in WebSphere MQ Workflow Runtime server, you can use the modeling tool provided by WebSphere MQ Workflow, called Buildtime, or you can use the WebSphere BI Modeler product. The use of this product and its integration with WebSphere MQ Workflow and other products is demonstrated in the redbook *Business Integration Management using WebSphere BI Modeler and Monitor A Real World Case Study*, SG24-7024. In this book, we use the standard Buildtime tool.

Chapter 4, “Implementing client components” on page 119 described the setup of a development client that included WebSphere MQ Workflow Buildtime. This environment is used in this chapter to implement the process model.

Modeling and validating a business process is often an iterative task. It would be unwise to model the process with all integration aspects from the beginning. A good approach is:

- Implement the business process with all activities, including conditions. All activities contain dummy programs so that a process tester can easily walk through the process model using the Web client.
- Deploy this simplified process model to the runtime server.
- Validate the condition logic using the Web client. Make sure that all paths are tested and that they can be reached.
- Add rules to assign activities to the correct persons or groups. Deploy again. This time, the validation requires the use of multiple sessions where each session is used by a different person representing a different role in the process model.
- Add integration logic. Update those activities that link to external programs or integration platforms, such as WebSphere InterChange Server and WebSphere Business Integration Message Broker.
- Deploy and validate again. This time, the validation will require the assistance of developers who are responsible for the programs or systems that WebSphere MQ Workflow is integrating with.

As such, each iteration adds complexity. If a defect is discovered and it requires a change at an already validated level, the test process should resume at that level again, to make sure that the change did not affect other interfaces.

6.2 Creating the process model

The technique to get from a blank model to a completed model can be different for each user of a modeling tool, in the same way that each developer will have a way to go from class specification to class implementation. The approach we favor is demonstrated below:

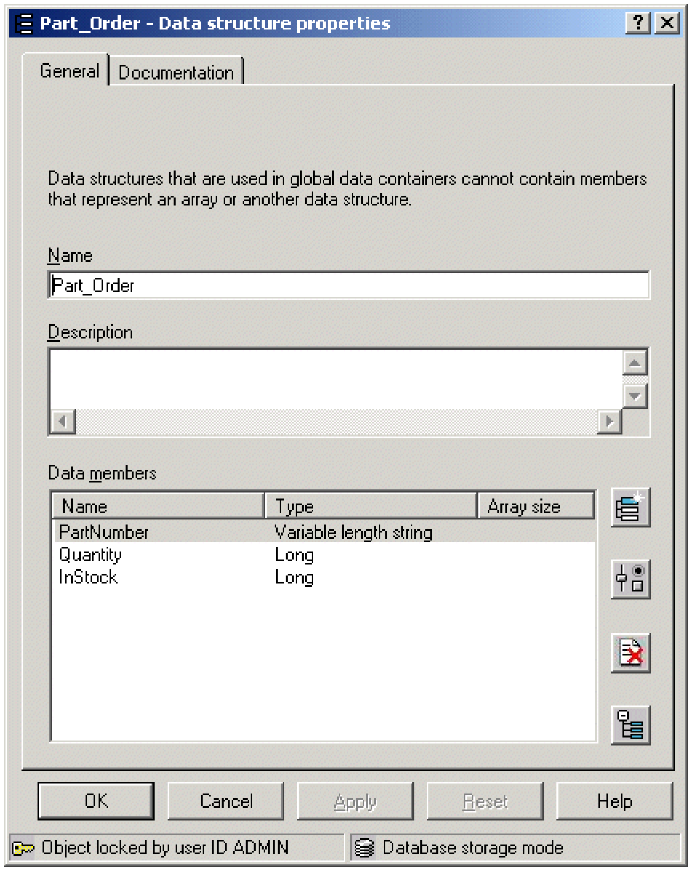
1. Start with creating data structures.
2. Create program objects that implement activities in the process model.
3. Add activities to the process model.
4. Connect the activities to each other with appropriate condition settings.
5. Add data mapping between output and input data structures.

6.2.1 Creating data structures

Defining data structures is performed within the Implementations tab in Buildtime. Buildtime considers two types of implementation objects: programs and data structures.

To create a new data structure, right-click the **Data structures** folder and select **New Data Structure**. This opens an interface on which you can add elements to the data structure. Elements can either be a simple data type or a complex data type, which is one that has been defined previously to Buildtime.

Figure 6-1 shows the elements in the Part_Order data structure.



Data structures that are used in global data containers cannot contain members that represent an array or another data structure.

Name: Part_Order

Description:

Data members

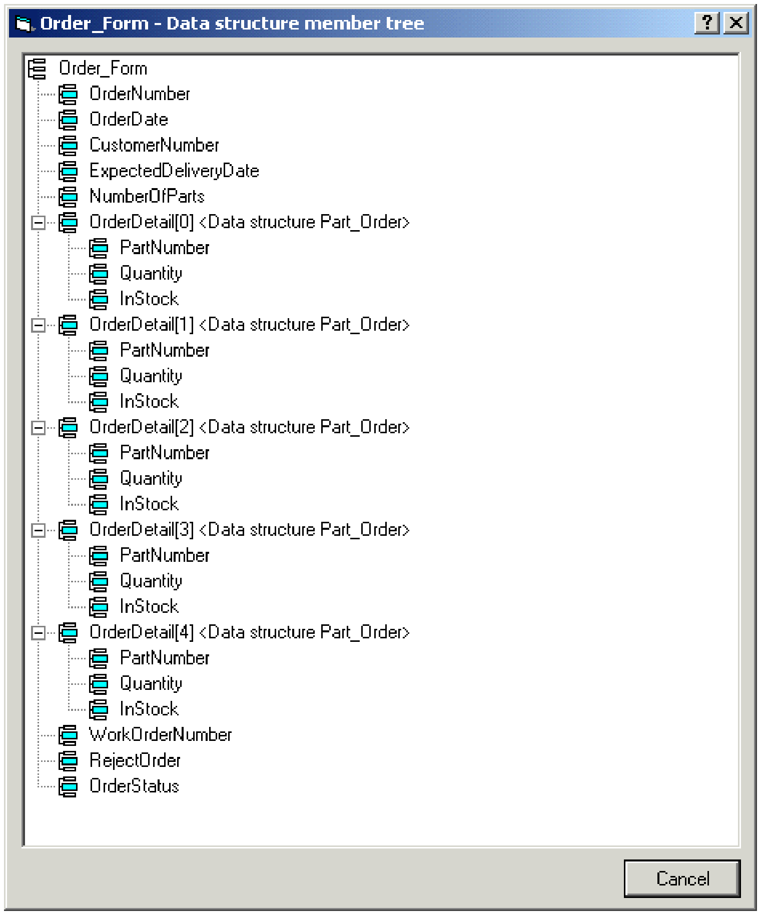
Name	Type	Array size
PartNumber	Variable length string	
Quantity	Long	
InStock	Long	

Object locked by user ID ADMIN

Database storage mode

Figure 6-1 Details of the Part_Order data structure

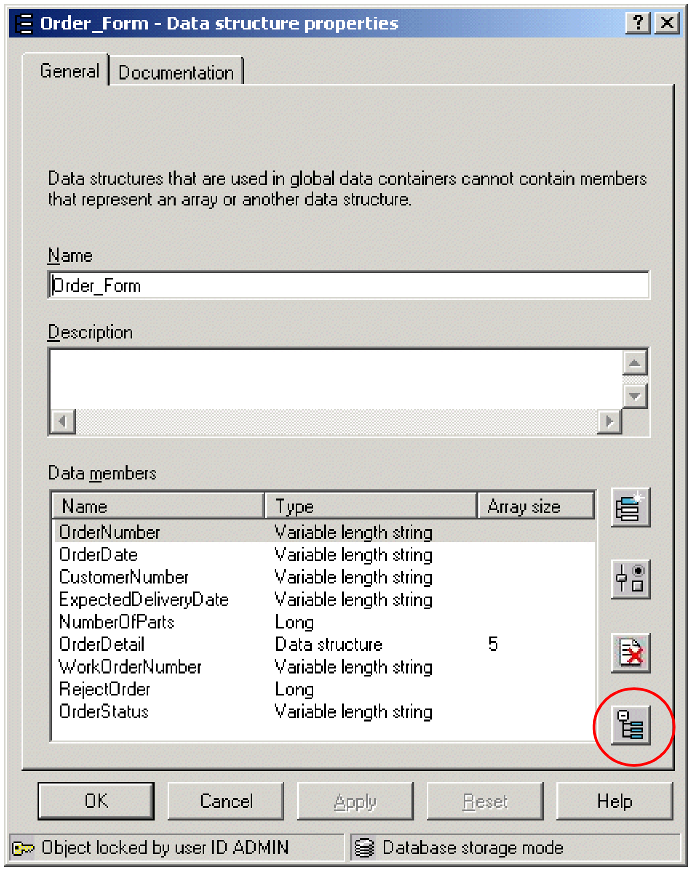
Defining complex data structures, such as the Order_Form structure shown in Figure 6-2, is a bottom-up process. First, define the data structures that are used as types in the overall structure, such as Part_Order. Then you can define the overall structure, Order_Form, that has an element of the Part_Order type.



```
Order_Form
    OrderNumber
    OrderDate
    CustomerNumber
    ExpectedDeliveryDate
    NumberOfParts
    OrderDetail[0] <Data structure Part_Order>
        PartNumber
        Quantity
        InStock
    OrderDetail[1] <Data structure Part_Order>
        PartNumber
        Quantity
        InStock
    OrderDetail[2] <Data structure Part_Order>
        PartNumber
        Quantity
        InStock
    OrderDetail[3] <Data structure Part_Order>
        PartNumber
        Quantity
        InStock
    OrderDetail[4] <Data structure Part_Order>
        PartNumber
        Quantity
        InStock
    WorkOrderNumber
    RejectOrder
    OrderStatus
```

Figure 6-2 Tree view of data structure Order_Form

While Figure 6-2 shows a structural view of the data structure Order_Form, it does not provide details about the data structure of each element. These details are shown in Figure 6-3. The tree view is shown when clicking the button that is circled in the figure.



Data structures that are used in global data containers cannot contain members that represent an array or another data structure.

Name: Order_Form

Description:

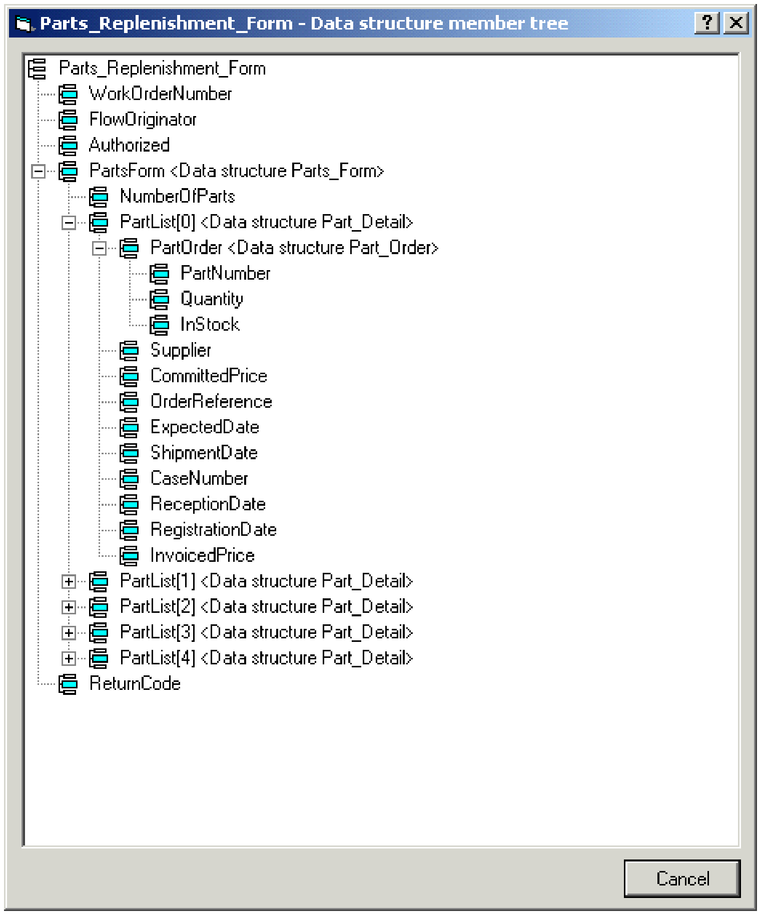
Data members

Name	Type	Array size
OrderNumber	Variable length string	
OrderDate	Variable length string	
CustomerNumber	Variable length string	
ExpectedDeliveryDate	Variable length string	
NumberOfParts	Long	
OrderDetail	Data structure	5
WorkOrderNumber	Variable length string	
RejectOrder	Long	
OrderStatus	Variable length string	

Figure 6-3 Detailed view of data structure Order_Form

The Order_Form structure is used to provide details about the actual order from a customer. To fulfill the order, parts might have to be ordered from suppliers.

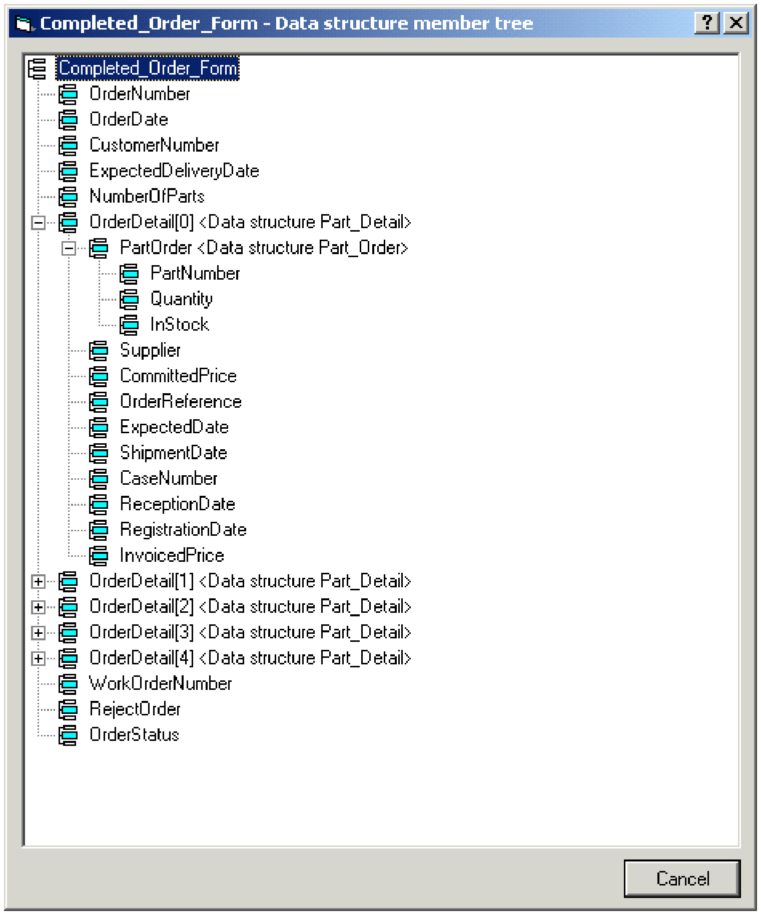
The Parts_Replenishment_Form data structure is used for these purposes. Figure 6-4 shows a tree view of this structure. Note again that another data structure has to be defined first: Part_Detail.


Parts_Replenishment_Form - Data structure member tree

```
Parts_Replenishment_Form
  WorkOrderNumber
  FlowOriginator
  Authorized
  PartsForm <Data structure Parts_Form>
    NumberOfParts
    PartList[0] <Data structure Part_Detail>
      PartOrder <Data structure Part_Order>
        PartNumber
        Quantity
        InStock
      Supplier
      CommittedPrice
      OrderReference
      ExpectedDate
      ShipmentDate
      CaseNumber
      ReceptionDate
      RegistrationDate
      InvoicedPrice
    PartList[1] <Data structure Part_Detail>
    PartList[2] <Data structure Part_Detail>
    PartList[3] <Data structure Part_Detail>
    PartList[4] <Data structure Part_Detail>
  ReturnCode
```

Figure 6-4 Tree view of the data structure Parts_Replenishment_Form

Figure 6-5 shows a third data structure, Completed_Order_Form, which combines elements from the preceding two data structures.


Completed_Order_Form - Data structure member tree

```
Completed_Order_Form
    OrderNumber
    OrderDate
    CustomerNumber
    ExpectedDeliveryDate
    NumberOfParts
    OrderDetail[0] <Data structure Part_Detail>
        PartOrder <Data structure Part_Order>
            PartNumber
            Quantity
            InStock
        Supplier
        CommittedPrice
        OrderReference
        ExpectedDate
        ShipmentDate
        CaseNumber
        ReceptionDate
        RegistrationDate
        InvoicedPrice
    OrderDetail[1] <Data structure Part_Detail>
    OrderDetail[2] <Data structure Part_Detail>
    OrderDetail[3] <Data structure Part_Detail>
    OrderDetail[4] <Data structure Part_Detail>
    WorkOrderNumber
    RejectOrder
    OrderStatus
```

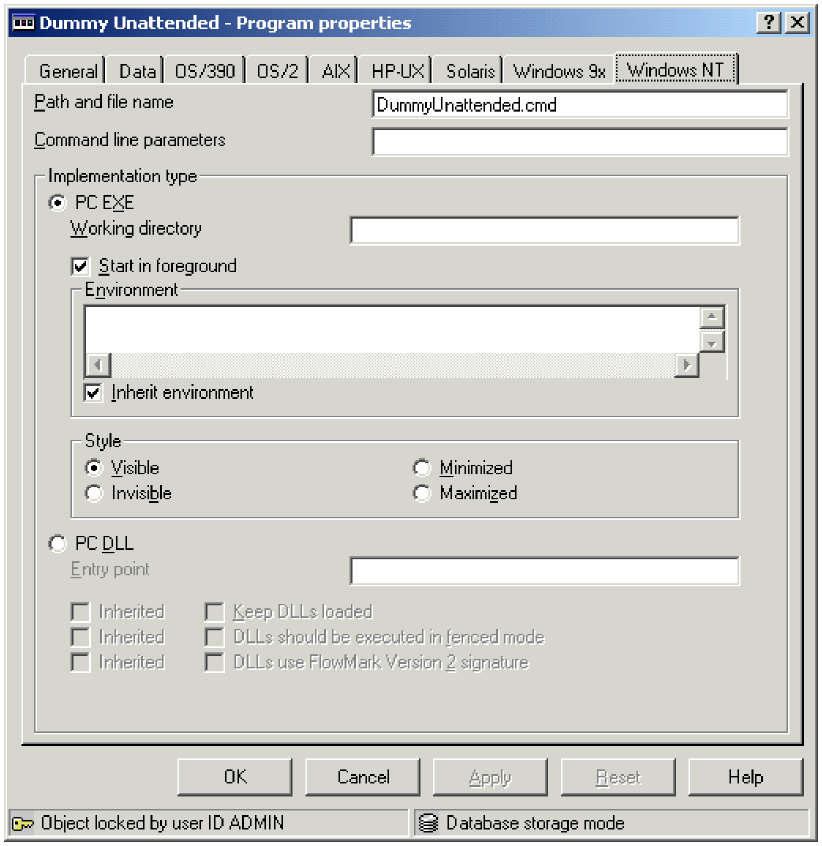
Figure 6-5 Tree view of the data structure Completed_Order_Form

6.2.2 Creating program objects

Developing and validating a process model is a repetitive process; you should deploy and validate the process model before adding more features to it. The first model that we deploy and validate only has human interaction by a single person. It enables us to validate the control logic and the data mapping. For this level of validation we only need one single program object that does not represent anything specific.

To create such a dummy program object, follow these steps:

1. Right-click the **Programs** folder and select **New Program**. An interface similar to Figure 6-6 appears, on which you provide details about the program that will run.
2. Enter a name for the program on the **General** tab.
3. On the **Data** tab, make sure that the **Program can handle any data structures** option is selected.
4. On the Windows NT® tab, provide the name of the executable, which in this case does not even have to exist physically on the WebSphere MQ Workflow server system. Of course, when you want to invoke a real program as part of the process model, you will have to provide the path and name of the program.



Path and file name: DummyUnattended.cmd

Command line parameters:

Implementation type

● PC EXE

Working directory:

☑ Start in foreground

Environment

☑ Inherit environment

Style

● Visible ○ Minimized

○ Invisible ○ Maximized

○ PC DLL

Entry point:

☐ Inherited ☐ Keep DLLs loaded

☐ Inherited ☐ DLLs should be executed in fenced mode

☐ Inherited ☐ DLLs use FlowMark Version 2 signature

Object locked by user ID ADMIN

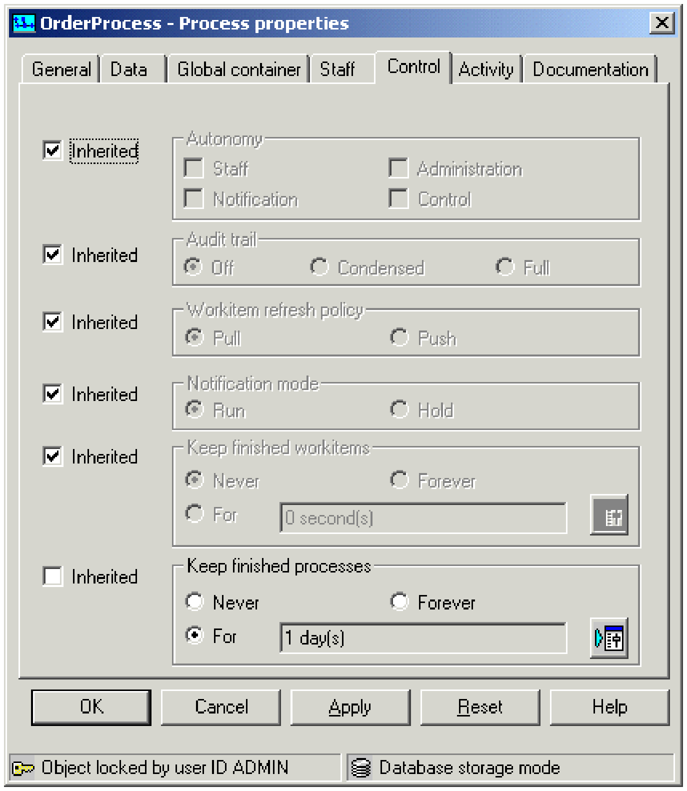
Database storage mode

Figure 6-6 Creating a dummy program object

6.2.3 Creating the process diagram

To create a process, switch to the **Processes** tab in Buildtime. Processes can be organized in categories. For the purposes of this implementation, we do not need a separate categorization, so we use the default category.

1. Right-click the **No category assigned** folder and select **New Process**.
2. On the **General** tab, provide a name for the new process, such as `OrderProcess`.
3. On the **Data** tab, identify the input and output data structures for the process itself, which is `Order_Form` in our case.
4. The Control tab, shown in Figure 6-7 on page 211, provides several options that are usually set at a higher level, for example at the domain or server level. However, here we can override the global settings with specific values. One option that we would like to override is the setting that controls whether finished processes are kept in the system for a period of time or deleted as soon as they are completed. In this case, we *deselect* the **Inherited** box to choose to **Keep finished processes** in the system **For 1 day**.


OrderProcess - Process properties

[✓] Inherited — Autonomy: [ ] Staff [ ] Administration [ ] Notification [ ] Control

[✓] Inherited — Audit trail: ● Off ○ Condensed ○ Full

[✓] Inherited — Workitem refresh policy: ● Pull ○ Push

[✓] Inherited — Notification mode: ● Run ○ Hold

[✓] Inherited — Keep finished workitems: ● Never ○ Forever ○ For 0 second(s)

[ ] Inherited — Keep finished processes: ○ Never ○ Forever ● For 1 day(s)

Object locked by user ID ADMIN

Database storage mode

Figure 6-7 Control properties of a process

5. When all properties of the process have appropriate values, click **OK**.

To see the completed process model diagram, as shown in Figure 6-8, right-click the process and select **Diagram**. Here, we provide details about this process model and how to build it.

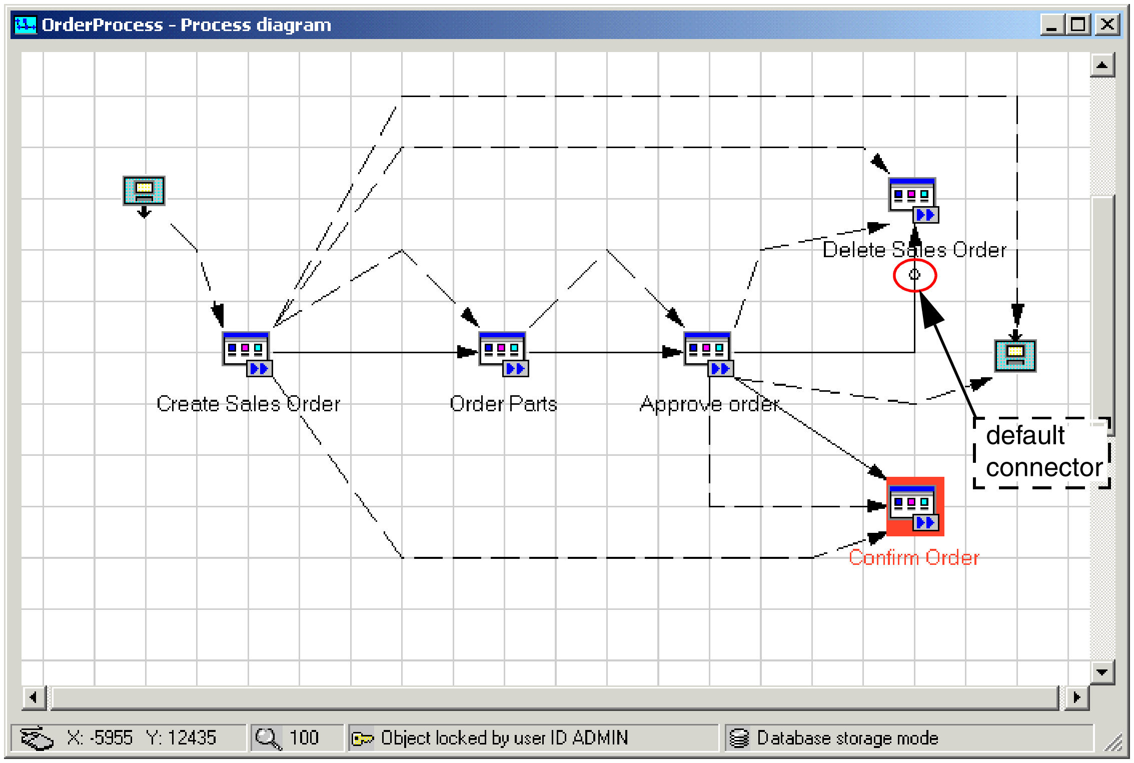
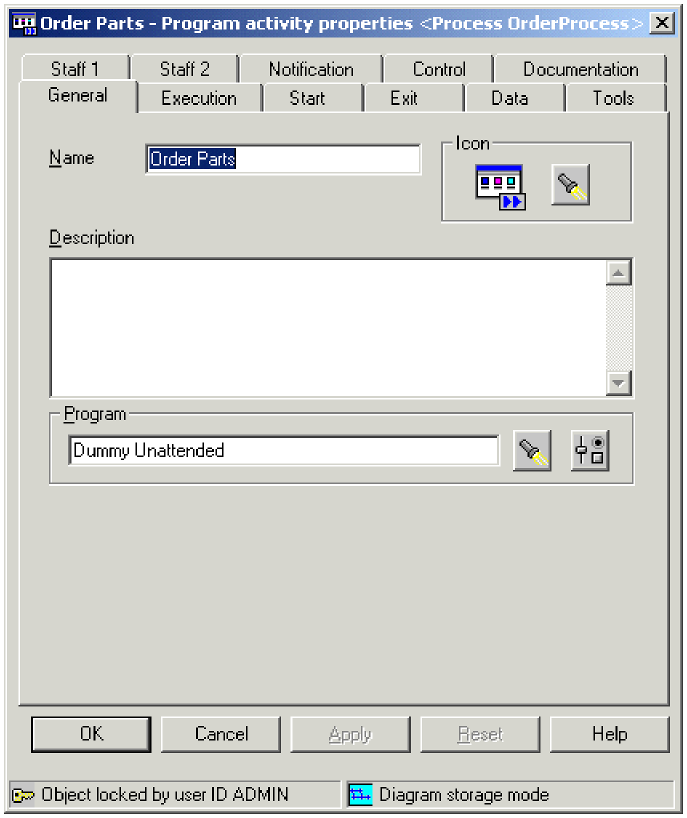


Figure 6-8 Simplified process model

1. The first step in modeling the process is to add a source node and a sink node to the diagram. These two nodes basically represent the incoming and outgoing data structure of the process. Usually, the source node has a data connector pointing to the first activity of the process model.
2. Add activities for each step in the process. Each activity is linked to a program that has to be executed. This program can be a dummy entry or a real program. Figure 6-9 on page 213 shows the details of the Order_Parts activity. The program activity is the dummy program that we created earlier.



Name: Order Parts

Description:

Program: Dummy Unattended

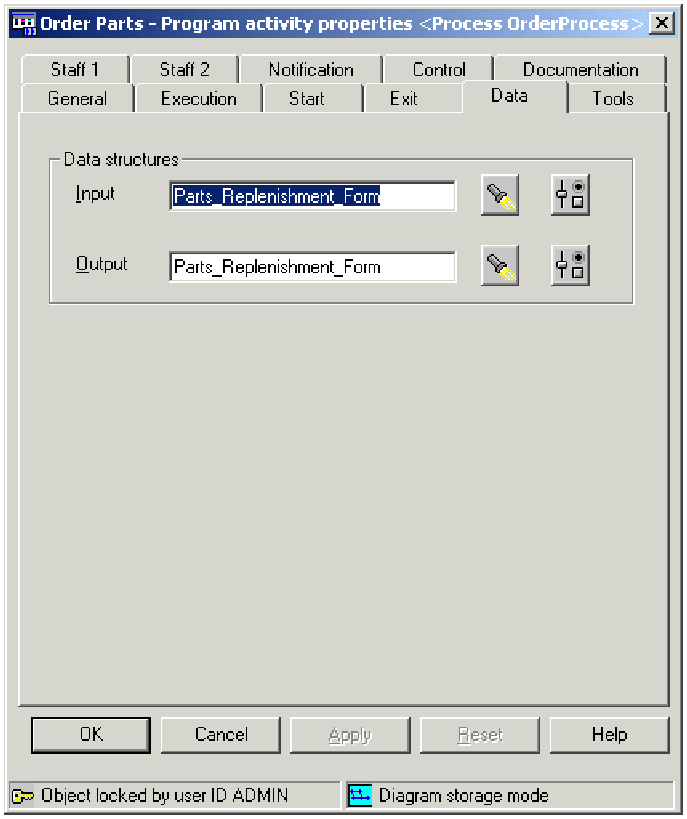
Object locked by user ID ADMIN

Diagram storage mode

Figure 6-9 General details about an activity

3. On the **Execution** tab, select the **User program execution agent** option. For automated activities, we change this to point to User program execution server (UPES), which is, at the lowest level, an MQ queue that acts as the input queue for a server component.
4. On the **Start** tab, set the option to start the activity automatically.
5. On the **Exit** tab, set the option to end the activity automatically. An alternative is the use of an exit condition in which the workflow server validates the output and, if the exit condition is not met, the activity is restarted.

At this time, all five activities have the same kind of details, except for the Data tab. Figure 6-10 shows the Data tab for the Order_Parts activity. Note that the Data structure is the same (Parts_Replenishment_Form) for both Input and Output.



Data structures

Input Parts_Replenishment_Form

Output Parts_Replenishment_Form

Object locked by user ID ADMIN

Diagram storage mode

Figure 6-10 Data structures for the Order_Parts activity

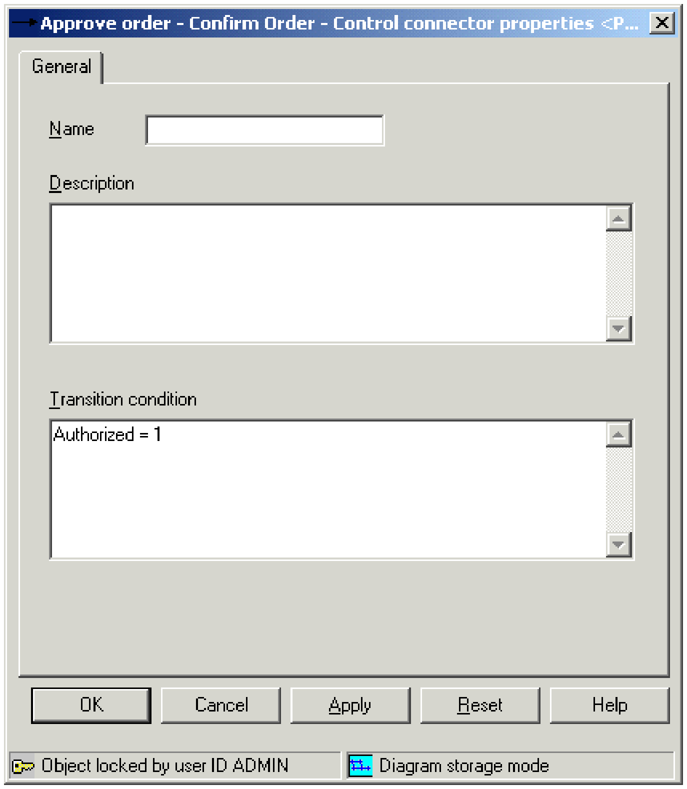
Table 6-1 lists the data structures that are used for the other activities in the process model.

Table 6-1 Data structures for the other activities

Activity	**Data structure**
Create Sales Order	Order_Form
Approve Order	Parts_Replenishment_Form
Delete Sales Order	Order_Form
Confirm_Order	Completed_Order_Form

The next step is to add the control logic to the process model by adding control connectors between the activities. Except for the link between Approve Order and Delete Order, the links are all standard connectors.

The connector between Approve Order and Confirm Order has a condition attached to it. Right-click the connector and select **Properties**. Figure 6-11 shows the properties of this connector and the transition condition. The condition refers to an element of the output data structure. If the value of the element Authorized is 1, then the condition evaluates to true. One of the start conditions of the activity is that at least one incoming connector has to evaluate to true.


Approve order - Confirm Order - Control connector properties <P...

Name:

Description:

Transition condition:
Authorized = 1

Object locked by user ID ADMIN

Diagram storage mode

Figure 6-11 Transition condition to test approval

The connector between Approve Order and Delete Order is a default connector, which is represented by a small circle in the middle (see Figure 6-8 on page 212). This type of connector is essentially the equivalent of the concept of *else* in the programming language construct *if-then-else*. The effect is that the runtime engine will traverse this path in the process model as long as the Authorized element does not have the value 1.

Besides a control flow, a process diagram also has a data flow, which is modeled with data connectors. Data connectors are represented with dotted lines. Data flow can be different from control flow. For example, the input to the Confirm Order activity is populated with the output of Create Sales Order and Approve Order, while the Confirm Order activity is only executed after the Approve Order is completed and the transition condition evaluates to true. There is no control link directly from the Create Sales Order to the Conform order activity.

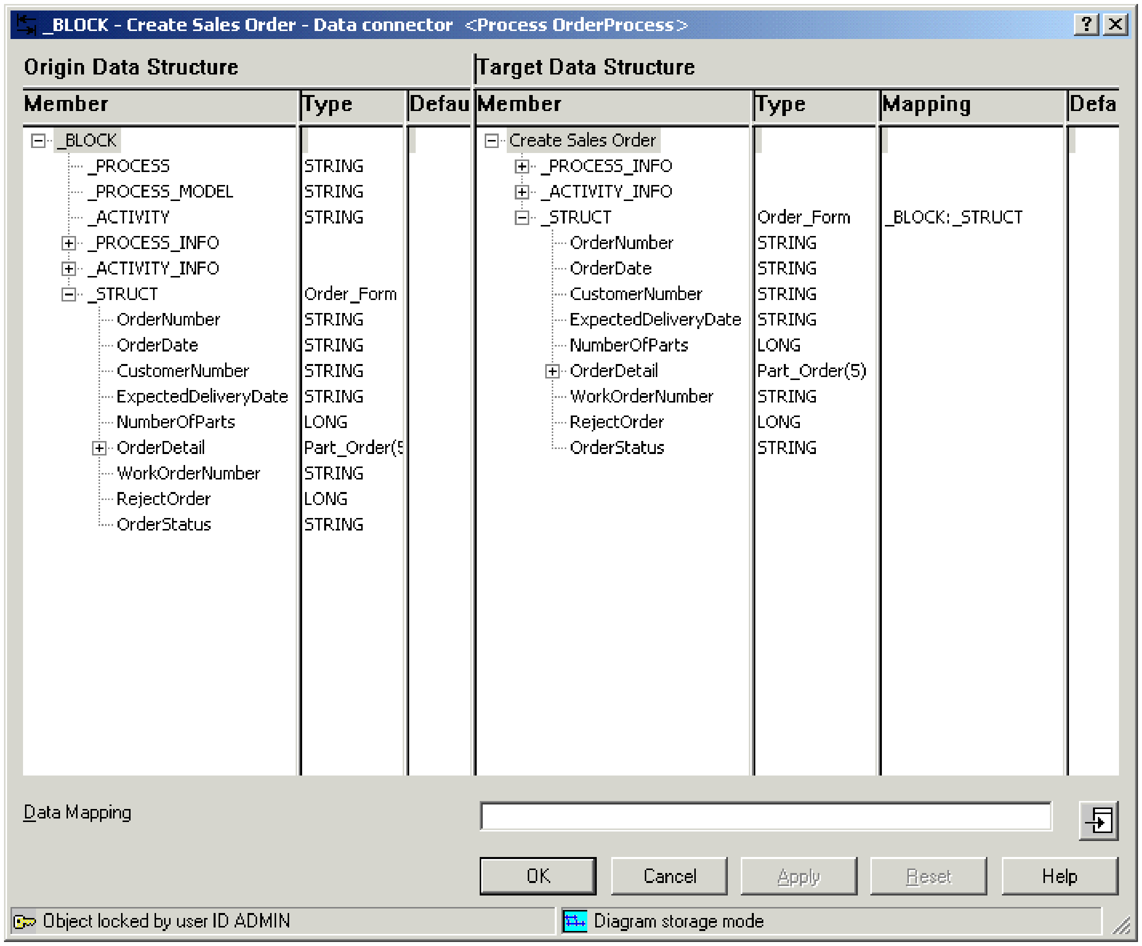
6.2.4 Creating a data mapping

The final step before we can perform the first deployment and validation of the process model is the addition of data mapping.

Note: For more complex process models, it might be a good idea to perform process deployment and validation before adding data mapping. In such a case, use a default data structure for each activity. This makes it easier to perform validation of the process logic, as long as the conditions do not refer to any elements in the data structures.

Mapping to Create Sales Order activity

To map the process input to the activity input, right-click **Create Sales Order** and select **Container Mapping** → **Mapping To**. This opens the data mapping window (Figure 6-12 on page 217).



_BLOCK - Create Sales Order - Data connector <Process OrderProcess>

Origin Data Structure

Member	Type	Defau
_BLOCK		
_PROCESS	STRING	
_PROCESS_MODEL	STRING	
_ACTIVITY	STRING	
_PROCESS_INFO		
_ACTIVITY_INFO		
_STRUCT	Order_Form	
OrderNumber	STRING	
OrderDate	STRING	
CustomerNumber	STRING	
ExpectedDeliveryDate	STRING	
NumberOfParts	LONG	
OrderDetail	Part_Order(5	
WorkOrderNumber	STRING	
RejectOrder	LONG	
OrderStatus	STRING	

Target Data Structure

Member	Type	Mapping	Defa
Create Sales Order			
_PROCESS_INFO			
_ACTIVITY_INFO			
_STRUCT	Order_Form	_BLOCK:_STRUCT	
OrderNumber	STRING		
OrderDate	STRING		
CustomerNumber	STRING		
ExpectedDeliveryDate	STRING		
NumberOfParts	LONG		
OrderDetail	Part_Order(5)		
WorkOrderNumber	STRING		
RejectOrder	LONG		
OrderStatus	STRING		

Figure 6-12 Data mapping for activity Create Sales Order

The actual mapping for this activity is easy because it uses the same data structure as the actual process. Drag the member _STRUCT from the left data structure and drop it onto the member _STRUCT on the right side to create the mapping statement _BLOCK:_STRUCT. This means that the runtime engine will copy the complete structure from the process input to the activity input.

Mapping to the Order Parts activity

The mapping for the Order Parts activity is a bit more complex. This time, the input data structure for Order Parts is different from the output data structure for Create Sales Order.

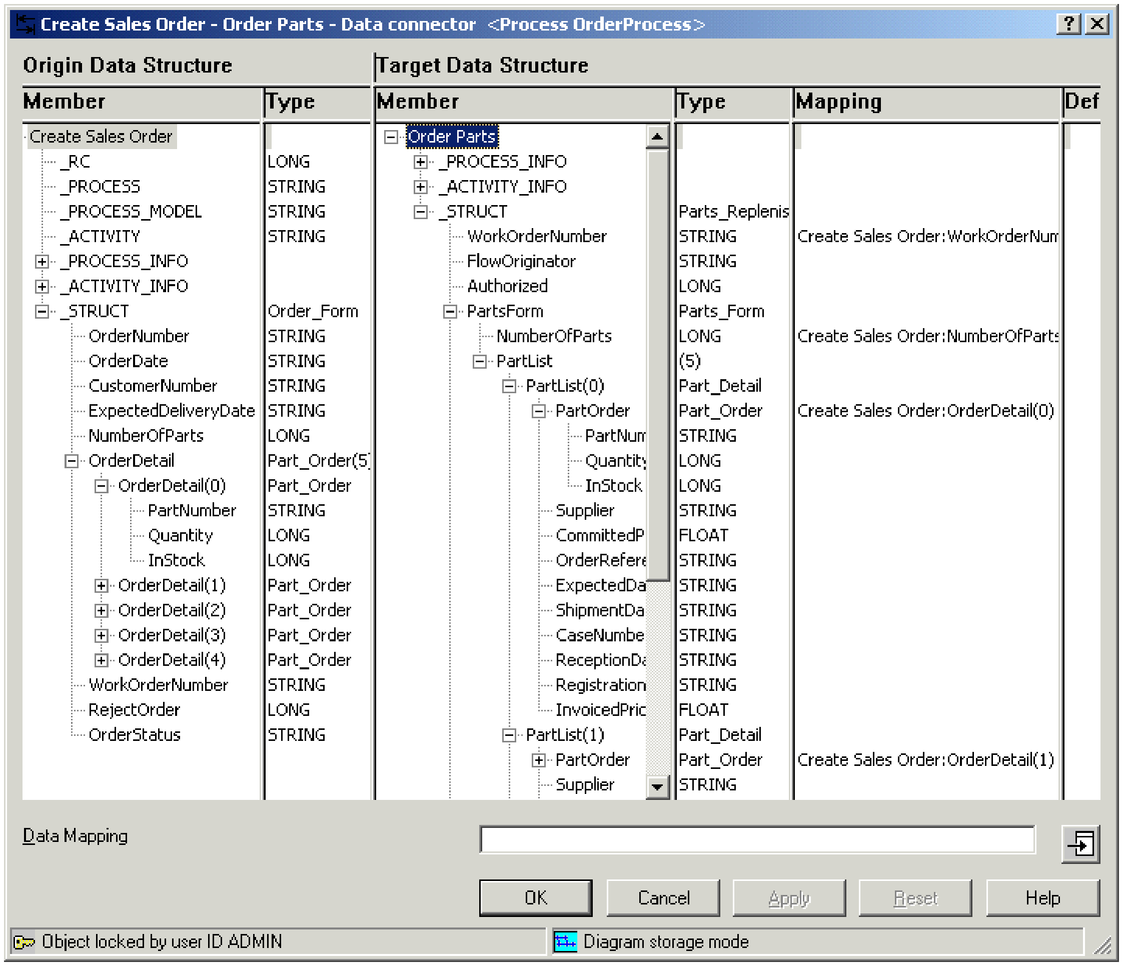
Right-click the activity and select **Container Mapping** → **Mapping To**.

Table 6-2 shows the different mappings. Each is created by dragging the element from Order_Form to the corresponding element in Parts_Replenishment_Form.

Table 6-2 Data mapping from Order_Form to Parts_Replenishment_Form

Order_Form	**Parts_Replenishment_Form**
WorkOrderNumber	WorkOrderNumber
NumberOfParts	NumberOfParts
OrderDetail(0)	PartOrder inside PartList(0)
OrderDetail(1)	PartOrder inside PartList(1)
OrderDetail(2)	PartOrder inside PartList(2)
OrderDetail(3)	PartOrder inside PartList(3)
OrderDetail(4)	PartOrder inside PartList(4)

When the mapping is completed, the data mapping window looks as shown in Figure 6-13.


Create Sales Order - Order Parts - Data connector <Process OrderProcess>

Origin Data Structure		Target Data Structure			
Member	Type	Member	Type	Mapping	Def
Create Sales Order		Order Parts			
_RC	LONG	_PROCESS_INFO			
_PROCESS	STRING	_ACTIVITY_INFO			
_PROCESS_MODEL	STRING	_STRUCT	Parts_Replenis		
_ACTIVITY	STRING	WorkOrderNumber	STRING	Create Sales Order:WorkOrderNum	
_PROCESS_INFO		FlowOriginator	STRING		
_ACTIVITY_INFO		Authorized	LONG		
_STRUCT	Order_Form	PartsForm	Parts_Form		
OrderNumber	STRING	NumberOfParts	LONG	Create Sales Order:NumberOfParts	
OrderDate	STRING	PartList	(5)		
CustomerNumber	STRING	PartList(0)	Part_Detail		
ExpectedDeliveryDate	STRING	PartOrder	Part_Order	Create Sales Order:OrderDetail(0)	
NumberOfParts	LONG	PartNum	STRING		
OrderDetail	Part_Order(5	Quantity	LONG		
OrderDetail(0)	Part_Order	InStock	LONG		
PartNumber	STRING	Supplier	STRING		
Quantity	LONG	CommittedP	FLOAT		
InStock	LONG	OrderRefere	STRING		
OrderDetail(1)	Part_Order	ExpectedDa	STRING		
OrderDetail(2)	Part_Order	ShipmentDa	STRING		
OrderDetail(3)	Part_Order	CaseNumbe	STRING		
OrderDetail(4)	Part_Order	ReceptionDa	STRING		
WorkOrderNumber	STRING	Registration	STRING		
RejectOrder	LONG	InvoicedPric	FLOAT		
OrderStatus	STRING	PartList(1)	Part_Detail		
		PartOrder	Part_Order	Create Sales Order:OrderDetail(1)	
		Supplier	STRING		

Object locked by user ID ADMIN   Diagram storage mode

Figure 6-13 Data mapping for activity Order Parts

Mapping to the Approve Order activity

This mapping is similar to the mapping for Create Sales Order. Both activities, Order Parts and Approve Order, use the same data structure. It is sufficient to map the _STRUCT members to each other.

Mapping to the Confirm Order activity

The mapping for this activity is a bit more complex because it has two incoming data connectors. The Confirm Order activity must have access to the details of the actual order and the details of the parts ordering step. The combined data is stored in the Completed_Order_Form data structure.

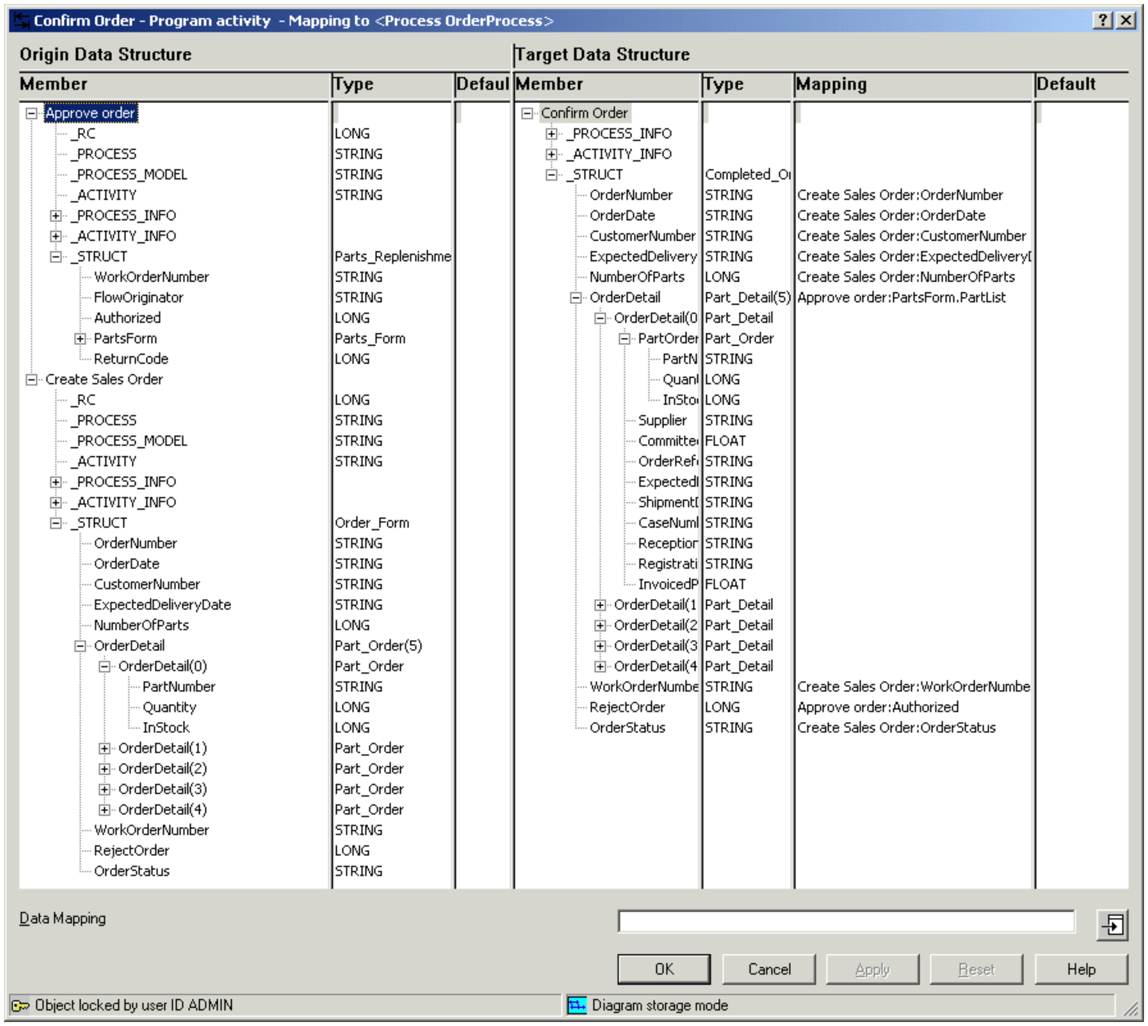
Right-click the **Confirm Order** activity and select **Container Mapping** → **Mapping To**. This time, the data mapping window shows two structures in the left part of the window, corresponding with the two incoming data connectors.

Table 6-3 lists the mappings to populate the input data structure for the Confirm Order activity. The overall values for the order are copied from the Create Sales Order activity, and the specific details about each part are copied from the output of the Approve Order activity.

Table 6-3 Data mapping for Confirm Order

Parts_Replenishment_Form	**Completed_Order_Form**
PartsForm.PartList	OrderDetail
Authorized	Authorized
Order_Form	
OrderNumber	OrderNumber
OrderDate	OrderDate
CustomerNumber	CustomerNumber
ExpectedDeliveryDate	ExpectedDeliveryDate
NumberOfParts	NumberOfParts
WorkOrderNumber	WorkOrderNumber
OrderStatus	OrderStatus

Figure 6-14 shows the data mapping window when the mappings of Table 6-3 are implemented.


Confirm Order - Program activity - Mapping to <Process OrderProcess>

Origin Data Structure

Member	Type	Defaul
Approve order		
_RC	LONG	
_PROCESS	STRING	
_PROCESS_MODEL	STRING	
_ACTIVITY	STRING	
_PROCESS_INFO		
_ACTIVITY_INFO		
_STRUCT	Parts_Replenishme	
WorkOrderNumber	STRING	
FlowOriginator	STRING	
Authorized	LONG	
PartsForm	Parts_Form	
ReturnCode	LONG	
Create Sales Order		
_RC	LONG	
_PROCESS	STRING	
_PROCESS_MODEL	STRING	
_ACTIVITY	STRING	
_PROCESS_INFO		
_ACTIVITY_INFO		
_STRUCT	Order_Form	
OrderNumber	STRING	
OrderDate	STRING	
CustomerNumber	STRING	
ExpectedDeliveryDate	STRING	
NumberOfParts	LONG	
OrderDetail	Part_Order(5)	
OrderDetail(0)	Part_Order	
PartNumber	STRING	
Quantity	LONG	
InStock	LONG	
OrderDetail(1)	Part_Order	
OrderDetail(2)	Part_Order	
OrderDetail(3)	Part_Order	
OrderDetail(4)	Part_Order	
WorkOrderNumber	STRING	
RejectOrder	LONG	
OrderStatus	STRING	

Target Data Structure

Member	Type	Mapping	Default
Confirm Order			
_PROCESS_INFO			
_ACTIVITY_INFO			
_STRUCT	Completed_Or		
OrderNumber	STRING	Create Sales Order:OrderNumber	
OrderDate	STRING	Create Sales Order:OrderDate	
CustomerNumber	STRING	Create Sales Order:CustomerNumber	
ExpectedDelivery	STRING	Create Sales Order:ExpectedDelivery[	
NumberOfParts	LONG	Create Sales Order:NumberOfParts	
OrderDetail	Part_Detail(5)	Approve order:PartsForm.PartList	
OrderDetail(0	Part_Detail		
PartOrder	Part_Order		
PartN	STRING		
Quan	LONG		
InSto	LONG		
Supplier	STRING		
Committe	FLOAT		
OrderRef	STRING		
Expected	STRING		
Shipment	STRING		
CaseNum	STRING		
Receptior	STRING		
Registrati	STRING		
InvoicedP	FLOAT		
OrderDetail(1	Part_Detail		
OrderDetail(2	Part_Detail		
OrderDetail(3	Part_Detail		
OrderDetail(4	Part_Detail		
WorkOrderNumbe	STRING	Create Sales Order:WorkOrderNumbe	
RejectOrder	LONG	Approve order:Authorized	
OrderStatus	STRING	Create Sales Order:OrderStatus	

Data Mapping

Object locked by user ID ADMIN

Diagram storage mode

Figure 6-14 Data mapping for the Confirm Order activity

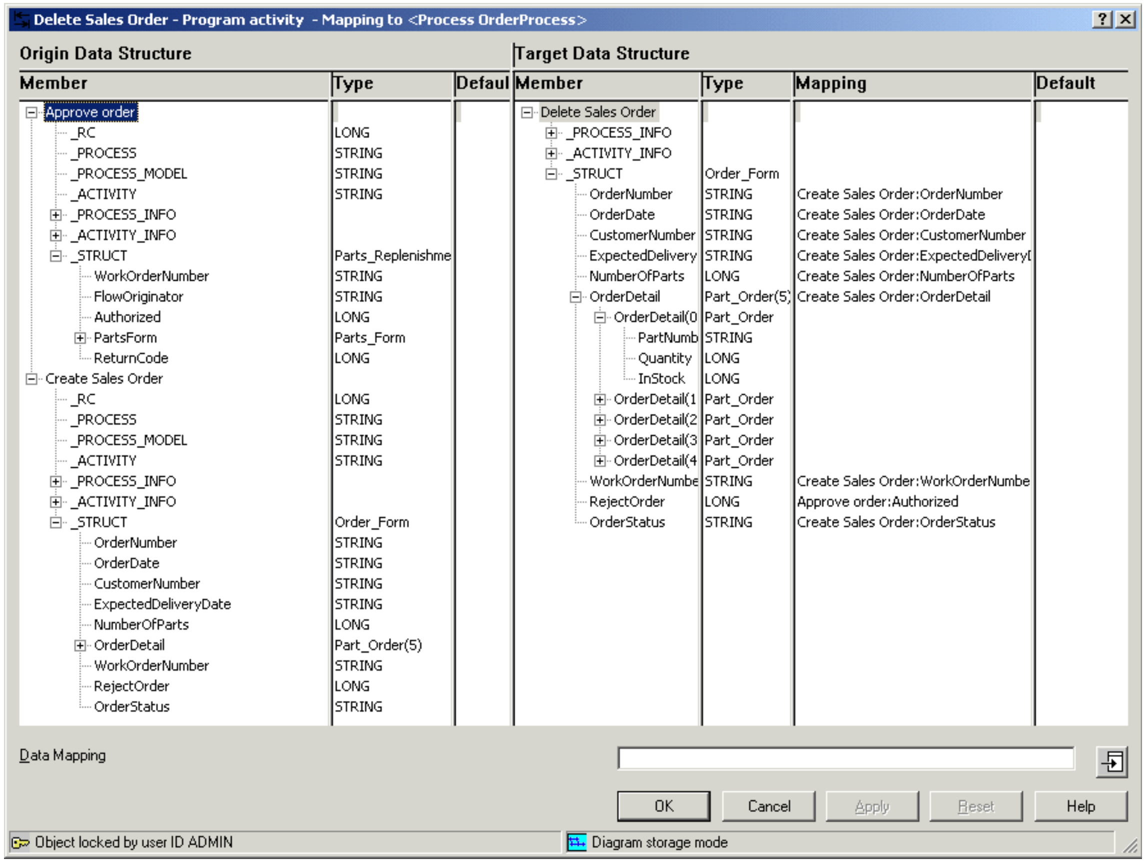
Mapping to the Delete Order activity

The mapping to the Delete Order activity is similar to mapping to Confirm Order. The two sources are the Order_Form from the Create Sales Order activity and the Parts_Replenishment_Form from the Approve Order activity. The target data structure this time is the Order_Form data structure, because there is no need to carry the details of the parts ordering step into the Delete Order activity. The only element to be mapped from the Parts_Replenishment_Form is the Authorized element that carries the disapproval code for this order.

Table 6-4 lists the mapping details for this activity, and Figure 6-15 on page 223 shows the data mapping window with the mapping statements.

Table 6-4 Data mapping for Delete Order

Parts_Replenishment_Form	**Order_Form**
Authorized	Authorized
Order_Form	
OrderNumber	OrderNumber
OrderDate	OrderDate
CustomerNumber	CustomerNumber
ExpectedDeliveryDate	ExpectedDeliveryDate
NumberOfParts	NumberOfParts
WorkOrderNumber	WorkOrderNumber
OrderStatus	OrderStatus



Delete Sales Order - Program activity - Mapping to <Process OrderProcess>

Origin Data Structure

Member	Type	Defaul
Approve order		
_RC	LONG	
_PROCESS	STRING	
_PROCESS_MODEL	STRING	
_ACTIVITY	STRING	
_PROCESS_INFO		
_ACTIVITY_INFO		
_STRUCT	Parts_Replenishme	
WorkOrderNumber	STRING	
FlowOriginator	STRING	
Authorized	LONG	
PartsForm	Parts_Form	
ReturnCode	LONG	
Create Sales Order		
_RC	LONG	
_PROCESS	STRING	
_PROCESS_MODEL	STRING	
_ACTIVITY	STRING	
_PROCESS_INFO		
_ACTIVITY_INFO		
_STRUCT	Order_Form	
OrderNumber	STRING	
OrderDate	STRING	
CustomerNumber	STRING	
ExpectedDeliveryDate	STRING	
NumberOfParts	LONG	
OrderDetail	Part_Order(5)	
WorkOrderNumber	STRING	
RejectOrder	LONG	
OrderStatus	STRING	

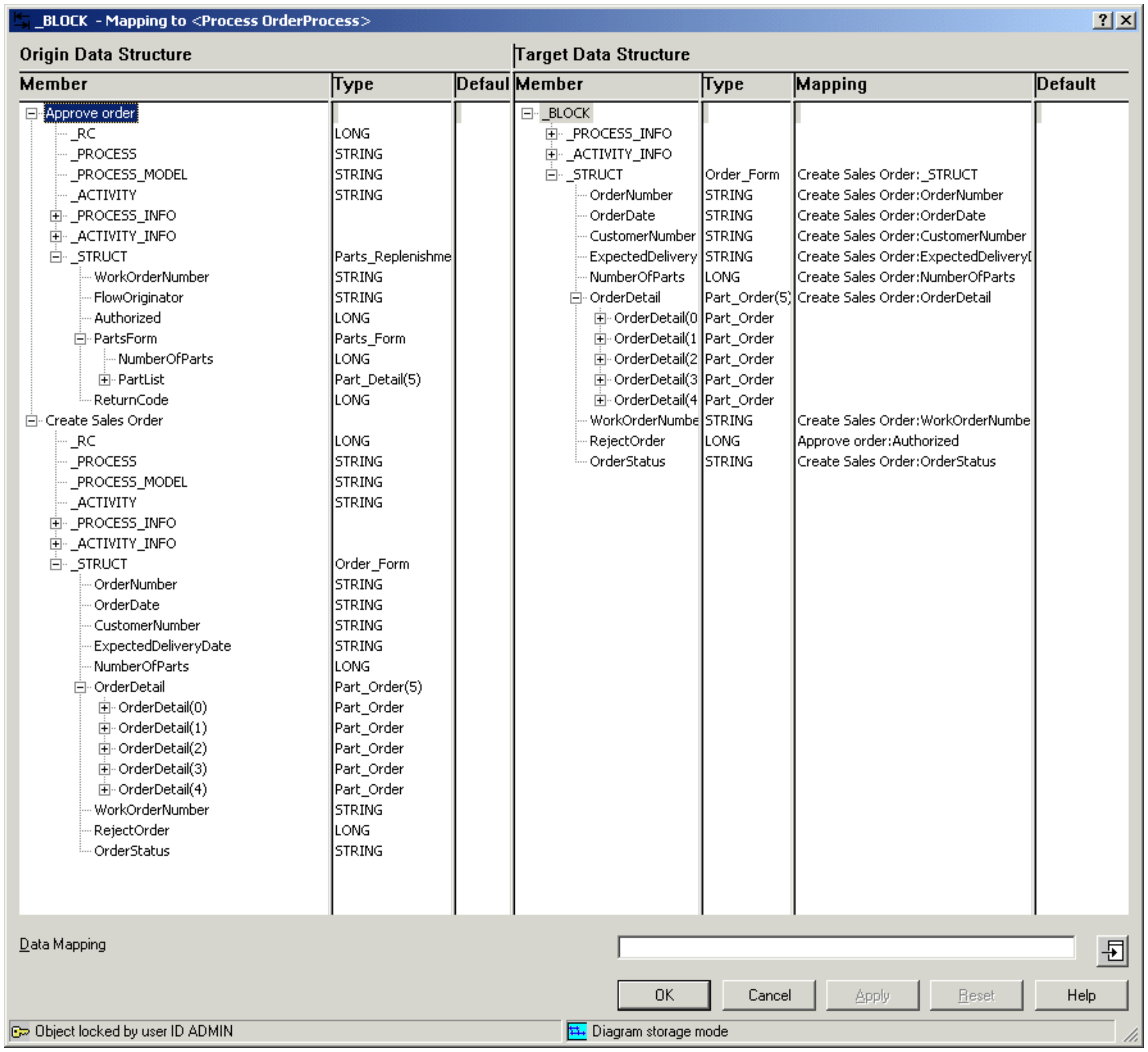
Target Data Structure

Member	Type	Mapping	Default
Delete Sales Order			
_PROCESS_INFO			
_ACTIVITY_INFO			
_STRUCT	Order_Form		
OrderNumber	STRING	Create Sales Order:OrderNumber	
OrderDate	STRING	Create Sales Order:OrderDate	
CustomerNumber	STRING	Create Sales Order:CustomerNumber	
ExpectedDelivery	STRING	Create Sales Order:ExpectedDelivery[	
NumberOfParts	LONG	Create Sales Order:NumberOfParts	
OrderDetail	Part_Order(5)	Create Sales Order:OrderDetail	
OrderDetail(0	Part_Order		
PartNumb	STRING		
Quantity	LONG		
InStock	LONG		
OrderDetail(1	Part_Order		
OrderDetail(2	Part_Order		
OrderDetail(3	Part_Order		
OrderDetail(4	Part_Order		
WorkOrderNumbe	STRING	Create Sales Order:WorkOrderNumbe	
RejectOrder	LONG	Approve order:Authorized	
OrderStatus	STRING	Create Sales Order:OrderStatus	

Figure 6-15 Data mapping for the Delete Order activity

Mapping to the sink node

The final mapping is to the sink node, which represents the output data structure. Here we want to copy the original Order_Form and the Authorized field from the approval step. The mapping details are similar to the previous activities. Figure 6-16 on page 224 shows the details.


_BLOCK - Mapping to <Process OrderProcess>

Origin Data Structure

Member	Type	Defaul
Approve order		
_RC	LONG	
_PROCESS	STRING	
_PROCESS_MODEL	STRING	
_ACTIVITY	STRING	
_PROCESS_INFO		
_ACTIVITY_INFO		
_STRUCT	Parts_Replenishme	
WorkOrderNumber	STRING	
FlowOriginator	STRING	
Authorized	LONG	
PartsForm	Parts_Form	
NumberOfParts	LONG	
PartList	Part_Detail(5)	
ReturnCode	LONG	
Create Sales Order		
_RC	LONG	
_PROCESS	STRING	
_PROCESS_MODEL	STRING	
_ACTIVITY	STRING	
_PROCESS_INFO		
_ACTIVITY_INFO		
_STRUCT	Order_Form	
OrderNumber	STRING	
OrderDate	STRING	
CustomerNumber	STRING	
ExpectedDeliveryDate	STRING	
NumberOfParts	LONG	
OrderDetail	Part_Order(5)	
OrderDetail(0)	Part_Order	
OrderDetail(1)	Part_Order	
OrderDetail(2)	Part_Order	
OrderDetail(3)	Part_Order	
OrderDetail(4)	Part_Order	
WorkOrderNumber	STRING	
RejectOrder	LONG	
OrderStatus	STRING	

Target Data Structure

Member	Type	Mapping	Default
_BLOCK			
_PROCESS_INFO			
_ACTIVITY_INFO			
_STRUCT	Order_Form	Create Sales Order:_STRUCT	
OrderNumber	STRING	Create Sales Order:OrderNumber	
OrderDate	STRING	Create Sales Order:OrderDate	
CustomerNumber	STRING	Create Sales Order:CustomerNumber	
ExpectedDelivery	STRING	Create Sales Order:ExpectedDelivery[	
NumberOfParts	LONG	Create Sales Order:NumberOfParts	
OrderDetail	Part_Order(5)	Create Sales Order:OrderDetail	
OrderDetail(0	Part_Order		
OrderDetail(1	Part_Order		
OrderDetail(2	Part_Order		
OrderDetail(3	Part_Order		
OrderDetail(4	Part_Order		
WorkOrderNumbe	STRING	Create Sales Order:WorkOrderNumbe	
RejectOrder	LONG	Approve order:Authorized	
OrderStatus	STRING	Create Sales Order:OrderStatus	

Data Mapping

OK Cancel Apply Reset Help

Object locked by user ID ADMIN | Diagram storage mode

Figure 6-16 Data mapping for the sink node

This completes the process definition. We can now deploy it to the runtime and verify that the model is correct.

6.3 Deploying the process flow in Runtime server

To ensure that no errors are present in the newly created FDL file, a sanity check can be done within the Buildtime environment.

Go to **Process** → **Verify** (Figure 6-17).

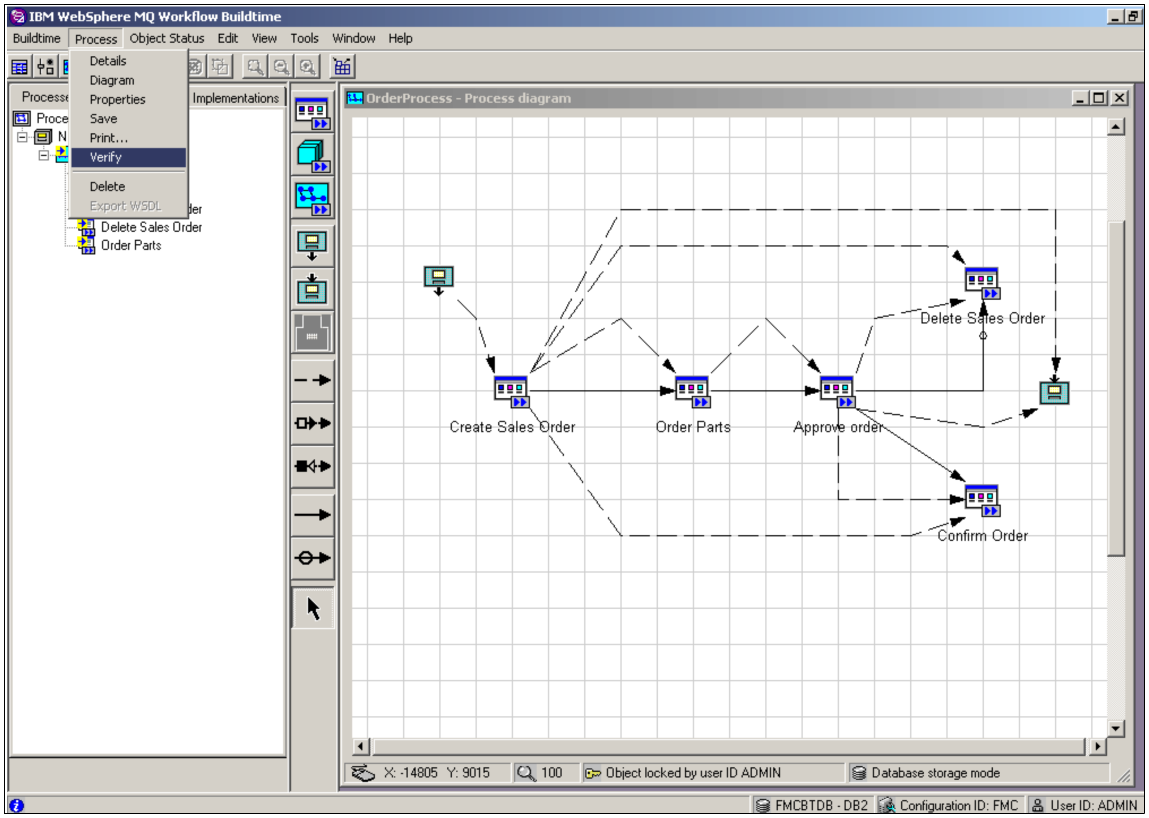
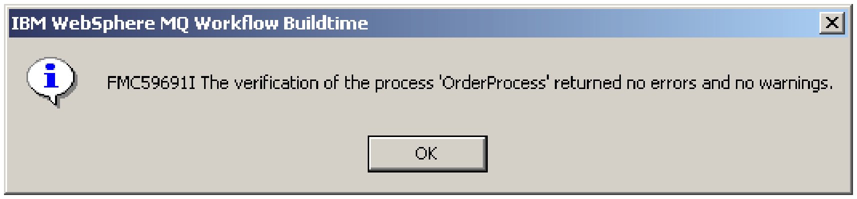


Figure 6-17 Buildtime verification



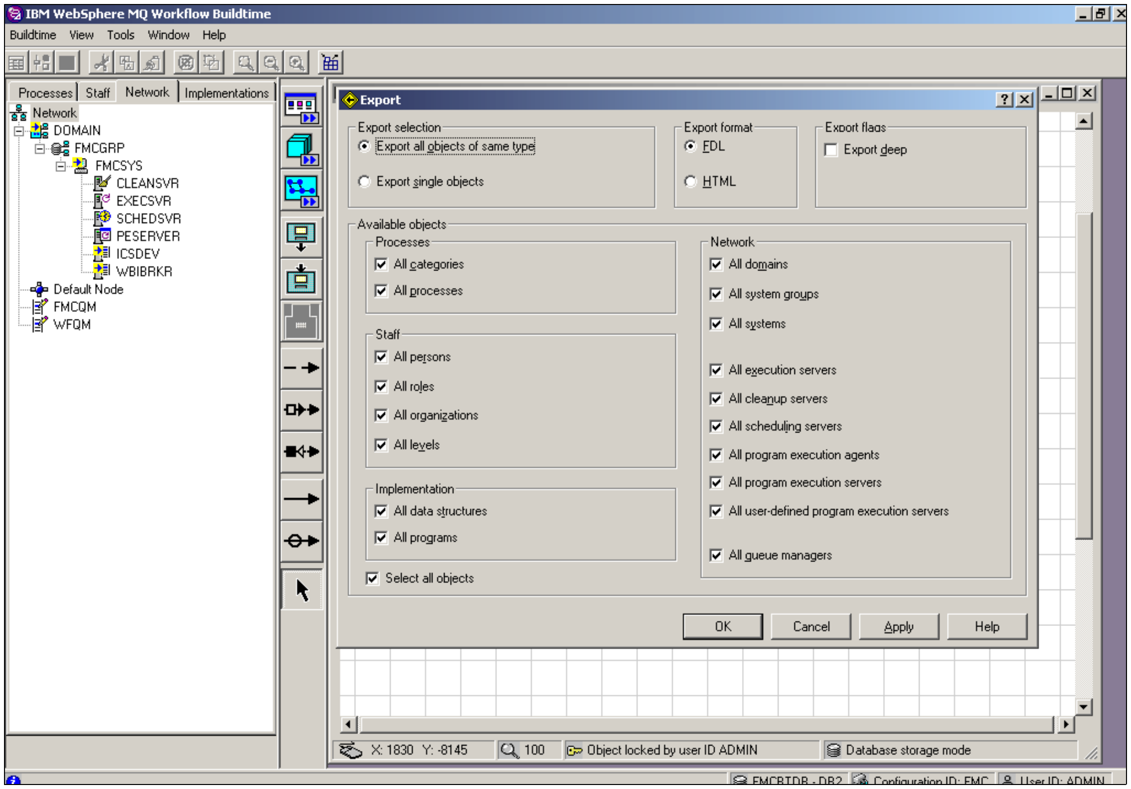
FMC59691I The verification of the process 'OrderProcess' returned no errors and no warnings.

Figure 6-18 Verification response

When the FDL is validated (Figure 6-18), you may proceed to export the FDL.

To start the export, follow these steps:

1. Select **File** → **export**. A new dialog box opens (Figure 6-19).



Export

Export selection
- [✓] Export all objects of same type
- [ ] Export single objects

Export format
- [✓] FDL
- [ ] HTML

Export flags
- [ ] Export deep

Available objects

Processes
- [✓] All categories
- [✓] All processes

Staff
- [✓] All persons
- [✓] All roles
- [✓] All organizations
- [✓] All levels

Implementation
- [✓] All data structures
- [✓] All programs

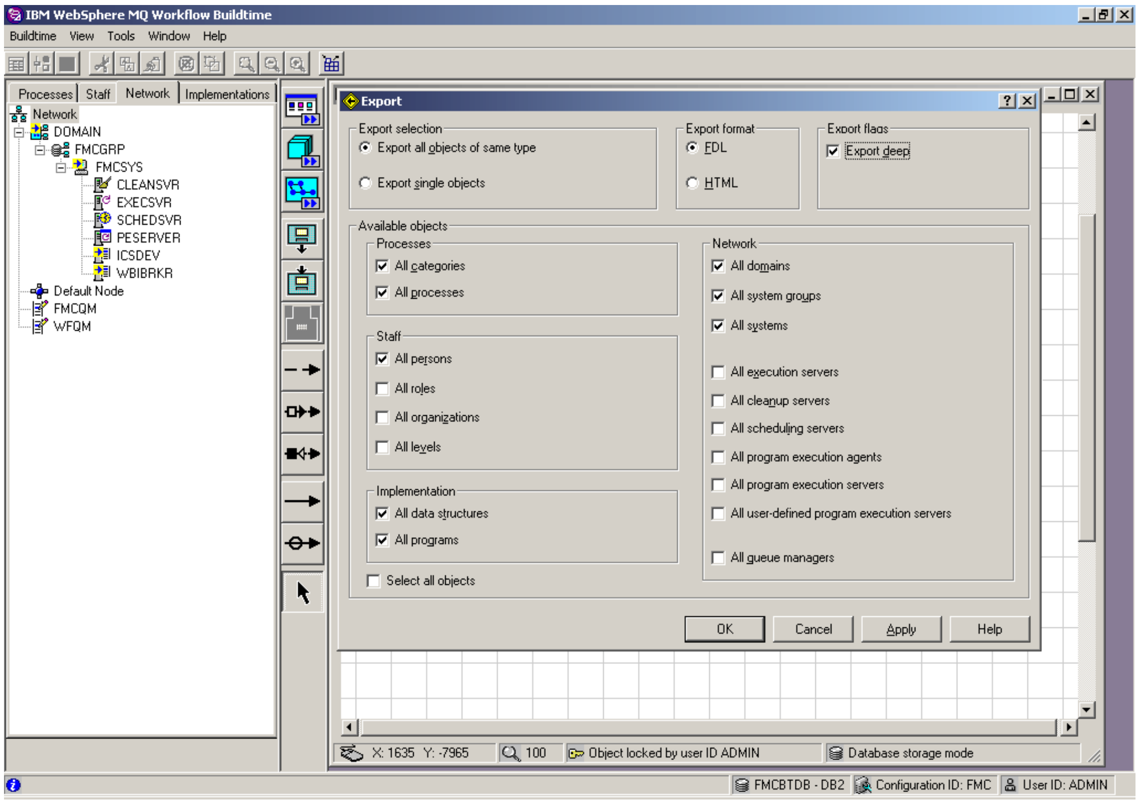
Network
- [✓] All domains
- [✓] All system groups
- [✓] All systems
- [✓] All execution servers
- [✓] All cleanup servers
- [✓] All scheduling servers
- [✓] All program execution agents
- [✓] All program execution servers
- [✓] All user-defined program execution servers
- [✓] All queue managers

- [✓] Select all objects

OK Cancel Apply Help

Figure 6-19 Export process model

2. Deselect **Select all objects**. By deselecting this item, you have the option of choosing specific objects to be exported (Figure 6-20 on page 227).



Export

Export selection
- (●) Export all objects of same type
- ( ) Export single objects

Export format
- (●) FDL
- ( ) HTML

Export flags
- [✓] Export deep

Available objects

Processes
- [✓] All categories
- [✓] All processes

Staff
- [✓] All persons
- [ ] All roles
- [ ] All organizations
- [ ] All levels

Implementation
- [✓] All data structures
- [✓] All programs

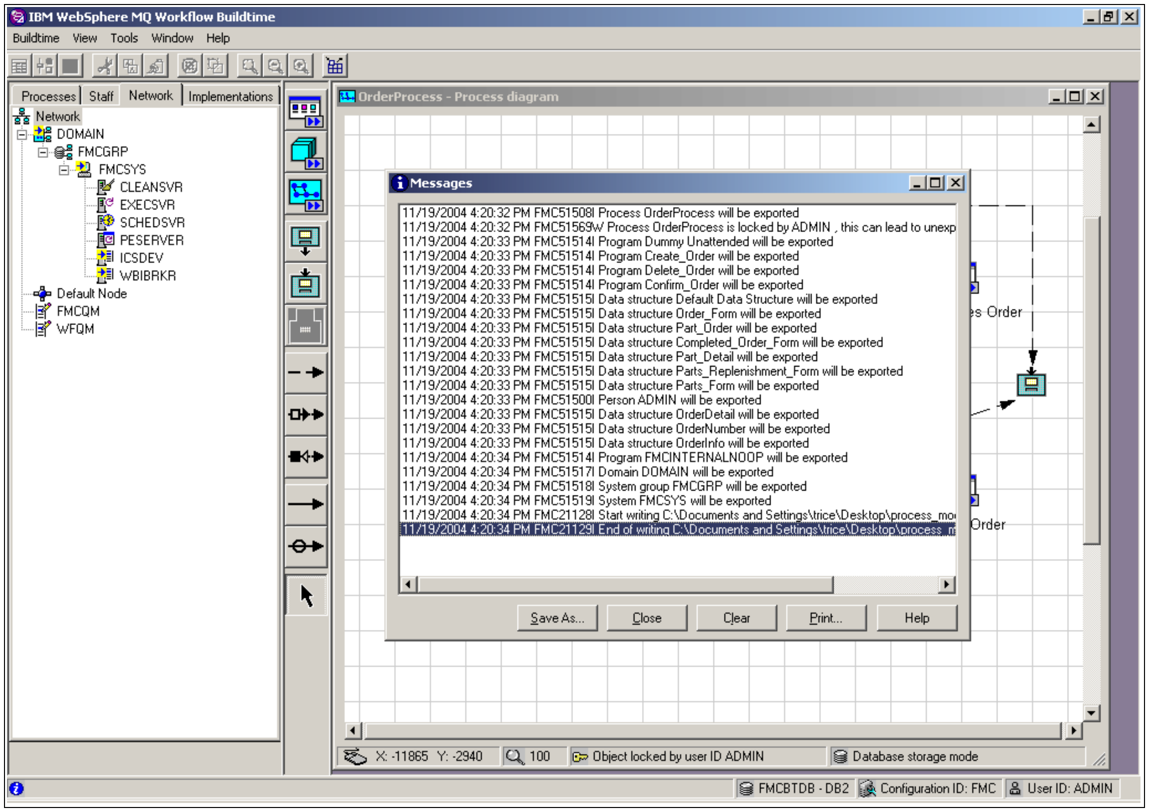
- [ ] Select all objects

Network
- [✓] All domains
- [✓] All system groups
- [✓] All systems
- [ ] All execution servers
- [ ] All cleanup servers
- [ ] All scheduling servers
- [ ] All program execution agents
- [ ] All program execution servers
- [ ] All user-defined program execution servers
- [ ] All queue managers

Figure 6-20 Export choices

3. After choosing the objects to be included in the FDL file, select **OK** and enter a file name. Press **save.**

   A log of the export is generated (Figure 6-21 on page 228). Review it to see that everything is as expected.


IBM WebSphere MQ Workflow Buildtime
Buildtime View Tools Window Help
Processes Staff Network Implementations
Network
DOMAIN
FMCGRP
FMCSYS
CLEANSVR
EXECSVR
SCHEDSVR
PESERVER
ICSDEV
WBIBRKR
Default Node
FMCQM
WFQM
OrderProcess - Process diagram
Messages
Save As... Close Clear Print... Help
X: -11865 Y: -2940
100
Object locked by user ID ADMIN
Database storage mode
FMCBTDB - DB2
Configuration ID: FMC
User ID: ADMIN

```
11/19/2004 4:20:32 PM FMC51508I Process OrderProcess will be exported
11/19/2004 4:20:32 PM FMC51569W Process OrderProcess is locked by ADMIN , this can lead to unexp
11/19/2004 4:20:33 PM FMC51514I Program Dummy Unattended will be exported
11/19/2004 4:20:33 PM FMC51514I Program Create_Order will be exported
11/19/2004 4:20:33 PM FMC51514I Program Delete_Order will be exported
11/19/2004 4:20:33 PM FMC51514I Program Confirm_Order will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure Default Data Structure will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure Order_Form will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure Part_Order will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure Completed_Order_Form will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure Part_Detail will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure Parts_Replenishment_Form will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure Parts_Form will be exported
11/19/2004 4:20:33 PM FMC51500I Person ADMIN will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure OrderDetail will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure OrderNumber will be exported
11/19/2004 4:20:33 PM FMC51515I Data structure OrderInfo will be exported
11/19/2004 4:20:34 PM FMC51514I Program FMCINTERNALNOOP will be exported
11/19/2004 4:20:34 PM FMC51517I Domain DOMAIN will be exported
11/19/2004 4:20:34 PM FMC51518I System group FMCGRP will be exported
11/19/2004 4:20:34 PM FMC51519I System FMCSYS will be exported
11/19/2004 4:20:34 PM FMC21128I Start writing C:\Documents and Settings\trice\Desktop\process_mo
11/19/2004 4:20:34 PM FMC21129I End of writing C:\Documents and Settings\trice\Desktop\process_m
```

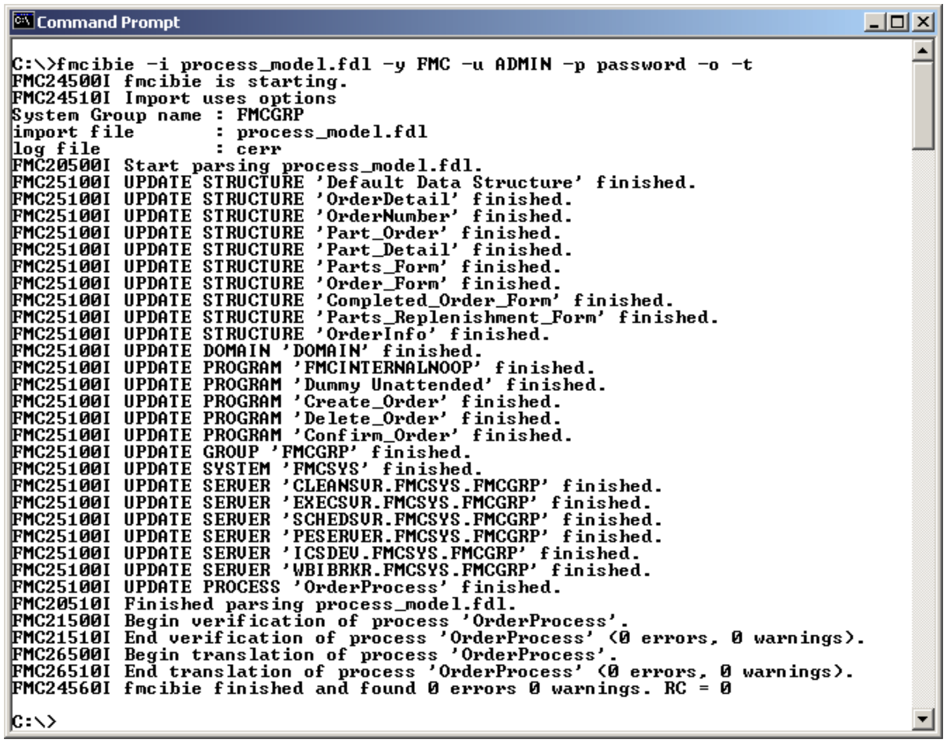
Figure 6-21 Messages during export to FDL

After the FDL is exported, the file must be transferred and imported into the WebSphere MQ Worklfow runtime environment. The import process can be performed either directly on the runtime server or by using a client machine that has the administrative components of WebSphere MQ Workflow installed, such as the administrative client we implemented in Chapter 4, “Implementing client components” on page 119.

Note: To use the import utility, the administrative client machine must have DB2 installed. For other functions, only an MQ client is required.

To import the process model into the runtime, we use the **`fmcibie`** utility. The `-i` option is used to point to the file that we want to import. The `-y` parameter points to the name of the configuration. The `-u` and `-p` parameters are used to provide the user ID and password of the workflow administrator. Finally, the `-to` parameter tells the utility to overwrite any existing objects if required and to translate the process model into a process template, which can be executed in the runtime environment.

Figure 6-22 shows the output of the import (`fmcibie`) utility. When the import completes, we can use the runtime client facilities to test the workflow process.



```
C:\>fmcibie -i process_model.fdl -y FMC -u ADMIN -p password -o -t
FMC24500I fmcibie is starting.
FMC24510I Import uses options
System Group name : FMCGRP
import file       : process_model.fdl
log file          : cerr
FMC20500I Start parsing process_model.fdl.
FMC25100I UPDATE STRUCTURE 'Default Data Structure' finished.
FMC25100I UPDATE STRUCTURE 'OrderDetail' finished.
FMC25100I UPDATE STRUCTURE 'OrderNumber' finished.
FMC25100I UPDATE STRUCTURE 'Part_Order' finished.
FMC25100I UPDATE STRUCTURE 'Part_Detail' finished.
FMC25100I UPDATE STRUCTURE 'Parts_Form' finished.
FMC25100I UPDATE STRUCTURE 'Order_Form' finished.
FMC25100I UPDATE STRUCTURE 'Completed_Order_Form' finished.
FMC25100I UPDATE STRUCTURE 'Parts_Replenishment_Form' finished.
FMC25100I UPDATE STRUCTURE 'OrderInfo' finished.
FMC25100I UPDATE DOMAIN 'DOMAIN' finished.
FMC25100I UPDATE PROGRAM 'FMCINTERNALNOOP' finished.
FMC25100I UPDATE PROGRAM 'Dummy Unattended' finished.
FMC25100I UPDATE PROGRAM 'Create_Order' finished.
FMC25100I UPDATE PROGRAM 'Delete_Order' finished.
FMC25100I UPDATE PROGRAM 'Confirm_Order' finished.
FMC25100I UPDATE GROUP 'FMCGRP' finished.
FMC25100I UPDATE SYSTEM 'FMCSYS' finished.
FMC25100I UPDATE SERVER 'CLEANSVR.FMCSYS.FMCGRP' finished.
FMC25100I UPDATE SERVER 'EXECSVR.FMCSYS.FMCGRP' finished.
FMC25100I UPDATE SERVER 'SCHEDSVR.FMCSYS.FMCGRP' finished.
FMC25100I UPDATE SERVER 'PESERVER.FMCSYS.FMCGRP' finished.
FMC25100I UPDATE SERVER 'ICSDEV.FMCSYS.FMCGRP' finished.
FMC25100I UPDATE SERVER 'WBIBRKR.FMCSYS.FMCGRP' finished.
FMC25100I UPDATE PROCESS 'OrderProcess' finished.
FMC20510I Finished parsing process_model.fdl.
FMC21500I Begin verification of process 'OrderProcess'.
FMC21510I End verification of process 'OrderProcess' (0 errors, 0 warnings).
FMC26500I Begin translation of process 'OrderProcess'.
FMC26510I End translation of process 'OrderProcess' (0 errors, 0 warnings).
FMC24560I fmcibie finished and found 0 errors 0 warnings. RC = 0

C:\>
```

Figure 6-22 Output from fmcibie utility

6.4 Validating the workflow process flow

There are many different ways to start a WebSphere MQ Workflow process flow, including:

- WebSphere MQ Workflow runtime client
- WebSphere MQ Workflow Web client
- Custom-written client interface
- Sending a WebSphere MQ message to WebSphere MQ Workflow

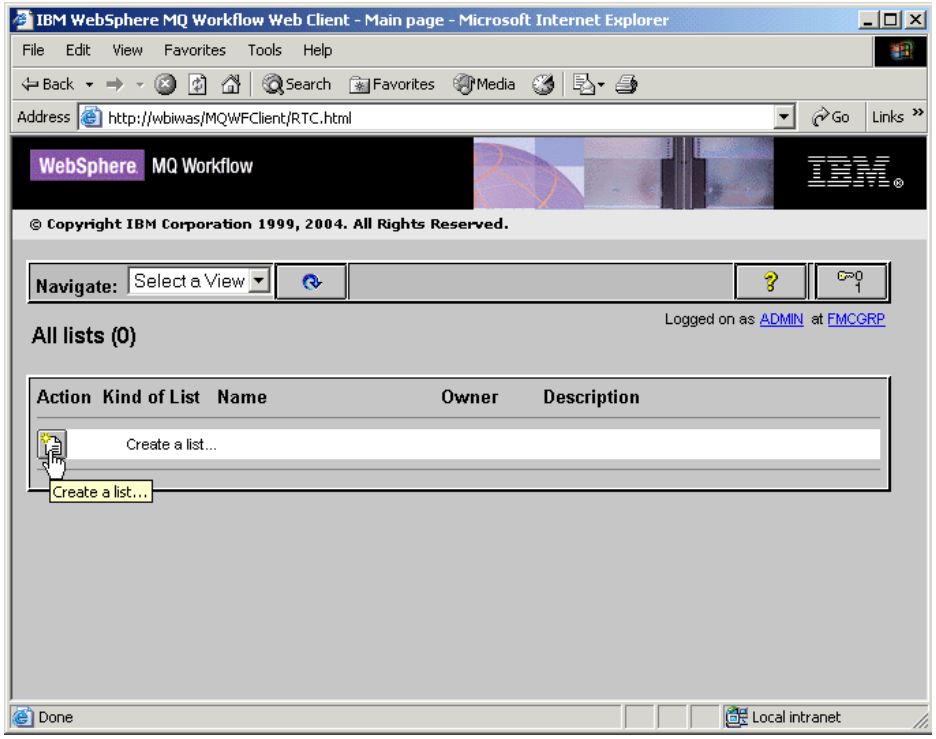
For our test and validation purposes, the Web client is the most appropriate interface. In 4.1, “Implementing WebSphere MQ Workflow Web Client” on page 120 we described the implementation of the Web Client. This is the process to test it.

1. Open a browser and point it to:

   `http://`*`wbiwas`*`/MQWFClient/RTC.html`

   `wbiwas` is the host name of the server where WebSphere Application Server is running and where the Web Client is deployed.

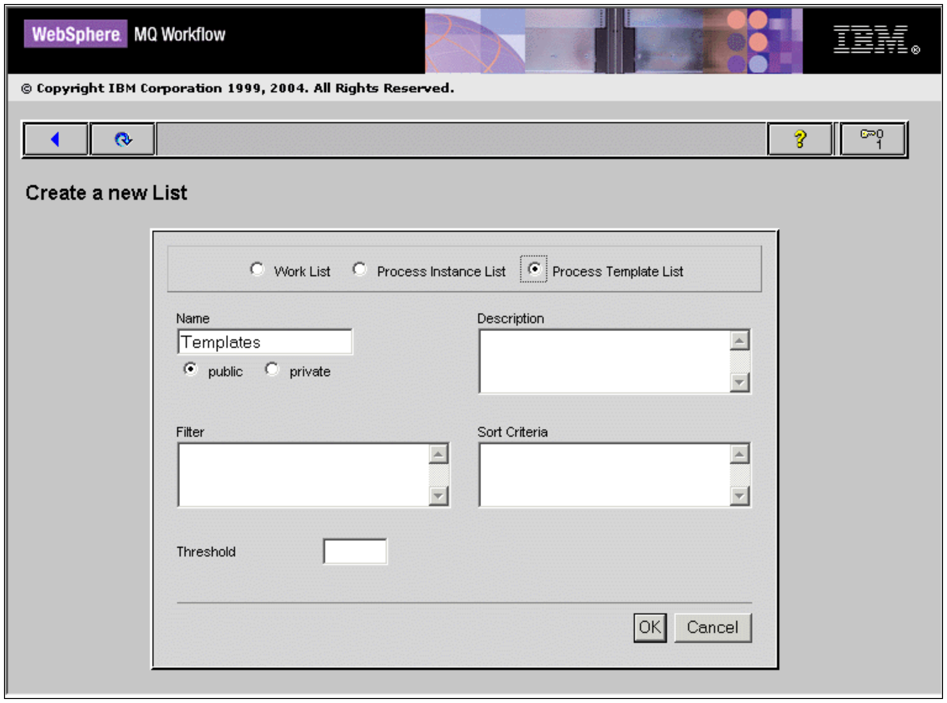
2. After logging on with the user ID `admin` and password `password`, you see the interface shown in Figure 6-23 in which we now create the lists to customize the Web Client interface. Click the icon next to **Create a list**.



IBM WebSphere MQ Workflow Web Client - Main page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://wbiwas/MQWFClient/RTC.html
Go Links
WebSphere MQ Workflow
IBM
© Copyright IBM Corporation 1999, 2004. All Rights Reserved.
Navigate: Select a View
Logged on as ADMIN at FMCGRP
All lists (0)
Action Kind of List Name Owner Description
Create a list...
Create a list...
Done
Local intranet

Figure 6-23 Create a list

3. In the window that popped up, select **Process Template List** and provide a name for the list such as `Templates` (Figure 6-24). Click **OK**.



© Copyright IBM Corporation 1999, 2004. All Rights Reserved.

Create a new List

( ) Work List ( ) Process Instance List [✓ Process Template List]

Name: Templates
[✓ public] ( ) private

Description:

Filter:

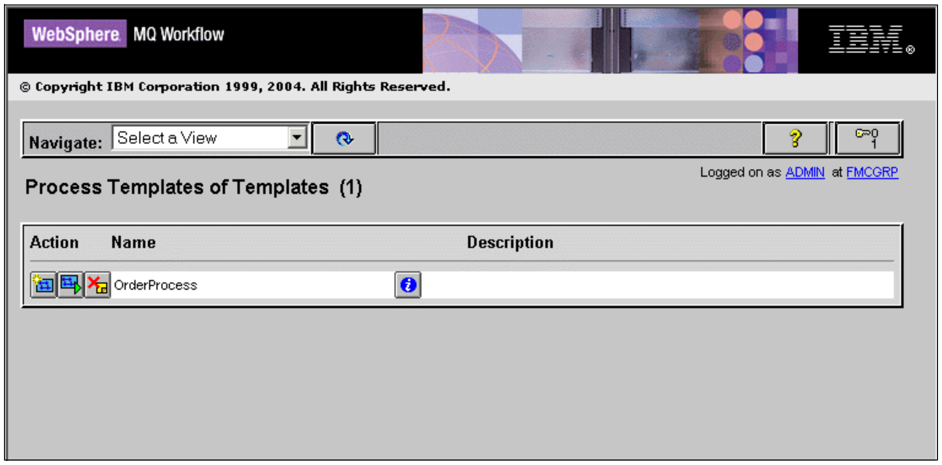
Sort Criteria:

Threshold:

Figure 6-24 Create a template list

As shown in Figure 6-24, this is a public list, meaning that every workflow user can use it. This figure also shows that more advanced features can be used to create filtered and or private lists.

Our list of templates (Figure 6-25) is displayed. The list contains only one process because we have imported only one single process model (OrderProcess).

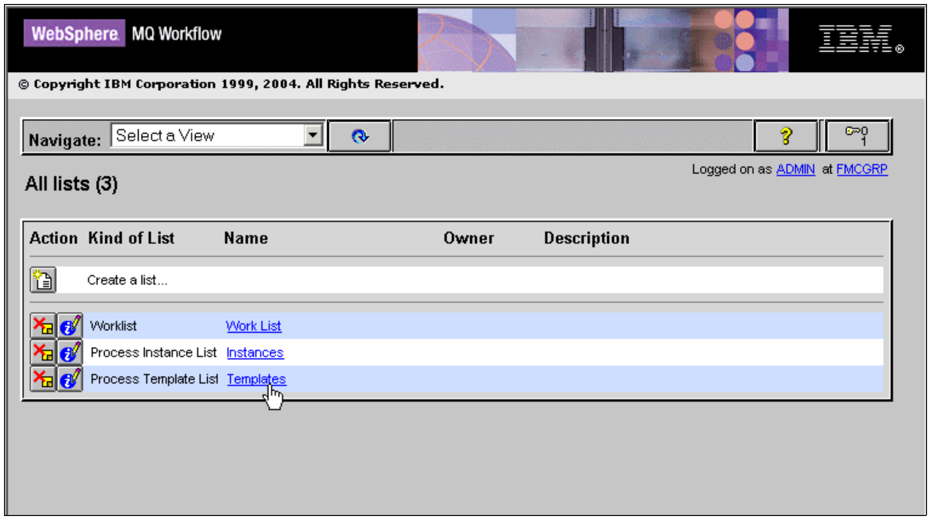

WebSphere MQ Workflow
IBM
© Copyright IBM Corporation 1999, 2004. All Rights Reserved.
Navigate: Select a View
Logged on as ADMIN at FMCGRP

Process Templates of Templates (1)

Action	Name	Description
	OrderProcess	

Figure 6-25 List of templates

4. Follow the same steps to create a list of process instances (we named ours `Instances`) and a list of work items (which we named `Work List`).
5. In the Navigate list, select the view **List of Lists** to generate a list of defined lists (Figure 6-26). Click the **Templates** link.

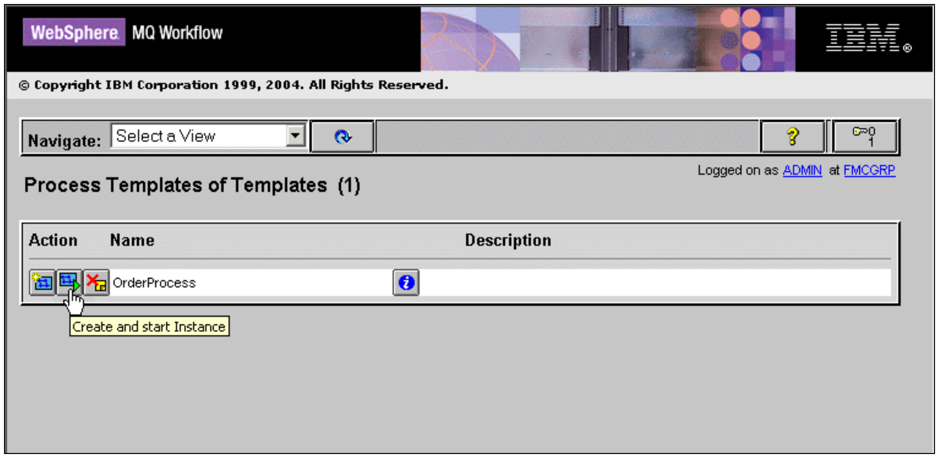

WebSphere MQ Workflow
IBM
© Copyright IBM Corporation 1999, 2004. All Rights Reserved.
Navigate: Select a View
Logged on as ADMIN at FMCGRP

All lists (3)

Action	Kind of List	Name	Owner	Description
	Create a list...			
	Worklist	Work List		
	Process Instance List	Instances		
	Process Template List	Templates		

Figure 6-26 List of lists

6. This brings us back to our list of templates (Figure 6-27). Click the **Create and Start Instance** icon to create an instance and provide it with input data.



© Copyright IBM Corporation 1999, 2004. All Rights Reserved.

Logged on as ADMIN at FMCGRP

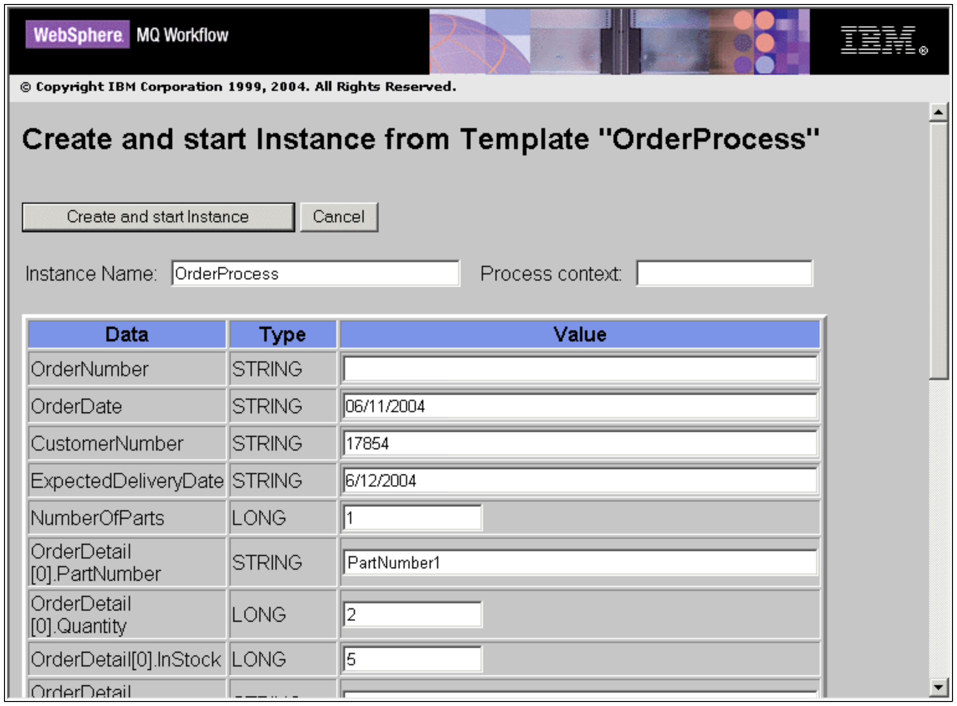
Process Templates of Templates (1)

Action	Name	Description
	OrderProcess	

Create and start Instance

Figure 6-27 Process templates - create and start an instance icon

7. When the Create and start instance page appears (Figure 6-28), enter data for the process and click **Create and start Instance**.


WebSphere MQ Workflow
IBM
© Copyright IBM Corporation 1999, 2004. All Rights Reserved.
Create and start Instance from Template "OrderProcess"
Create and start Instance
Cancel

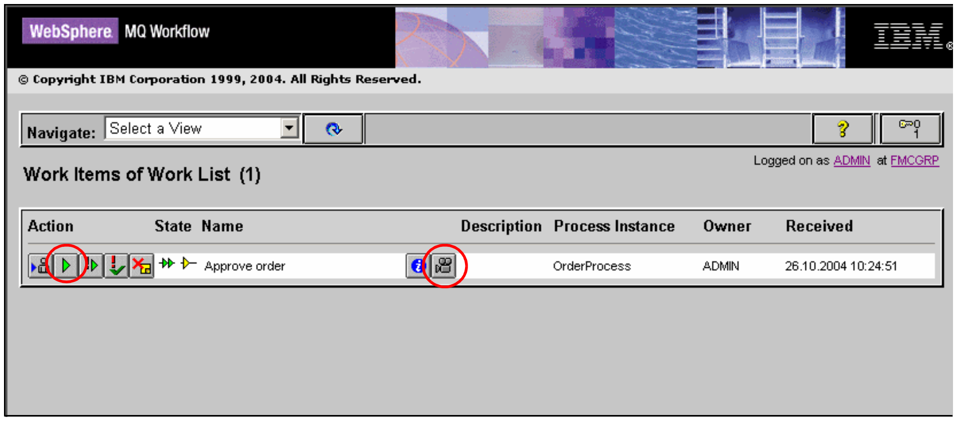
Instance Name: OrderProcess   Process context:

Data	Type	Value
OrderNumber	STRING	
OrderDate	STRING	06/11/2004
CustomerNumber	STRING	17854
ExpectedDeliveryDate	STRING	6/12/2004
NumberOfParts	LONG	1
OrderDetail [0].PartNumber	STRING	PartNumber1
OrderDetail [0].Quantity	LONG	2
OrderDetail[0].InStock	LONG	5
OrderDetail		

Figure 6-28 Create and start instance from template "OrderProcess"

When the instance is created, the first activity usually starts automatically. In any case, this first activity will appear in an individual's work list. Use the Navigate list to switch to the work list, which is shown in Figure 6-29 on page 235.

8. Click the check-out icon (circled in the left of Figure 6-29) to see the output data structure for this activity.



© Copyright IBM Corporation 1999, 2004. All Rights Reserved.

Work Items of Work List (1)

Action	State	Name	Description	Process Instance	Owner	Received
		Approve order		OrderProcess	ADMIN	26.10.2004 10:24:51

Figure 6-29 Work list for user ID admin

By providing data and stepping through the process, you can validate that data is passed along as expected and that conditions are evaluated as expected. At any time, you can access the process monitor interface by selecting the camera icon (circled in the right of Figure 6-29) in the work list.

Figure 6-30 shows an example of a process instance that is now in the Confirm Order step. All other steps have been completed successfully.

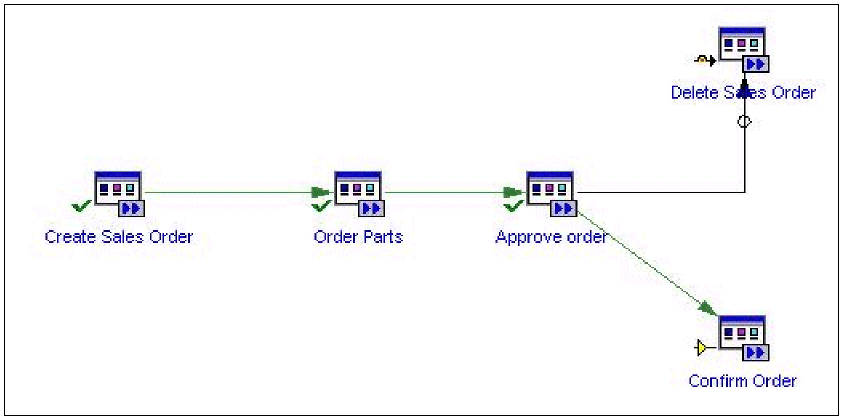


Figure 6-30 Process monitor

By repeating the tests in this section, we can validate the control logic and the mapping of our process model. When the testing is complete, we can return to the Buildtime to update activities that are implemented by other server components, such as the Message Broker and the InterChange Server.

6.5 Update activities to integrate automated activities

Now that the process model is validated, we can make the appropriate activities automated activities instead of user activities.

6.5.1 Create user-defined program execution servers

To create user-defined program execution servers, follow these steps:

1. Within the Buildtime tool, select the **Network** tab to define two UPES objects corresponding with the InterChange Server and the Message Broker.
2. Right-click **FMCSYS** and select **New User-Defined Program Execution Server** (Figure 6-31).

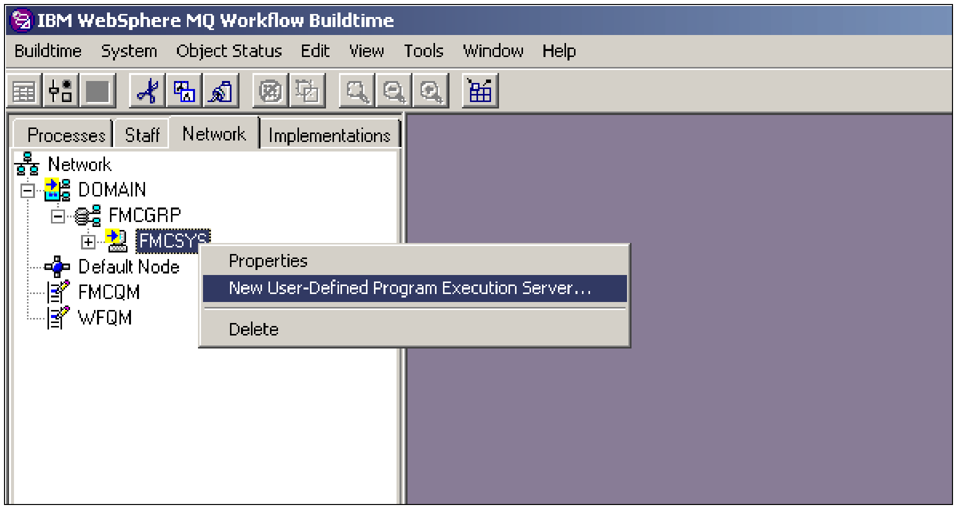
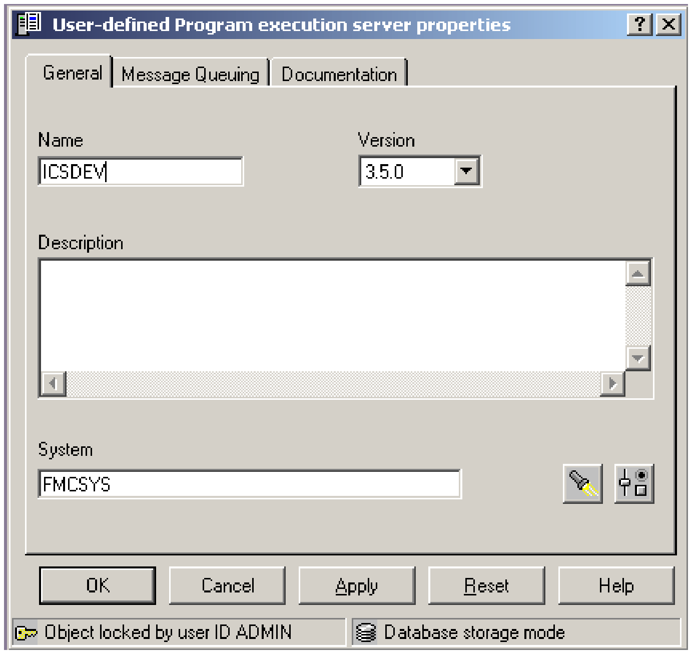

IBM WebSphere MQ Workflow Buildtime
Buildtime System Object Status Edit View Tools Window Help
Processes Staff Network Implementations
Network
DOMAIN
FMCGRP
FMCSYS
Default Node
FMCQM
WFQM
Properties
New User-Defined Program Execution Server...
Delete

Figure 6-31 New UPES

A UPES has a name and points to a remote queue defined on a local queue manager (WFQM).

3. To create the UPES for the InterChange Server, we use the InterChange Server of the development environment. Provide a name for this UPES, for example `ICSDEV`, on the **General** tab (Figure 6-32).


User-defined Program execution server properties

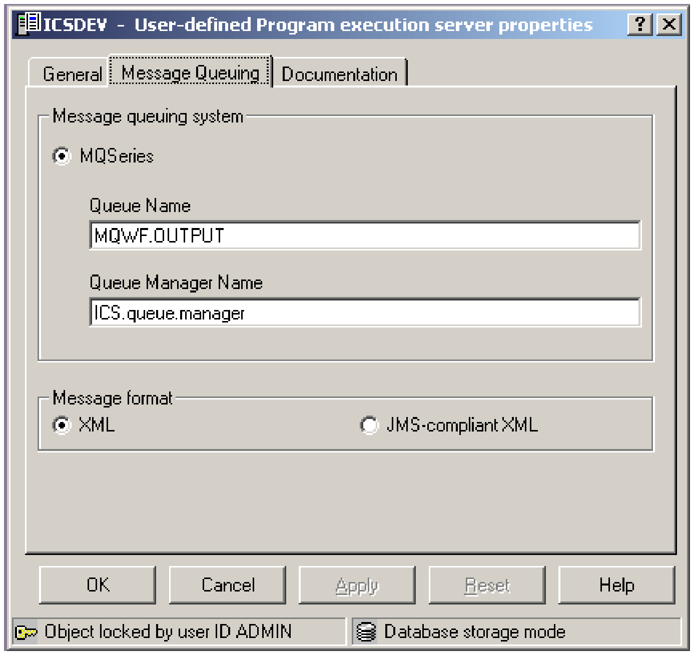
Name: ICSDEV
Version: 3.5.0
Description:
System
FMCSYS

Object locked by user ID ADMIN
Database storage mode

Figure 6-32 The General tab of the UPES ICSDEV

4. On the **Message Queue** tab (Figure 6-33 on page 238), enter the name of the remote queue (`MQWF.OUPUT`), and the local queue manager (`WFQM`). The remote queue defined on WFQM, will point to a local queue (MQWF.INPUT) on the WBIICS server. Defining these WebSphere MQ objects will allow the XML message generated by Workflow to be delivered to the input queue defined on the remote queue manager (`ICS.queue.manager`). Through the use of a WebSphere MQ Workflow connector, WebSphere InterChange Server will pull in the newly delivered message. Refer to Chapter 7, “Sales order management in InterChange Server” on page 249 about the setup of the WebSphere MQ Workflow connector.

Note: For purposes of deploying FDL’s across various development, test, and production environments, the queue manager name can be left blank in the Messaging Queuing tab (Figure 6-33). In doing so, you do not have to edit the FDL every time a promotion occurs. The runtime environment defaults to the WebSphere Workflow queue manager running on the given server.


ICSDEV - User-defined Program execution server properties

Message queuing system

(●) MQSeries

Queue Name: MQWF.OUTPUT

Queue Manager Name: ICS.queue.manager

Message format

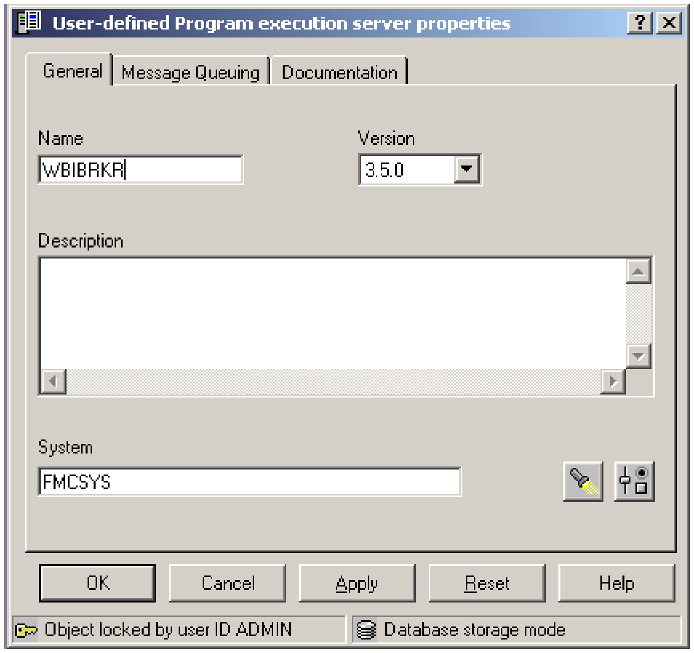
(●) XML ( ) JMS-compliant XML

Object locked by user ID ADMIN

Database storage mode

Figure 6-33 The Message Queuing tab for the UPES ICSDEV

5. Repeat the same steps to create a UPES that represents the broker (Figure 6-34 on page 239).



Name

WBIBRKR

Version

3.5.0

Description

System

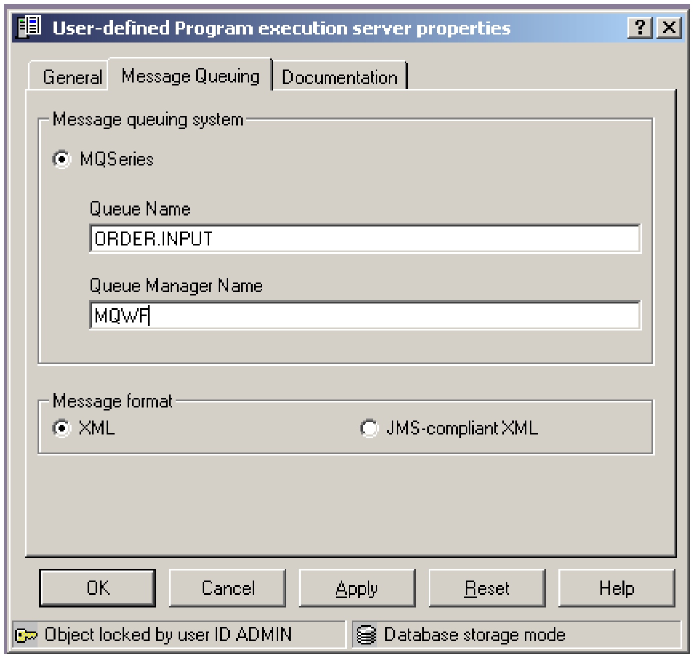
FMCSYS

Object locked by user ID ADMIN

Database storage mode

Figure 6-34 The General tab of the UPES WBIBRKR

6. Unlike ICS.queue.manager, the Message Broker queue manager is in a queue manager cluster with the Workflow queue manager. So, in Figure 6-35 on page 240, the queue, `ORDER.INPUT` (a shared queue), acts as the input queue for a message flow. The name of the queue manager is `BKQM`. Refer to Chapter 8, “Replenishing parts in WebSphere BI Message Broker” on page 377 for the details about this message flow.



Message queuing system

(● MQSeries)

Queue Name: ORDER.INPUT

Queue Manager Name: MQWF

Message format: [✓ XML] JMS-compliant XML

Object locked by user ID ADMIN | Database storage mode

Figure 6-35 The Message Queuing tab of the UPES WBIBRKR

6.5.2 Create program objects for InterChange Server

The WebSphere MQ Workflow connector requires that specific parameters be passed to it through the program object. Therefore, we must create three new program objects:

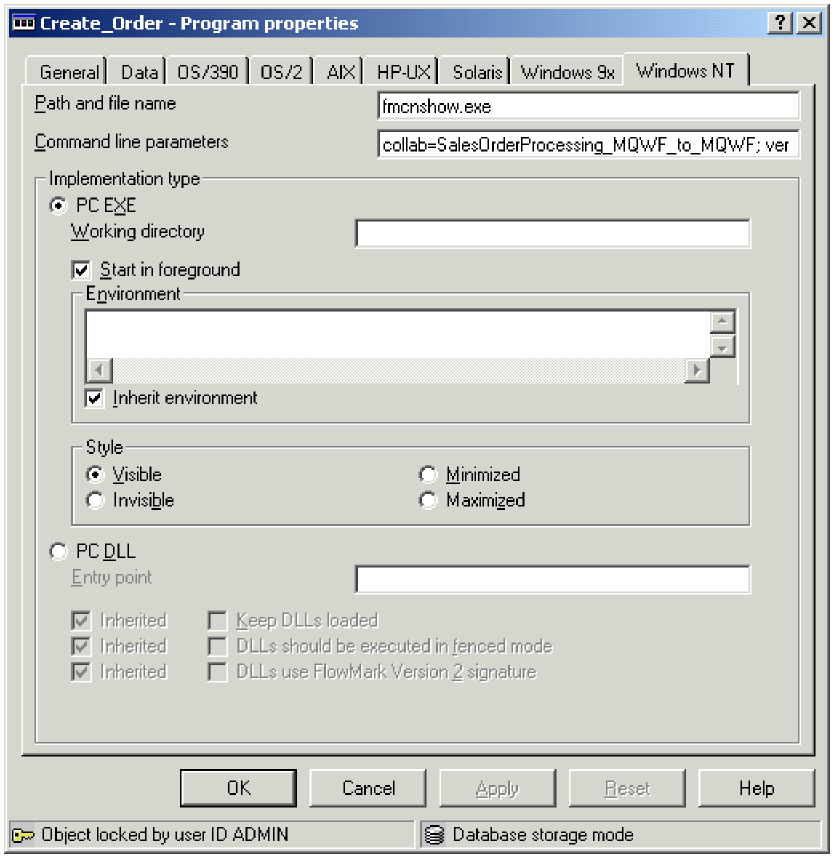
- Create_Order
- Delete_Order
- Confirm_Order

Follow similar steps to those used to create the dummy program object:

1. As shown in Figure 6-36 on page 241, the command line requires specific parameters. The command line string is:

   ```
   collab=SalesOrderProcessing_MQWF_to_MQWF; verb=Create
   ```

   This string is used to tell the WebSphere MQ Workflow connector what collaboration to use and what verb to use to process the incoming message. Note that the name of the business object is derived directly from the name of the data structure.


Create_Order - Program properties

General | Data | OS/390 | OS/2 | AIX | HP-UX | Solaris | Windows 9x | Windows NT

Path and file name: fmcnshow.exe

Command line parameters: collab=SalesOrderProcessing_MQWF_to_MQWF; ver

Implementation type

[✓ PC EXE]

Working directory:

[✓] Start in foreground

Environment

[✓] Inherit environment

Style

[✓ Visible] Minimized

Invisible Maximized

PC DLL

Entry point

[✓] Inherited [ ] Keep DLLs loaded

[✓] Inherited [ ] DLLs should be executed in fenced mode

[✓] Inherited [ ] DLLs use FlowMark Version 2 signature

Object locked by user ID ADMIN

Database storage mode

Figure 6-36 Details of the Create_Order program

2. Create the Delete_Order program using the command line string:

```
collab=SalesOrderProcessing_MQWF_to_MQWF; verb=Delete
```

3. For the program Confirm_Order, the command line string is:

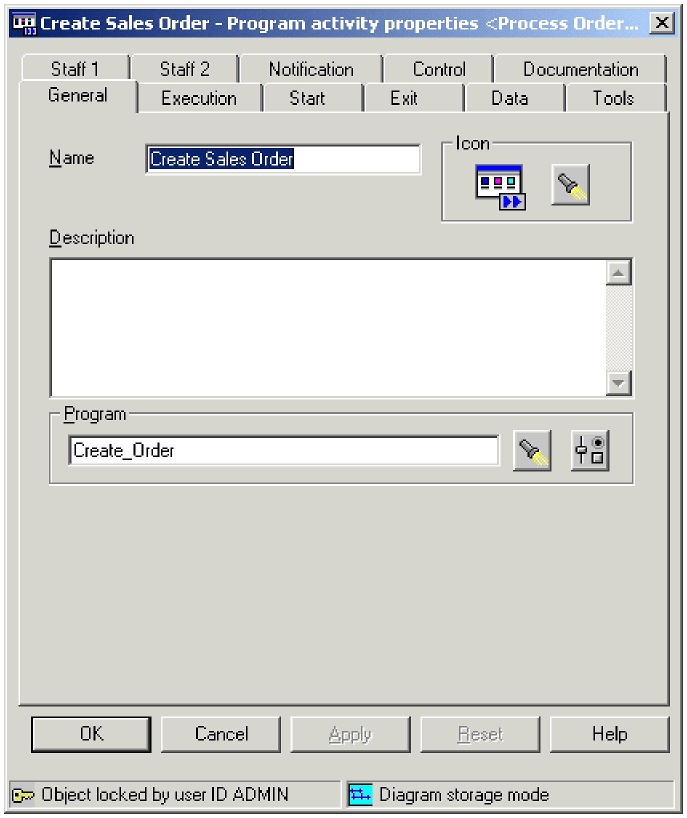
```
collab=SalesOrderProcessing_MQWF_to_MQWF; verb=Update
```

There are no specific requirements for the broker, so we can continue to use the dummy program for the Order Parts activity.

6.5.3 Update activities in process diagram

With new programs defined, we can return to the process diagram and link the automated activities to the corresponding program objects:

1. Open the properties of the Create Sales Order activity. On the **General** tab, update the name of the program to point to the Create_Order program (Figure 6-37). You can use the flashlight icon to pick the program from a pop-up window.



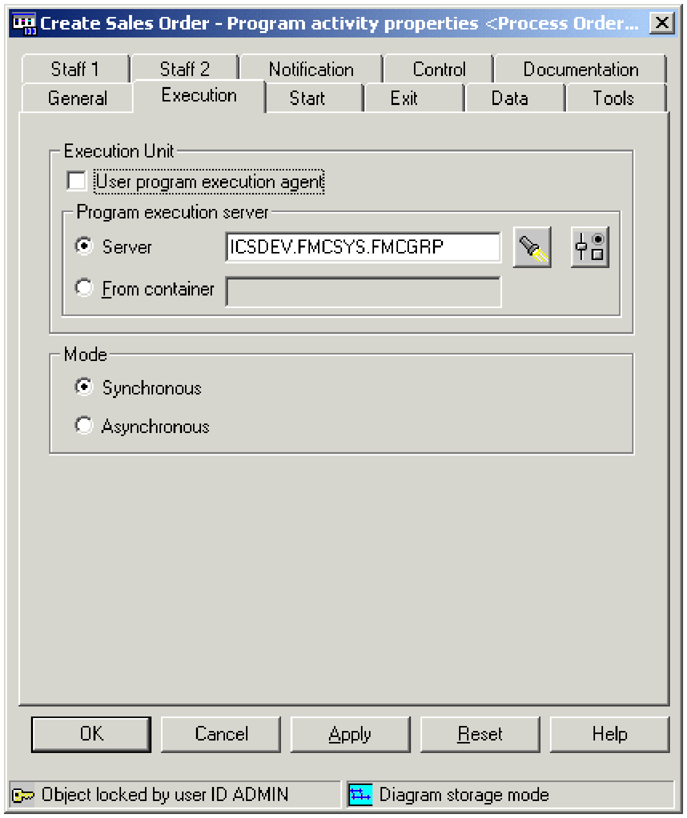
Name: Create Sales Order

Program: Create_Order

Figure 6-37 Update Create Sales Order activity

2. In the **Execution** tab (Figure 6-38), unselect **User program execution agent**.

   Because we expect a reply from the InterChange Server about the new sales order, we should select **Synchronous**. This means that this activity will keep running and waiting for a response. However, the actual runtime engine will not wait for a response; it will switch to execute another process instance if one is available. When the actual reply message arrives, this waiting process is reactivated.



Execution Unit

☐ User program execution agent

Program execution server

● Server ICSDEV.FMCSYS.FMCGRP

○ From container

Mode

● Synchronous

○ Asynchronous

Object locked by user ID ADMIN

Diagram storage mode

Figure 6-38 Update activity Create Sales Order

Also, the underlying mechanism for passing the request message from WebSphere MQ Workflow to the UPES and to pass the response back to WebSphere MQ Workflow is implemented by message queuing, which is clearly not a synchronous mechanism.

Tip: You can click the **flashlight icon** to pick the correct UPES from the pop-up window shown in Figure 6-39 on page 244.

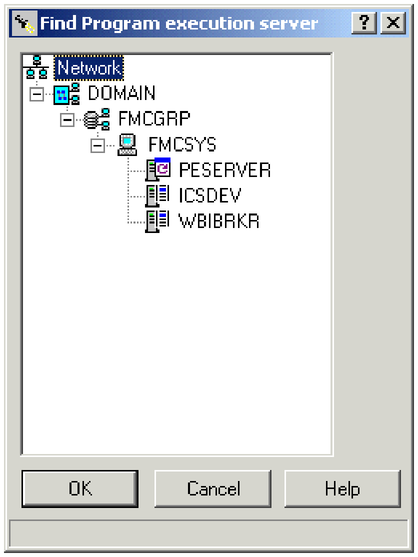
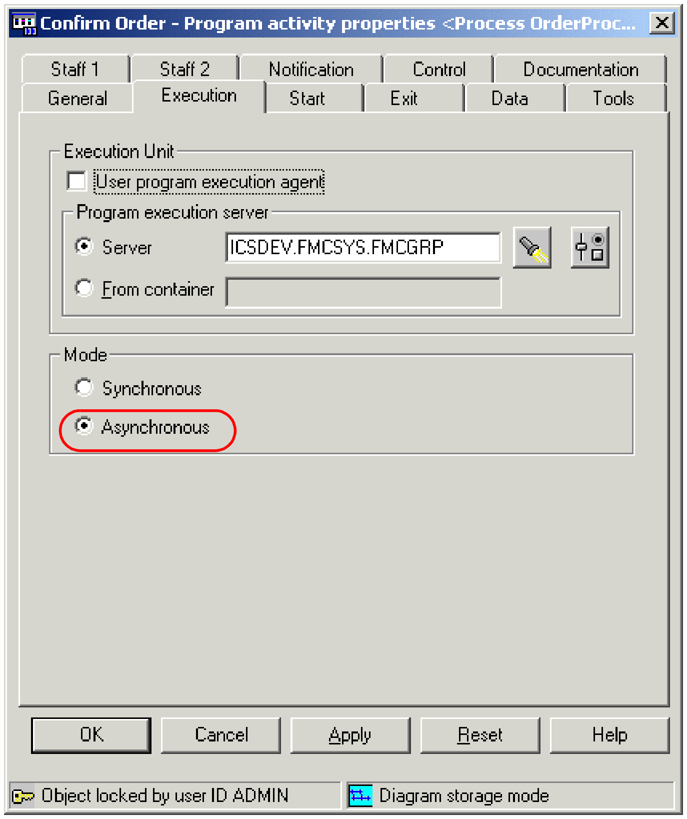


Figure 6-39 Pick the correct UPES

3. The Order Parts activity only has to be updated on the **Execution** tab. Select the UPES **WBIBRKR** and ensure that Mode is set to **Synchronous**, as we expect a response from the broker about the ordering of parts.
4. The Delete Order and Confirm Order activities have their respective programs on the **General** tab. For the **Execution** tab, they point to the same UPES that was used for the Create Sales Order activity.

However, the Mode is different, as the process instance does not have to wait for the completion of these activities. Select **Asynchronous** for both activities (Figure 6-40).



Execution Unit

☐ User program execution agent

Program execution server

● Server: ICSDEV.FMCSYS.FMCGRP

○ From container:

Mode

○ Synchronous

● Asynchronous

Object locked by user ID ADMIN

Diagram storage mode

Figure 6-40 Asynchronous mode for Confirm and Delete Order activities

When all of these changes are completed, we can again export the process model and reimport it into the runtime environment. Make sure that the new UPES objects and the new program objects are exported.

The import into the runtime environment is made in the same way as before.

The UPES objects rely on MQ definitions, so we must define these objects. Detailed instructions about the definitions for the setup of the ICSDEV UPES are included in Chapter 7, “Sales order management in InterChange Server” on page 249 for the setup of the ICSDEV UPES. Refer to Chapter 8, “Replenishing parts in WebSphere BI Message Broker” on page 377 for details about MQ definitions to support the use of the WBIBRKR UPES. In short, these instructions help define:

- Sender and receiver channels between the Workflow queue manager on one side and the InterChange Server and Message Broker queue managers on the other side
- Transmission queues that are used to store messages before transmission
- Remote queue objects
- Local queue objects

When this setup is in place, we can use the WebSphere MQ Workflow Web Client to create and start an instance. When the instance is started, an MQ message is sent for pick-up by the WebSphere MQ Workflow connector. This component has not yet been created, so no response will come. The activity remains in a running state, which can be seen by looking at the process monitor for this instance. In Figure 6-41, the double green triangle icons indicate that Create Sales Order is running.

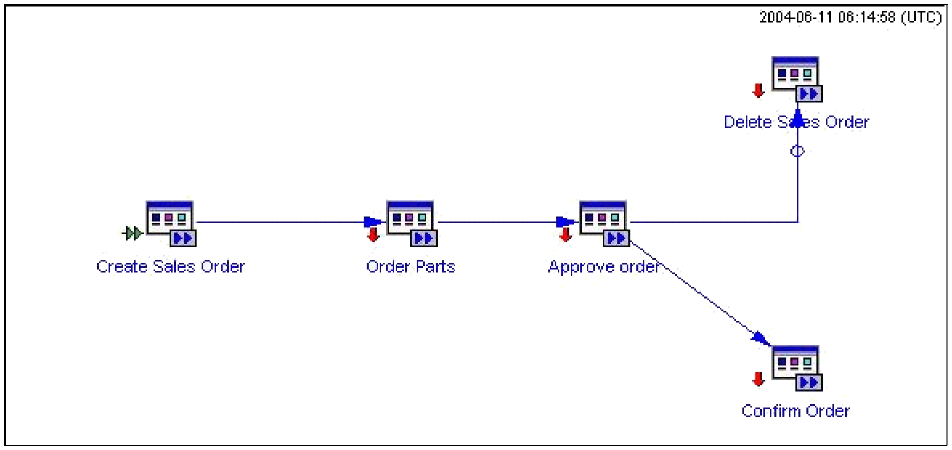


Figure 6-41 Process monitor view with a running UPES activity

When browsing the MQWF.INPUT queue on ICS.queue.manager, we can see the structure of the message that is waiting for processing by the WebSphere MQ Workflow connector (Example 6-1).

Example 6-1 Sample XML message for the WebSphere MQ Workflow connector

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<WfMessage>
 <WfMessageHeader>
  <ResponseRequired>Yes</ResponseRequired>
 </WfMessageHeader>
 <ActivityImplInvoke>

<ActImplCorrelID>RUEAAAABAAQAGgAAAAAAAAAAAAAAAwAAAAEABQAAAAAAAAAAAAAAAAADQQAAAAEABAAbAAAAAAAAAA
BF</ActImplCorrelID>
  <Starter>ADMIN</Starter>
  <ProgramID>
   <ProcTemplID>AAAAAQAFAAAAAAAAAAAAAA==</ProcTemplID>
   <ProgramName>Create_Order</ProgramName>
  </ProgramID>
  <ImplementationData>
   <ImplementationPlatform>WindowsNT</ImplementationPlatform>
   <ProgramParameters>collab=SalesOrderProcessing_MQWF_to_MQWF; verb=Create</ProgramParameters>
   <ExeOptions>
    <PathAndFileName>fmcnshow.exe</PathAndFileName>
    <InheritEnvironment>true</InheritEnvironment>
    <StartInForeGround>true</StartInForeGround>
    <WindowStyle>Visible</WindowStyle>
   </ExeOptions>
  </ImplementationData>
  <ProgramInputData>
   <_ACTIVITY>Create Sales Order</_ACTIVITY>
   <_PROCESS>OrderProcess$AAAAAQAEABoAAAAAAAAAAA==</_PROCESS>
   <_PROCESS_MODEL>OrderProcess</_PROCESS_MODEL>
   <Order_Form>
    <OrderNumber></OrderNumber>
    <OrderDate>06/11/2004</OrderDate>
    <CustomerNumber>17854</CustomerNumber>
    <ExpectedDeliveryDate>06/12/2004</ExpectedDeliveryDate>
    <NumberOfParts>1</NumberOfParts>
    <OrderDetail>
     <PartNumber>PartNumber1</PartNumber>
     <Quantity>2</Quantity>
     <InStock>5</InStock>
    </OrderDetail>
    <OrderDetail>
     <PartNumber></PartNumber>
    </OrderDetail>
    <OrderDetail>
```

```
     <PartNumber></PartNumber>
    </OrderDetail>
    <OrderDetail>
     <PartNumber></PartNumber>
    </OrderDetail>
    <OrderDetail>
     <PartNumber></PartNumber>
    </OrderDetail>
    <WorkOrderNumber></WorkOrderNumber>
    <OrderStatus></OrderStatus>
   </Order_Form>
  </ProgramInputData>
  <ProgramOutputDataDefaults>
   <_ACTIVITY>Create Sales Order</_ACTIVITY>
   <_PROCESS>OrderProcess$AAAAAQAEABoAAAAAAAAAAA==</_PROCESS>
   <_PROCESS_MODEL>OrderProcess</_PROCESS_MODEL>
   <Order_Form>       </Order_Form>
  </ProgramOutputDataDefaults>
 </ActivityImplInvoke>
</WfMessage>
```

In the next chapter, we describe the development of the sales order collaboration and the connector configuration, so that this message can be processed and a response can be sent for the workflow server.

7

Sales order management in InterChange Server

This chapter discusses the implementation of the required artifacts to complete the InterChange Server portion of this overall integration solution. This involves the usage of a prebuilt Sales Order Management Collaboration and two WebSphere Business Integration Adapters, namely the WebSphere Business Integration Adapter for WebSphere MQ Workflow and the WebSphere Business Integration Adapter for JDBC.

We start by defining the overall interaction between various integration components. This includes WebSphere MQ Workflow, WebSphere Business Integration Adapters, WebSphere InterChange Server, and the backend Order DB2 database.

7.1 Introduction

In this chapter, we explain all aspects for the scenario that are related to creating, deploying and testing the integration artifacts for the InterChange Server. The heart of the InterChange Server artifacts is the Sales Order Management Collaboration that uses business logic to choreograph the interactions with our source and target systems. This collaboration object is instantiated by certain activities of the workflow process.

To simulate our target backend system, we used a simple database. (See 7.3, "Application database ORDERMGT" on page 256.) Our Sales Order Management Collaboration listens on its *From Port* for the delivery of an Order Generic Business Object (GBO). The MQ Workflow Adapter originally receives the WebSphere MQ Workflow message. This XML message is parsed to the Application-Specific Business Object (ASBO) by the MQ Workflow Adapter and mapped to the GBO, which is passed to the collaboration. In order to pass the GBO to our backend, as directed by the collaboration logic, the InterChange Server converts it to the corresponding database ASBO and sends it to the JDBC Adapter.

The response message follows the inverse path, to return the response message to the WebSphere MQ Workflow calling activity. Figure 7-1 shows this interaction schematically.

Note: Throughout this book the words *adapters* and *connectors* are used synonymously to reference the WebSphere Business Integration Adapters.

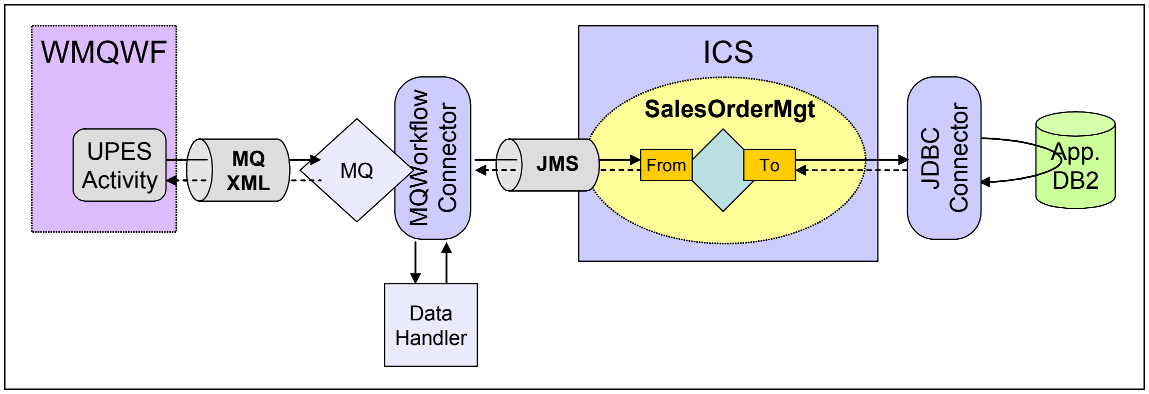


Figure 7-1 Overview interaction between WebSphere MQ Workflow and InterChange Server

7.2 Scenario implementation overview

In our scenario, there are three different MQ Workflow activities that call the InterChange Server. Those MQ Workflow activity names are:

- Create_Order
- Confirm_Order
- Delete_Order

Create_Order activity uses the MQ Workflow data structure called Order_Form, while both the Confirm_Order and Delete_Order activities share the same MQ Workflow data structure called Completed_Order_Form. The InterChange Server defines these data structures as ASBOs.

The Create_Order activity directs the InterChange Server to create a record in the ORDER table and ITEMS table. Confirm_Order changes the status of the existing ORDER record that was created in the original Create_Order activity, while Delete_Order performs a soft delete of the original record by changing the status field to `Rejected` status.

WebSphere MQ Workflow communicates with the InterChange Server by placing XML messages on a predefined queue that are in the form of the corresponding data structure, namely Order_Form or Completed_Order_Form. Example 7-1 on page 252 and Example 7-2 on page 254 are examples of those XML input message from MQ Workflow for the two data structures. Notice that the Create_Order activity is synchronous, while the Confirm_Order and Delete_Order activities are each asynchronous and thus do not require a response from the InterChange Server.

Example 7-1   Sample input message from MQ Workflow for Create_Order activity

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<WfMessage>
 <WfMessageHeader>
  <ResponseRequired>Yes</ResponseRequired>
 </WfMessageHeader>
 <ActivityImplInvoke>

<ActImplCorrelID>RUEAAAABAA1ABAAAAAAAAAAAAAAAAwAAAAEADAAAAAAAAAAAAAAAAAADQQAAAA
EADUAFAAAAAAAAAABF</ActImplCorrelID>
  <Starter>ADMIN</Starter>
  <ProgramID>
   <ProcTemplID>AAAAAQAMAAAAAAAAAAAAAA==</ProcTemplID>
   <ProgramName>Create_Order</ProgramName>
  </ProgramID>
  <ImplementationData>
   <ImplementationPlatform>WindowsNT</ImplementationPlatform>
   <ProgramParameters>collab=SalesOrderProcessing_MQWF_to_MQWF;
verb=Create</ProgramParameters>
   <ExeOptions>
    <PathAndFileName>fmcnshow.exe</PathAndFileName>
    <InheritEnvironment>true</InheritEnvironment>
    <StartInForeGround>true</StartInForeGround>
    <WindowStyle>Visible</WindowStyle>
   </ExeOptions>
  </ImplementationData>
  <ProgramInputData>
   <_ACTIVITY>Create Sales Order</_ACTIVITY>
   <_PROCESS>OrderProcess$AAAAAQANQAQAAAAAAAAAAA==</_PROCESS>
   <_PROCESS_MODEL>OrderProcess</_PROCESS_MODEL>
   <Order_Form>
    <OrderNumber></OrderNumber>
    <OrderDate>10/20/2004</OrderDate>
    <CustomerNumber>TRAVIS001</CustomerNumber>
    <ExpectedDeliveryDate>10/30/2004</ExpectedDeliveryDate>
    <NumberOfParts>5</NumberOfParts>
    <OrderDetail>
     <PartNumber>PartNumber1</PartNumber>
     <Quantity>22</Quantity>
     <InStock>35</InStock>
    </OrderDetail>
    <OrderDetail>
     <PartNumber>PartNumber2</PartNumber>
     <Quantity>21</Quantity>
     <InStock>52</InStock>
    </OrderDetail>
    <OrderDetail>
     <PartNumber>PartNumber3</PartNumber>
     <Quantity>12</Quantity>
     <InStock>15</InStock>
    </OrderDetail>
    <OrderDetail>
     <PartNumber>PartNumber4</PartNumber>
     <Quantity>2</Quantity>
     <InStock>5</InStock>
    </OrderDetail>
    <OrderDetail>
     <PartNumber>PartNumber5</PartNumber>
     <Quantity>21</Quantity>
     <InStock>54</InStock>
    </OrderDetail>
    <WorkOrderNumber></WorkOrderNumber>
    <OrderStatus></OrderStatus>
   </Order_Form>
  </ProgramInputData>
  <ProgramOutputDataDefaults>
   <_ACTIVITY>Create Sales Order</_ACTIVITY>
   <_PROCESS>OrderProcess$AAAAAQANQAQAAAAAAAAAAA==</_PROCESS>
   <_PROCESS_MODEL>OrderProcess</_PROCESS_MODEL>
   <Order_Form>       </Order_Form>
  </ProgramOutputDataDefaults>
 </ActivityImplInvoke>
</WfMessage>
```

Example 7-2 Sample input message from MQWorkflow for Confirm_Order activity

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<WfMessage>
  <WfMessageHeader>
    <ResponseRequired>No</ResponseRequired>
  </WfMessageHeader>
  <ActivityImplInvoke>

<ActImplCorrelID>RUEAAAABABDAAgAAAAAAAAAAAAAAAgAAAAEAD8AAAAAAAAAAAAAAAAACQQAAAA
EAEIAGAAAAAAAAAABF</ActImplCorrelID>
    <Starter>ADMIN</Starter>
    <ProgramID>
      <ProcTemplID>AAAAAQAPwAAAAAAAAAAAAA==</ProcTemplID>
      <ProgramName>Confirm_Order</ProgramName>
    </ProgramID>
    <ImplementationData>
      <ImplementationPlatform>WindowsNT</ImplementationPlatform>
      <ProgramParameters>collab=salesOrderProcessing_MQWF_to_MQWF;
verb=Update</ProgramParameters>
      <ExeOptions>
        <PathAndFileName>fmcnshow.exe</PathAndFileName>
        <InheritEnvironment>true</InheritEnvironment>
        <StartInForeGround>true</StartInForeGround>
        <WindowStyle>Visible</WindowStyle>
      </ExeOptions>
    </ImplementationData>
    <ProgramInputData>
      <_ACTIVITY>Confirm Order</_ACTIVITY>
      <_PROCESS>OrderProcess$AAAAAQAQwAIAAAAAAAAAAA==</_PROCESS>
      <_PROCESS_MODEL>OrderProcess</_PROCESS_MODEL>
      <Completed_Order_Form>
        <OrderNumber>78945620041021160735</OrderNumber>
        <OrderDate>10/21/2004</OrderDate>
        <CustomerNumber>789456</CustomerNumber>
        <ExpectedDeliveryDate>12/21/2004</ExpectedDeliveryDate>
        <NumberOfParts>5</NumberOfParts>
        <OrderDetail>
          <PartOrder>
            <PartNumber>L1</PartNumber>
            <Quantity>1</Quantity>
            <InStock>2</InStock>
          </PartOrder>
          <Supplier></Supplier>
          <OrderReference></OrderReference>
          <ExpectedDate>2004-10-28</ExpectedDate>
          <ShipmentDate>2004-11-04</ShipmentDate>
          <CaseNumber>12</CaseNumber>
          <ReceptionDate></ReceptionDate>
    <RegistrationDate></RegistrationDate>
  </OrderDetail>
  <OrderDetail>
    <PartOrder>
      <PartNumber>L2</PartNumber>
      <Quantity>1</Quantity>
      <InStock>2</InStock>
    </PartOrder>
    <Supplier></Supplier>
    <OrderReference></OrderReference>
    <ExpectedDate>2004-10-28</ExpectedDate>
    <ShipmentDate>2004-11-04</ShipmentDate>
    <CaseNumber>12</CaseNumber>
    <ReceptionDate></ReceptionDate>
    <RegistrationDate></RegistrationDate>
  </OrderDetail>
  <OrderDetail>
    <PartOrder>
      <PartNumber>L3</PartNumber>
      <Quantity>1</Quantity>
      <InStock>2</InStock>
    </PartOrder>
    <Supplier></Supplier>
    <OrderReference></OrderReference>
    <ExpectedDate>2004-10-28</ExpectedDate>
    <ShipmentDate>2004-11-04</ShipmentDate>
    <CaseNumber>12</CaseNumber>
    <ReceptionDate></ReceptionDate>
    <RegistrationDate></RegistrationDate>
  </OrderDetail>
  <OrderDetail>
    <PartOrder>
      <PartNumber>L4</PartNumber>
      <Quantity>1</Quantity>
      <InStock>2</InStock>
    </PartOrder>
    <Supplier></Supplier>
    <OrderReference></OrderReference>
    <ExpectedDate>2004-10-28</ExpectedDate>
    <ShipmentDate>2004-11-04</ShipmentDate>
    <CaseNumber>12</CaseNumber>
    <ReceptionDate></ReceptionDate>
    <RegistrationDate></RegistrationDate>
  </OrderDetail>
  <OrderDetail>
    <PartOrder>
      <PartNumber>L5</PartNumber>
      <Quantity>1</Quantity>
      <InStock>2</InStock>
```

```
          </PartOrder>
          <Supplier></Supplier>
          <OrderReference></OrderReference>
          <ExpectedDate>2004-10-28</ExpectedDate>
          <ShipmentDate>2004-11-04</ShipmentDate>
          <CaseNumber>12</CaseNumber>
          <ReceptionDate></ReceptionDate>
          <RegistrationDate></RegistrationDate>
        </OrderDetail>
        <WorkOrderNumber></WorkOrderNumber>
        <RejectOrder>1</RejectOrder>
        <OrderStatus>Confirmed</OrderStatus>
      </Completed_Order_Form>
    </ProgramInputData>
    <ProgramOutputDataDefaults>
      <_ACTIVITY>Confirm Order</_ACTIVITY>
      <_PROCESS>OrderProcess$AAAAAQAQwAIAAAAAAAAAAA==</_PROCESS>
      <_PROCESS_MODEL>OrderProcess</_PROCESS_MODEL>
      <Completed_Order_Form>
      </Completed_Order_Form>
    </ProgramOutputDataDefaults>
  </ActivityImplInvoke>
</WfMessage>
```

7.3 Application database ORDERMGT

Before we start developing the different components, consider the database application for storing the sales orders. The backend system is a simple database that contains two tables, ORDERS (Table 7-1) and ITEMS (Table 7-2 on page 257).

Table 7-1 ORDERS table structure

Name	Type	Primary key	Others
ORDER_ID	INTEGER	Yes	Self-generated
CUSTOMER_ID	VARCHAR(50)	No	
STATUS	VARCHAR(10)	No	
ORDER_DATE	VARCHAR(10)	No	
PO_NUMBER	VARCHAR(10)	No	
DELIVERY_DATE	VARCHAR(10)	No	

Name	Type	Primary key	Others
REJECT_ORDER	VARCHAR(10)	No	
NUM_PARTS	VARCHAR(10)	No	
ORDER_NUM	VARCHAR(50)	No	

Table 7-2 ITEMS table structure

Name	Type	Primary key	Foreign key
ORDER_ID	INTEGER	No	Yes
PART_CODE	VARCHAR(50)	No	No
QUANTITY	INTEGER	No	No
ITEM_ID	INTEGER	Yes	No
IN_STOCK	INTEGER	No	No

1. Create the database and assign the correct privileges to access it. The database is called `ORDERMGT`, and we grant the user `wbiadmin` to access it. Enter the commands shown in Example 7-3 to create the database in a DB2 command window, or use the DB2 Control Center.

   Attention: The wbiadmin user will be used when we configure the JDBC adapter in 7.7.3, "JDBC adapter configuration" on page 325.

Example 7-3 Create ORDERMGT database and assign user privileges

```
ATTACH TO DB2 user db2admin using sab414r
CREATE DATABASE ORDERMGT;
GRANT  DBADM,CREATETAB,BINDADD,CONNECT,CREATE_NOT_FENCED_ROUTINE,
IMPLICIT_SCHEMA,LOAD,CREATE_EXTERNAL_ROUTINE,QUIESCE_CONNECT ON DATABASE  TO
USER WBIADMIN;
```

2. We must catalog this new database to access it. Execute the following command in a DB2 command window:

   ```
   catalog tcpip node WBIDB remote WBIDB server 50000
   catalog database ORDERMGT at node WBIDB
   ```

   The first command registers the remote DB2 instance, and the second command registers the remote database on that remote node.

3. Example 7-4 on page 258 shows the script for creating the database tables. You could also use the DB2 Control Center, using the structure defined in Table 7-1 and Table 7-2.

Example 7-4 Scripts to create the ORDERS and ITEMS tables

```
CONNECT TO ORDERMGT USER wbiadmin USING sab414r

CREATE TABLE ORDERS (ORDER_ID INTEGER NOT NULL GENERATED ALWAYS AS IDENTITY
(START WITH 1, INCREMENT BY 1, NO CACHE), CUSTOMER_ID VARCHAR(50) NOT NULL,
STATUS VARCHAR(10), ORDER_DATE VARCHAR(10),PO_NUMBER VARCHAR(10), DELIVERY_DATE
VARCHAR(10), REJECT_ORDER VARCHAR(10), NUM_PARTS VARCHAR(10), ORDER_NUM
VARCHAR(50), PRIMARY KEY(ORDER_ID))

CREATE TABLE ITEMS (ORDER_ID INTEGER, PART_CODE VARCHAR(50) NOT NULL, QUANTITY
INTEGER NOT NULL, ITEM_ID INTEGER NOT NULL GENERATED ALWAYS AS IDENTITY (START
WITH 1, INCREMENT BY 1, NO CACHE), IN_STOCK INTEGER, PRIMARY KEY(ITEM_ID),
FOREIGN KEY(ORDER_ID) REFERENCES ORDERS(ORDER_ID) ON DELETE CASCADE ON UPDATE
RESTRICT)
```

7.4 Preparing Development Environment

In preparing our development environment for the InterChange Server we were conscious to the fact that by using a prebuilt industry collaboration, our overall development time would be greatly reduced. As a result, the majority of our development effort would be spent developing transformation maps. Our initial work was to draw out and document these mapping interactions as shown in Figure 7-2.

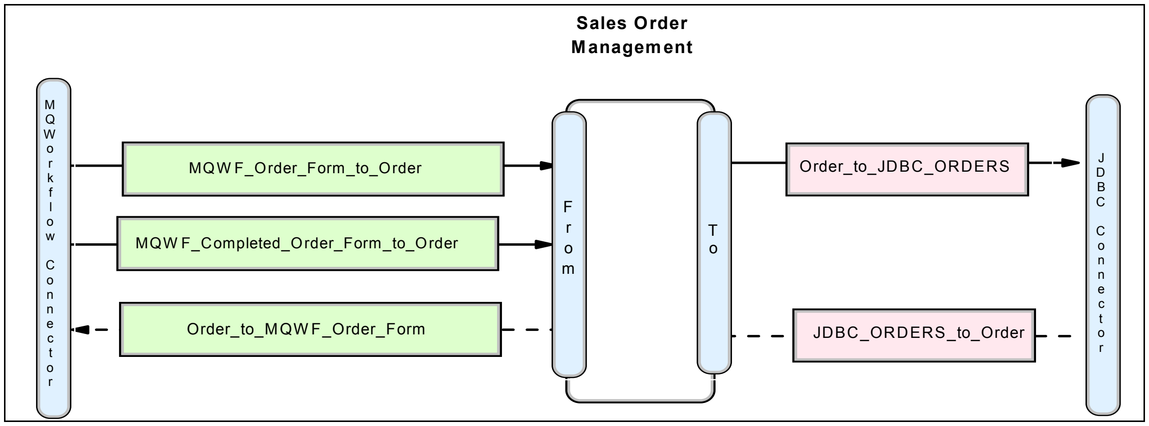


Figure 7-2 Processing in the InterChange Server

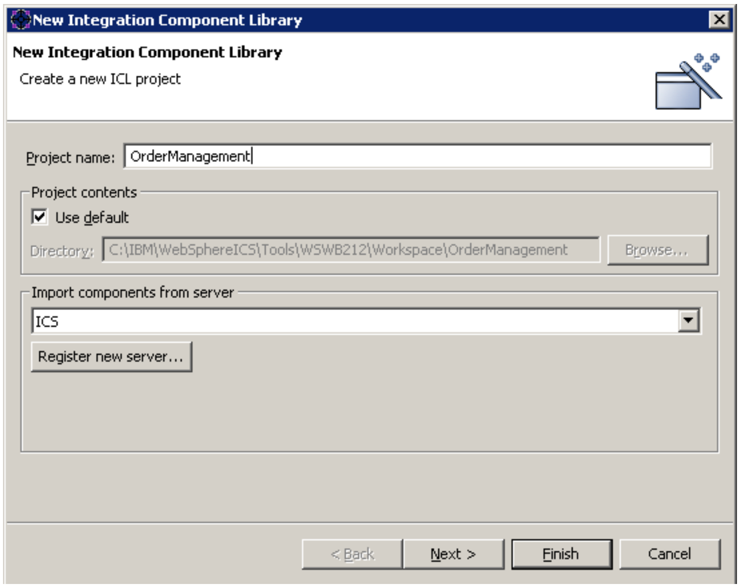
The development steps for this integration solution are as follows:

1. Generate ASBOs for the database schema using the JDBC Object Discovery Agent.
2. Create maps between the JDBC business object and the generic business object.
3. Generate ASBOs for the MQ Workflow process using the fdlborgen utility.
4. Create maps between the MQ Workflow business object and the generic business object.
5. Configure and test the WebSphere MQ Workflow connector.
6. Configure and test the JDBC connector.
7. Create the collaboration object from the pre-built collaboration template.

The main development environment and tools for the WebSphere InterChange Server V4.3 are accessed through the System Manager. Typically, all of these solution components are stored in a single Integration Component Library (ICL) within the System Manager. You can think of an ICL as an artifact repository in your local workspace.

To create a new ICL in System Manager, follow these steps:

1. Right-click the **Integrated Component Libraries** folder and select **New**. This opens the New Integration Component Library window (Figure 7-3 on page 260).
2. Provide a name for the new ICL. If you want to import components from an existing InterChange Server, you can select it now and click **Next**.


New Integration Component Library
New Integration Component Library
Create a new ICL project

Project name: OrderManagement

Project contents

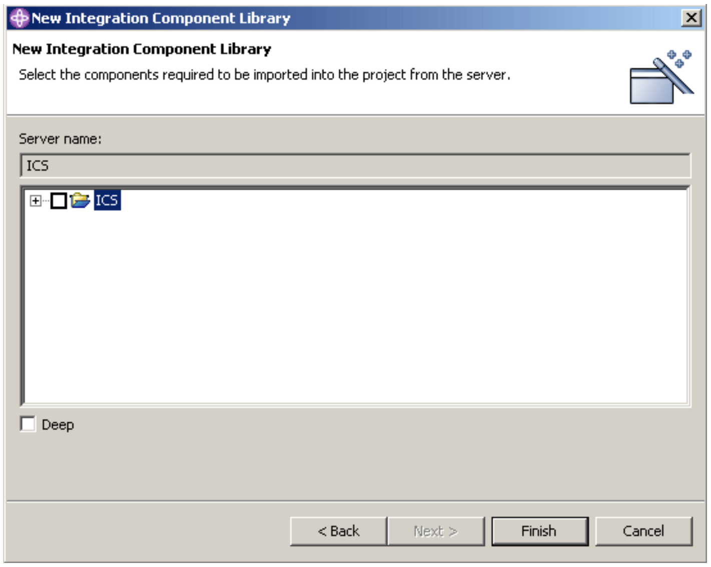
☑ Use default

Directory: C:\IBM\WebSphereICS\Tools\WSWB212\Workspace\OrderManagement

Import components from server: ICS

Figure 7-3 Create a new Integration Component Library

3. If you selected to import components from an existing, choose the components to download from that server. We did not deploy any components to our new server, so we selected no options and clicked **Finish**.


New Integration Component Library
New Integration Component Library

Select the components required to be imported into the project from the server.

Server name: ICS

ICS
Deep
< Back
Next >
Finish
Cancel

Figure 7-4 Importing components from a running InterChange Server

4. When the ICL is created, a folder structure opens (Figure 7-5) in which you can store the components that must be developed for this project.

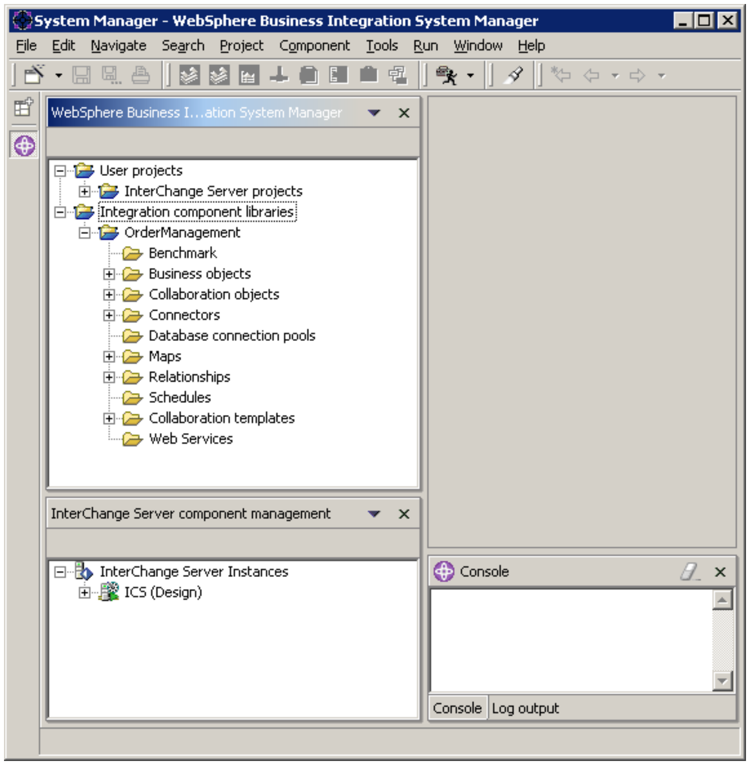

System Manager - WebSphere Business Integration System Manager
File Edit Navigate Search Project Component Tools Run Window Help
WebSphere Business I...ation System Manager
User projects
InterChange Server projects
Integration component libraries
OrderManagement
Benchmark
Business objects
Collaboration objects
Connectors
Database connection pools
Maps
Relationships
Schedules
Collaboration templates
Web Services
InterChange Server component management
InterChange Server Instances
ICS (Design)
Console
Console
Log output

Figure 7-5 New ICL is created

7.5 Create business objects

This section describes the generation of the application-specific business objects for the Order and Item tables and for WebSphere MQ Workflow. Figure 7-6 shows the circled business objects at each diagram end used in the overall solution.

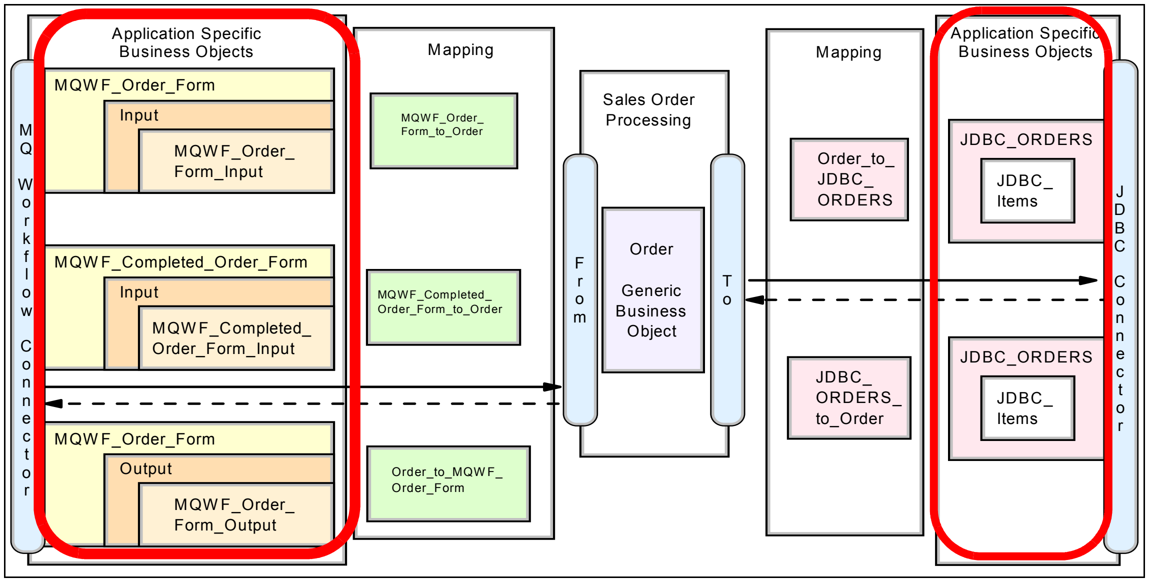


Figure 7-6 Business objects overview

7.5.1 DB2 application-specific business object

To generate the Order and Items ASBOs, we used the JDBC Object Discovery Agent (ODA). An ODA inspects the application data structure and automatically generates business objects that correspond to that specific application and adapter used, in this case the JDBC adapter.

Configuring the JDBC ODA

The script that starts the JDBCODA is called start_JDBC.bat and is located in \WebSphereICS\ODA\JDBC. In this script, we modify the environment variables DRIVERPATH and DRIVERLIB. For DB2, add a reference to DB2Java.zip to the variable DRIVERPATH.

```
REM Modify this Driver path to point to the specific driver you want to use.
set DRIVERPATH=C:\SQLLIB\java\db2java.zip;"%CROSSWORLDS%"\lib\xwutil.jar;...

REM Modify DRIVERLIB path to point to the specific driver dll libraries you
want to use.
set DRIVERLIB=C:\SQLLIB\bin
```

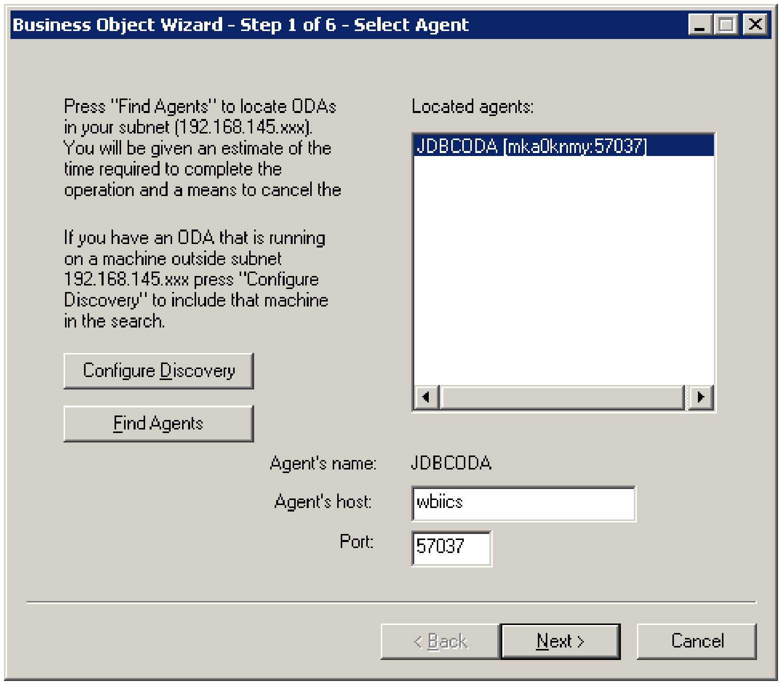
Starting the JDBC ODA

To start the JDBC ODA, select **Start** → **Programs** → **IBM WebSphere Business Integration Adapters** → **Adapters** → **Object Discovery Agent** → **JDBC Object Discovery Agent**. This launches a command-line-based program that listens for discovery requests from the Business Object Designer.

Create the ASBO

To create a JDBC ASBO, open the Business Object Designer and select **File** → **New Using ODA**. Search for all ODAs that are available in the subnet where the Business Object Designer is installed. Provide the necessary information and click **Next**. See Figure 7-7 on page 264. Either of these two methods can be used to connect with the JDBC ODA.

- If your ODA is running on a different subnet, click **Configure Discovery** and enter the IP address where the ODA is executing. You can also find available agents on that system outside your subnet.
- If you want to search for all available ODAs instead of entering the information, click **Find Agents**, select the appropriate ODA in the located agents, and click **Next**.



Business Object Wizard - Step 1 of 6 - Select Agent

Press "Find Agents" to locate ODAs in your subnet (192.168.145.xxx). You will be given an estimate of the time required to complete the operation and a means to cancel the

If you have an ODA that is running on a machine outside subnet 192.168.145.xxx press "Configure Discovery" to include that machine in the search.

Located agents:

JDBCODA [mka0knmy:57037]

Agent's name: JDBCODA

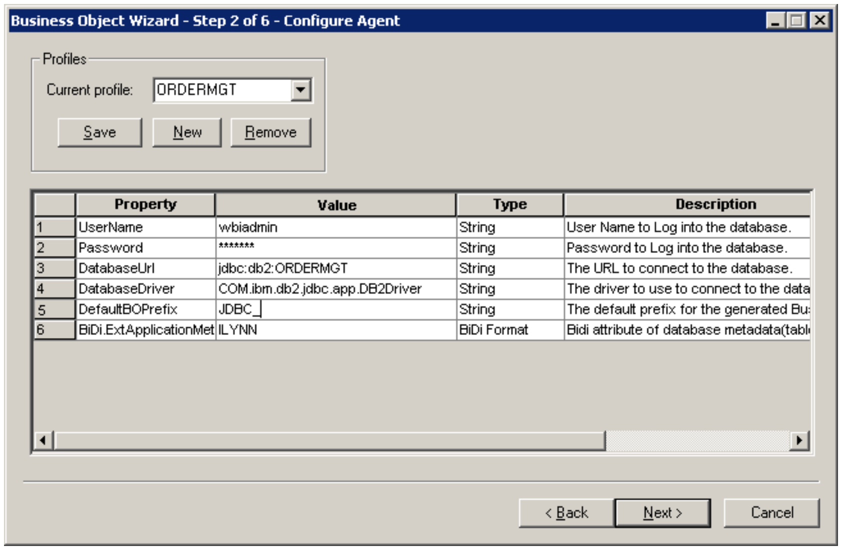
Agent's host: wbiics

Port: 57037

Figure 7-7 Using the Object Discovery Agent to create ASBOs

The Business Object Designer is now connected with the ODA.

1. Because we are using a JDBC ODA, the wizard asks about the appropriate values to connect to the database. Figure 7-8 on page 265 shows the values for the User name, Password, DatabaseUrl, DatabaseDriver, and DefaultBOPrefix. Click **Next**.


Business Object Wizard - Step 2 of 6 - Configure Agent

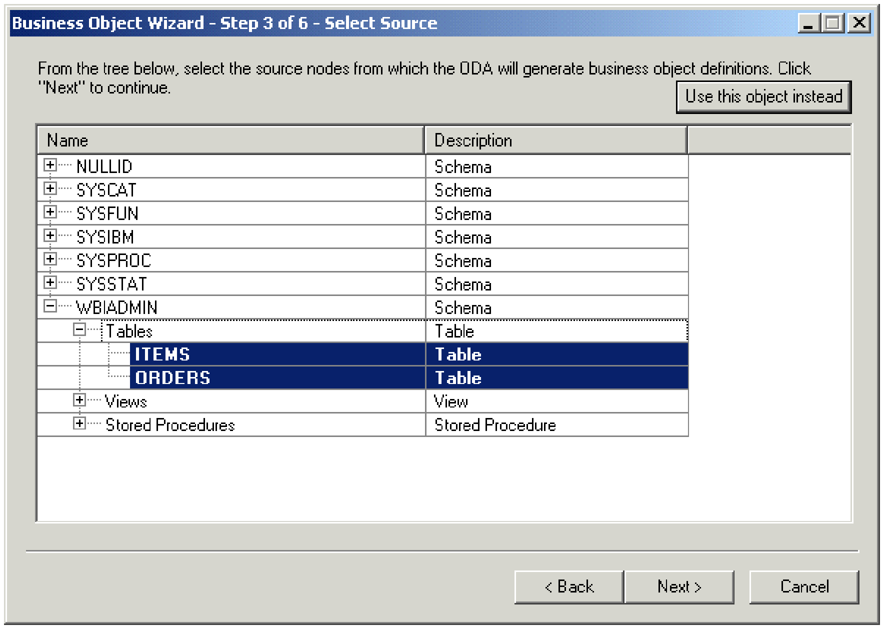
Profiles

Current profile: ORDERMGT

	Property	Value	Type	Description
1	UserName	wbiadmin	String	User Name to Log into the database.
2	Password	*******	String	Password to Log into the database.
3	DatabaseUrl	jdbc:db2:ORDERMGT	String	The URL to connect to the database.
4	DatabaseDriver	COM.ibm.db2.jdbc.app.DB2Driver	String	The driver to use to connect to the data
5	DefaultBOPrefix	JDBC_	String	The default prefix for the generated Bu:
6	BiDi.ExtApplicationMet	ILYNN	BiDi Format	Bidi attribute of database metadata(tabl

Figure 7-8 Enter properties values for the JDBC ODA

2. Select the tables to generate the corresponding business object (Figure 7-9). In our case, we select ITEMS and ORDERS. Click **Next**.

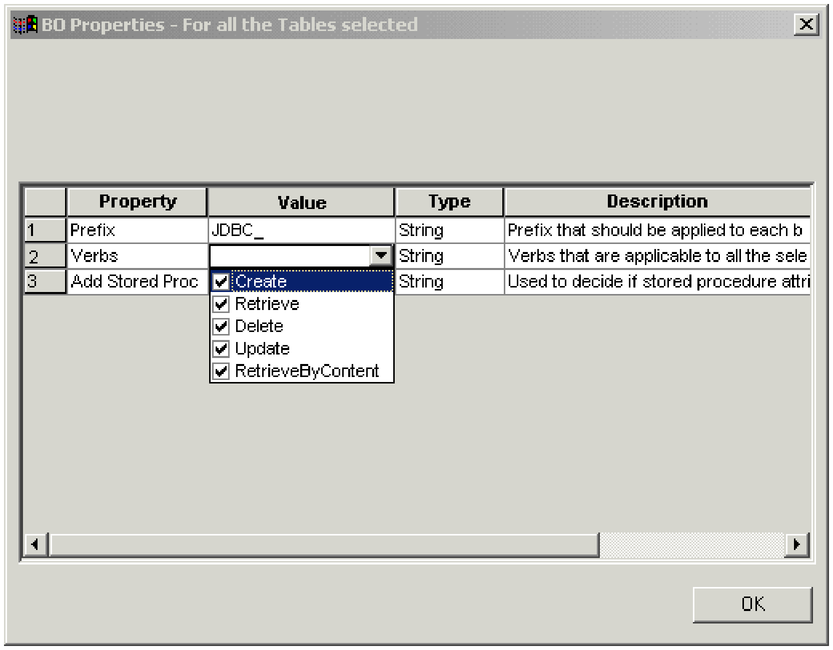

Business Object Wizard - Step 3 of 6 - Select Source

From the tree below, select the source nodes from which the ODA will generate business object definitions. Click "Next" to continue.

Name	Description
NULLID	Schema
SYSCAT	Schema
SYSFUN	Schema
SYSIBM	Schema
SYSPROC	Schema
SYSSTAT	Schema
WBIADMIN	Schema
Tables	Table
ITEMS	Table
ORDERS	Table
Views	View
Stored Procedures	Stored Procedure

Figure 7-9 Select the database tables

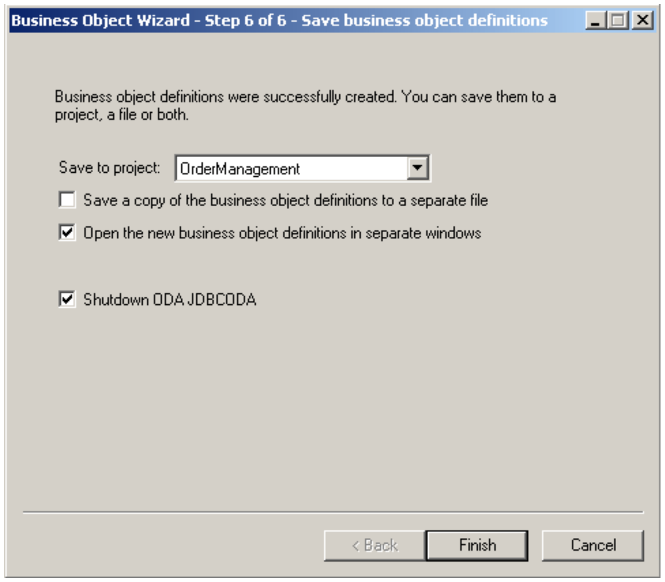
3. A confirmation window opens, showing the tables to generate the business object. Verify the tables and the select **Next**.
4. In the window shown in Figure 7-10, we indicate the properties for the business object that would be created. These properties are the prefix, the verbs, and the option to add stored procedure. Verify that the prefix is JDBC_ and that all verbs are marked. Set the **Add Store Proc** property to **No**. Click **OK**.


BO Properties - For all the Tables selected

	Property	Value	Type	Description
1	Prefix	JDBC_	String	Prefix that should be applied to each b
2	Verbs		String	Verbs that are applicable to all the sele
3	Add Stored Proc		String	Used to decide if stored procedure attri

Figure 7-10 JDBC ODA Properties for the business object to create

5. The business objects are now created. Select the project to store the business objects; we choose **OrderManagement**. You can also indicate that the ODA is no longer needed and can be shut down (Figure 7-11). Click **Finish**.


Business Object Wizard - Step 6 of 6 - Save business object definitions

Business object definitions were successfully created. You can save them to a project, a file or both.

Save to project: OrderManagement

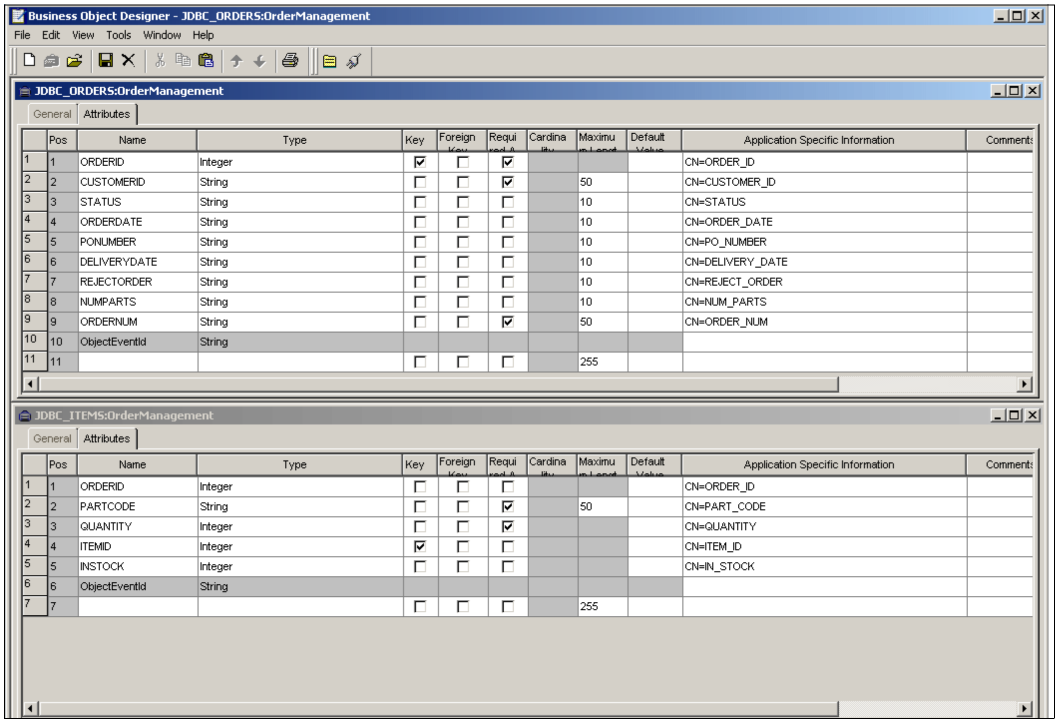
☐ Save a copy of the business object definitions to a separate file

☑ Open the new business object definitions in separate windows

☑ Shutdown ODA JDBCODA

Figure 7-11 JDBC ODA business object creation final screen

Figure 7-12 shows the business object immediately after creation. As we can see in the JDBC_ITEMS business object, the foreign key is not preserved and the ITEMS are not included in the JDBC_ORDERS business object.


Business Object Designer - JDBC_ORDERS:OrderManagement
File Edit View Tools Window Help
JDBC_ORDERS:OrderManagement
General
Attributes
JDBC_ITEMS:OrderManagement
General
Attributes

JDBC_ORDERS:OrderManagement

	Pos	Name	Type	Key	Foreign	Requi	Cardina	Maximu	Default	Application Specific Information	Comments
1	1	ORDERID	Integer	✓	☐	✓				CN=ORDER_ID	
2	2	CUSTOMERID	String	☐	☐	✓		50		CN=CUSTOMER_ID	
3	3	STATUS	String	☐	☐	☐		10		CN=STATUS	
4	4	ORDERDATE	String	☐	☐	☐		10		CN=ORDER_DATE	
5	5	PONUMBER	String	☐	☐	☐		10		CN=PO_NUMBER	
6	6	DELIVERYDATE	String	☐	☐	☐		10		CN=DELIVERY_DATE	
7	7	REJECTORDER	String	☐	☐	☐		10		CN=REJECT_ORDER	
8	8	NUMPARTS	String	☐	☐	☐		10		CN=NUM_PARTS	
9	9	ORDERNUM	String	☐	☐	✓		50		CN=ORDER_NUM	
10	10	ObjectEventId	String								
11	11			☐	☐	☐		255			

JDBC_ITEMS:OrderManagement

	Pos	Name	Type	Key	Foreign	Requi	Cardina	Maximu	Default	Application Specific Information	Comments
1	1	ORDERID	Integer	☐	☐	☐				CN=ORDER_ID	
2	2	PARTCODE	String	☐	☐	✓		50		CN=PART_CODE	
3	3	QUANTITY	Integer	☐	☐	✓				CN=QUANTITY	
4	4	ITEMID	Integer	✓	☐	☐				CN=ITEM_ID	
5	5	INSTOCK	Integer	☐	☐	☐				CN=IN_STOCK	
6	6	ObjectEventId	String								
7	7			☐	☐	☐		255			

Figure 7-12 Saved to project

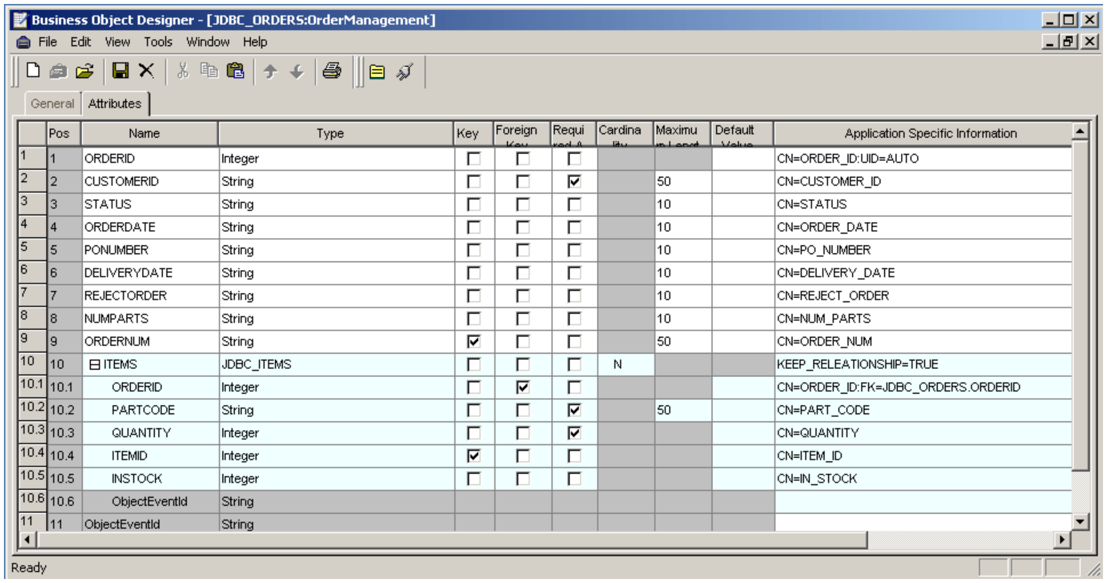
6. Add a new entry in JDBC_ORDERS called `ITEMS` with type `JDBC_ITEMS` and cardinality `N`.
7. Now expand the element that we just created and select the **Foreign** property for the ORDERNUMBER element. Add to the App Spec Info the relation with the parent. In this case the relation is:

   ```
   FK=JDBC_ORDERS.ORDERID
   ```

   Attention: The delimiter in the App Spec Info for the JDBC Adapter is a colon. This delimiter is not the same for all adapters. Check the corresponding manual to verify the delimiter.

8. Also, add the line `UID=AUTO` to the field Application-Specific Info for the element ORDERID. This indicates to the JDBC connector that it is a generated field.

Figure 7-13 shows the parent business object with all of the changes that we have made.


Business Object Designer - [JDBC_ORDERS:OrderManagement]

File Edit View Tools Window Help

General | Attributes

	Pos	Name	Type	Key	Foreign	Requi	Cardina	Maximu	Default	Application Specific Information
1	1	ORDERID	Integer	☐	☐	☐				CN=ORDER_ID:UID=AUTO
2	2	CUSTOMERID	String	☐	☐	☑		50		CN=CUSTOMER_ID
3	3	STATUS	String	☐	☐	☐		10		CN=STATUS
4	4	ORDERDATE	String	☐	☐	☐		10		CN=ORDER_DATE
5	5	PONUMBER	String	☐	☐	☐		10		CN=PO_NUMBER
6	6	DELIVERYDATE	String	☐	☐	☐		10		CN=DELIVERY_DATE
7	7	REJECTORDER	String	☐	☐	☐		10		CN=REJECT_ORDER
8	8	NUMPARTS	String	☐	☐	☐		10		CN=NUM_PARTS
9	9	ORDERNUM	String	☑	☐	☐		50		CN=ORDER_NUM
10	10	⊟ ITEMS	JDBC_ITEMS	☐	☐	☐	N			KEEP_RELEATIONSHIP=TRUE
10.1	10.1	ORDERID	Integer	☐	☑	☐				CN=ORDER_ID:FK=JDBC_ORDERS.ORDERID
10.2	10.2	PARTCODE	String	☐	☐	☑		50		CN=PART_CODE
10.3	10.3	QUANTITY	Integer	☐	☐	☑				CN=QUANTITY
10.4	10.4	ITEMID	Integer	☑	☐	☐				CN=ITEM_ID
10.5	10.5	INSTOCK	Integer	☐	☐	☐				CN=IN_STOCK
10.6	10.6	ObjectEventId	String							
11	11	ObjectEventId	String							

Ready

Figure 7-13 Displays JDBC_ORDERS business object in its final form

7.5.2 WebSphere MQ Workflow application-specific business object

To generate the WebSphere MQ Workflow ASBOs, we use the fdlborgen utility that comes with the WebSphere MQ Workflow adapter. There are two ASBOs required to be generated for the WebSphere MQ Workflow.

- Order_Form
- Completed_Order_Form

Order_Form is the Workflow data structure used in Create Sales Order activity and the Completed_Order_Form is the data structure used in the Confirm_Order activity and the Delete_Order activity.

The fdlborgen utility generates a single ASBO each time it is executed, thus we executed the utility three times to generate the three needed objects, each time, specifying the different workflow data structure name.

The syntax for the command is as follows:

```
fdlborgen -i[input-file] -o[output-file] -n[object-name] -p[prefix]
```

The variables stand for the following:

- *input-file* is the exported FDL file.
- *output-file* is the generated business object file.
- *object-name* is the Workflow data structure name.
- *prefix* is the prefix specified before the business object name.

The fdlborgen utility is available only when you have installed the WebSphere MQ Workflow Adapter. Move the FDL file that contains the data structures to \WebSphereICS\connectors\WebSphereMQWorkflow\utilities.

To generate MQWF_Order_Form business object, run the following command.

```
fdlborgen -iprocess_model.fdl -oOrder_Form.in -nOrder_Form -pMQWF_
```

In this command, the variables are:

- *process_model.fdl* is the exported process model.
- *Order_Form.in* is the name of the file that will contain the generated business object.
- *Order_Form* is the name of the data structure.
- *MQWF_ is* the prefix to be used when generating business objects.

To generate MQWF_Completed_Order_Form business object, run the following command.

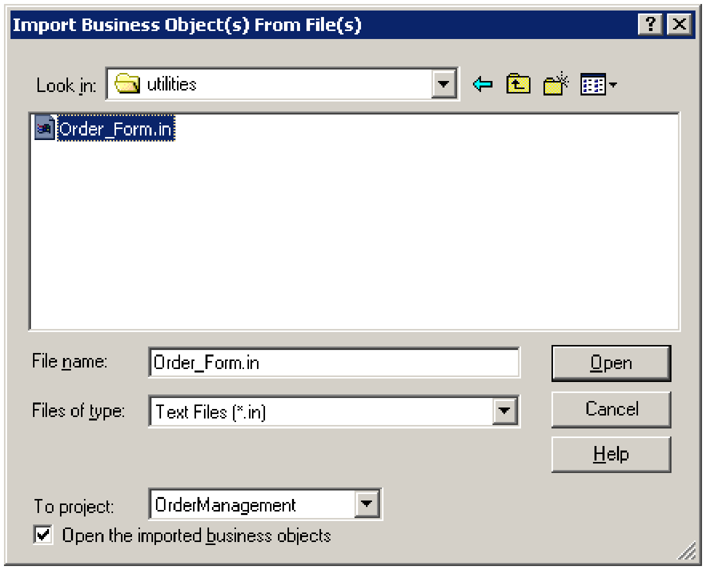
```
fdlborgen -iprocess_model.fdl -oCompleted_Order_Form.in -nCompleted_Order_Form
-pMQWF_
```

In this command:

- *process_model.fdl* is the exported process model.
- *Completed_Order_Form.in* is the name of the file that will contain the generated business object.
- *Completed_Order_Form* is the name of the data structure.
- *MQWF_* is the prefix to be used when generating business objects.

When the utility completes, start the Business Object Designer from the System Manager:

1. Select **File** → **Open From File**.
2. Browse for the generated file Order_Form.in and change the file type, which is by default *.xsd, to ***.in** (Figure 7-14). Also, make sure to select the correct project. The Business Object Designer will save the business objects in the selected ICL.


Import Business Object(s) From File(s)

Look in: utilities

Order_Form.in

File name: Order_Form.in

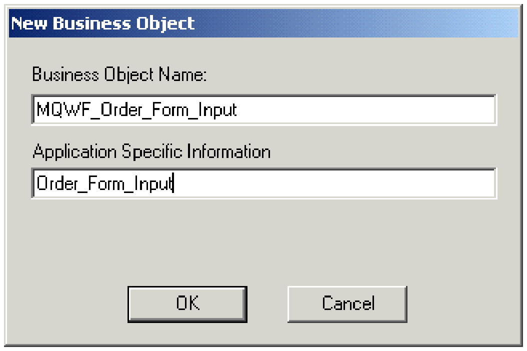
Files of type: Text Files (*.in)

To project: OrderManagement

☑ Open the imported business objects

Figure 7-14 Select business object file

3. Save the generated business object as `MQWF_Order_Form_Input` and `MQWF_Order_Output also within the OrderManagement ICL` (Figure 7-15).


New Business Object

Business Object Name:

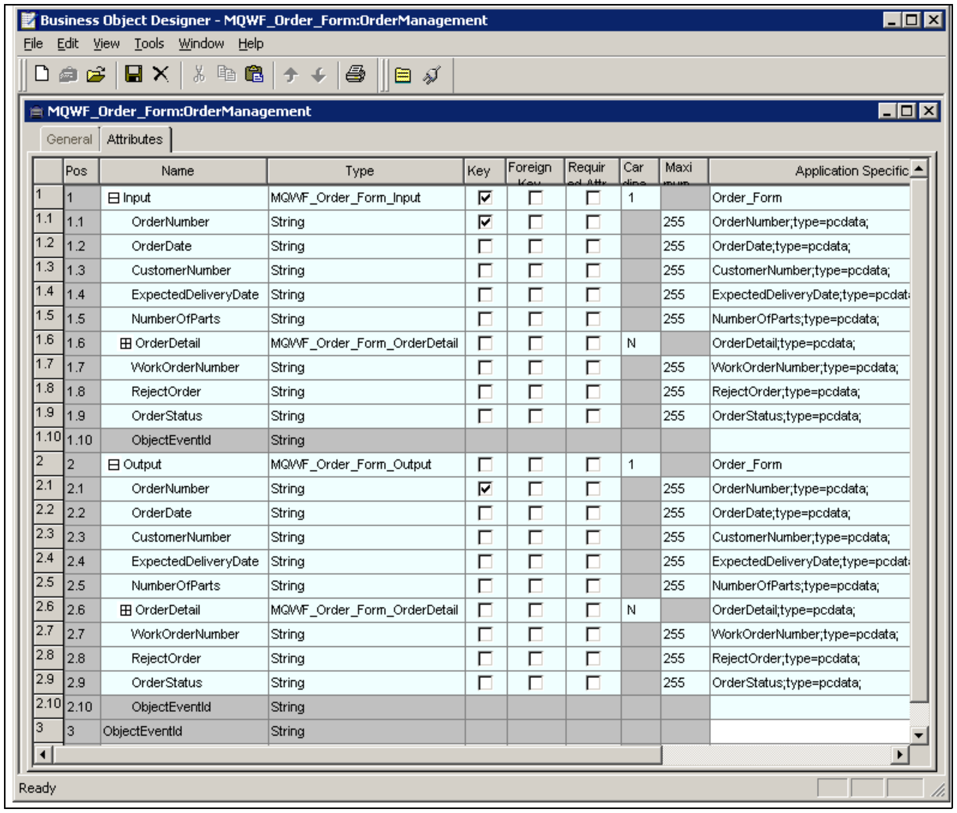
MQWF_Order_Form_Input

Application Specific Information

Order_Form_Input

Figure 7-15 Create input child business object

4. Re-open and modify the parent business object, MQWF_Order_Form, to have the input and output child business objects that we created. Remove all of the existing attributes and create two new attributes with the type of the child objects we just created, `MQWF_Order_Form_Input` and `MQWF_Order_Form_Output`.
5. Set the child object Input as a key, and the final MQWF_Order_Form parent business object is shown in Figure 7-16.



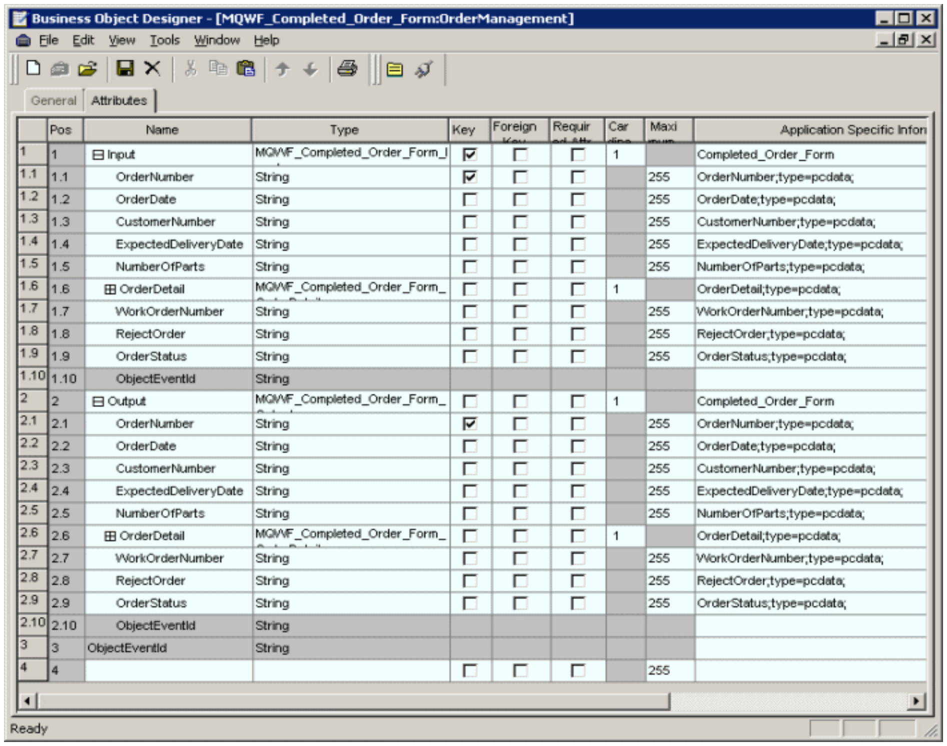
Business Object Designer - MQWF_Order_Form:OrderManagement

MQWF_Order_Form:OrderManagement

	Pos	Name	Type	Key	Foreign Key	Required Attr	Cardina	Maximum	Application Specific
1	1	⊟ Input	MQWF_Order_Form_Input	☑	☐	☐	1		Order_Form
1.1	1.1	OrderNumber	String	☑	☐	☐		255	OrderNumber;type=pcdata;
1.2	1.2	OrderDate	String	☐	☐	☐		255	OrderDate;type=pcdata;
1.3	1.3	CustomerNumber	String	☐	☐	☐		255	CustomerNumber;type=pcdata;
1.4	1.4	ExpectedDeliveryDate	String	☐	☐	☐		255	ExpectedDeliveryDate;type=pcdat
1.5	1.5	NumberOfParts	String	☐	☐	☐		255	NumberOfParts;type=pcdata;
1.6	1.6	⊞ OrderDetail	MQWF_Order_Form_OrderDetail	☐	☐	☐	N		OrderDetail;type=pcdata;
1.7	1.7	WorkOrderNumber	String	☐	☐	☐		255	WorkOrderNumber;type=pcdata;
1.8	1.8	RejectOrder	String	☐	☐	☐		255	RejectOrder;type=pcdata;
1.9	1.9	OrderStatus	String	☐	☐	☐		255	OrderStatus;type=pcdata;
1.10	1.10	ObjectEventId	String						
2	2	⊟ Output	MQWF_Order_Form_Output	☐	☐	☐	1		Order_Form
2.1	2.1	OrderNumber	String	☑	☐	☐		255	OrderNumber;type=pcdata;
2.2	2.2	OrderDate	String	☐	☐	☐		255	OrderDate;type=pcdata;
2.3	2.3	CustomerNumber	String	☐	☐	☐		255	CustomerNumber;type=pcdata;
2.4	2.4	ExpectedDeliveryDate	String	☐	☐	☐		255	ExpectedDeliveryDate;type=pcdat
2.5	2.5	NumberOfParts	String	☐	☐	☐		255	NumberOfParts;type=pcdata;
2.6	2.6	⊞ OrderDetail	MQWF_Order_Form_OrderDetail	☐	☐	☐	N		OrderDetail;type=pcdata;
2.7	2.7	WorkOrderNumber	String	☐	☐	☐		255	WorkOrderNumber;type=pcdata;
2.8	2.8	RejectOrder	String	☐	☐	☐		255	RejectOrder;type=pcdata;
2.9	2.9	OrderStatus	String	☐	☐	☐		255	OrderStatus;type=pcdata;
2.10	2.10	ObjectEventId	String						
3	3	ObjectEventId	String						

Figure 7-16 Workflow-specific MQWF_Order_Form business object

6. Repeat steps 1-5 for the Completed_Order_Form data structure as shown in Figure 7-17.



Business Object Designer - [MQWF_Completed_Order_Form:OrderManagement]

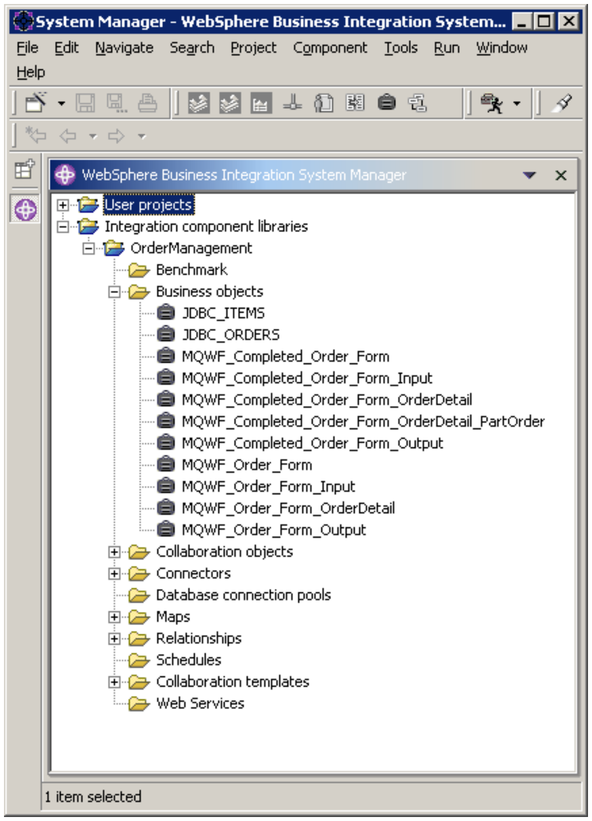
General | Attributes

	Pos	Name	Type	Key	Foreign	Requir	Car	Maxi	Application Specific Infor
1	1	⊟ Input	MQWF_Completed_Order_Form_I	☑	☐	☐	1		Completed_Order_Form
1.1	1.1	OrderNumber	String	☑	☐	☐		255	OrderNumber;type=pcdata;
1.2	1.2	OrderDate	String	☐	☐	☐		255	OrderDate;type=pcdata;
1.3	1.3	CustomerNumber	String	☐	☐	☐		255	CustomerNumber;type=pcdata;
1.4	1.4	ExpectedDeliveryDate	String	☐	☐	☐		255	ExpectedDeliveryDate;type=pcdata;
1.5	1.5	NumberOfParts	String	☐	☐	☐		255	NumberOfParts;type=pcdata;
1.6	1.6	⊞ OrderDetail	MQWF_Completed_Order_Form_	☐	☐	☐	1		OrderDetail;type=pcdata;
1.7	1.7	WorkOrderNumber	String	☐	☐	☐		255	WorkOrderNumber;type=pcdata;
1.8	1.8	RejectOrder	String	☐	☐	☐		255	RejectOrder;type=pcdata;
1.9	1.9	OrderStatus	String	☐	☐	☐		255	OrderStatus;type=pcdata;
1.10	1.10	ObjectEventId	String						
2	2	⊟ Output	MQWF_Completed_Order_Form_	☐	☐	☐	1		Completed_Order_Form
2.1	2.1	OrderNumber	String	☑	☐	☐		255	OrderNumber;type=pcdata;
2.2	2.2	OrderDate	String	☐	☐	☐		255	OrderDate;type=pcdata;
2.3	2.3	CustomerNumber	String	☐	☐	☐		255	CustomerNumber;type=pcdata;
2.4	2.4	ExpectedDeliveryDate	String	☐	☐	☐		255	ExpectedDeliveryDate;type=pcdata;
2.5	2.5	NumberOfParts	String	☐	☐	☐		255	NumberOfParts;type=pcdata;
2.6	2.6	⊞ OrderDetail	MQWF_Completed_Order_Form_	☐	☐	☐	1		OrderDetail;type=pcdata;
2.7	2.7	WorkOrderNumber	String	☐	☐	☐		255	WorkOrderNumber;type=pcdata;
2.8	2.8	RejectOrder	String	☐	☐	☐		255	RejectOrder;type=pcdata;
2.9	2.9	OrderStatus	String	☐	☐	☐		255	OrderStatus;type=pcdata;
2.10	2.10	ObjectEventId	String						
3	3	ObjectEventId	String						
4	4			☐	☐	☐		255	

Ready

Figure 7-17 Workflow specific business object MQWF_Completed_Order_Form

7. When the two parent business objects have been successfully modified, save all changes and close the Business Object Designer. In the System Manager (Figure 7-18), we can see all of the ASBOs we created.


System Manager - WebSphere Business Integration System...
File Edit Navigate Search Project Component Tools Run Window Help
WebSphere Business Integration System Manager

- User projects
- Integration component libraries
  - OrderManagement
    - Benchmark
    - Business objects
      - JDBC_ITEMS
      - JDBC_ORDERS
      - MQWF_Completed_Order_Form
      - MQWF_Completed_Order_Form_Input
      - MQWF_Completed_Order_Form_OrderDetail
      - MQWF_Completed_Order_Form_OrderDetail_PartOrder
      - MQWF_Completed_Order_Form_Output
      - MQWF_Order_Form
      - MQWF_Order_Form_Input
      - MQWF_Order_Form_OrderDetail
      - MQWF_Order_Form_Output
    - Collaboration objects
    - Connectors
    - Database connection pools
    - Maps
    - Relationships
    - Schedules
    - Collaboration templates
    - Web Services

1 item selected

Figure 7-18 List of business objects

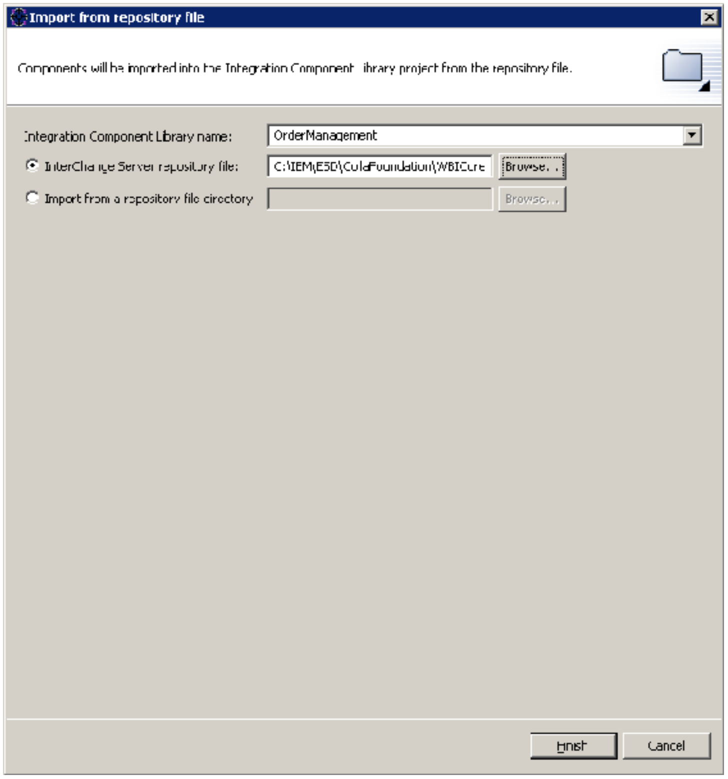
7.5.3 Generic business object

We use the Order business object as our GBO because it is the business object that is used by the collaboration in 7.8, “Collaboration template” on page 333.

The Order GBO is provided in a JAR file that contains many other business objects. This JAR file is obtained by either installing the Collaboration Foundation or by installing an Industry Collaboration as we did in Chapter 3, “Implementing the runtime components” on page 45.

To import this file in System Manager, right-click the ICL **OrderManagement** and select **Import from Repository File**. Click **Browse** to locate

BIA_BO_BaseCollabBOs.jar in the C:\IBM\IndustryCollabs\OrderMgmt\WBICoreCollabs directory.(Figure 7-19).



Components will be imported into the Integration Component Library project from the repository file.

Integration Component Library name: OrderManagement

(●) InterChange Server repository file: C:\IBM\ESD\ColaFoundation\WBICore

( ) Import from a repository file directory

Figure 7-19 Import Order business object

Now the System Manager shows all of the business objects that are part of this jar file, including Order.

7.6 Create maps

The next step is to create the maps that are used when converting from an ASBO to GBO and from GBO to ASBO.

As we are going from WebSphere MQ Workflow to WebSphere InterChange Server and to DB2 and then back to WebSphere InterChange Server and back to WebSphere MQ Workflow, we need four main maps. And as our business object contains child objects, we need four extra maps for the child business objects.

That makes a total of eight maps and plus one additional map for MQWF_Completed_Order_Form business object to Order.

7.6.1 From ASBO MQWF_Order_Form to GBO Order

This section discusses the development of the main map and submap to transform the incoming workflow input objects to the generic business object Order. Figure 7-20 shows these maps within the overall solution.

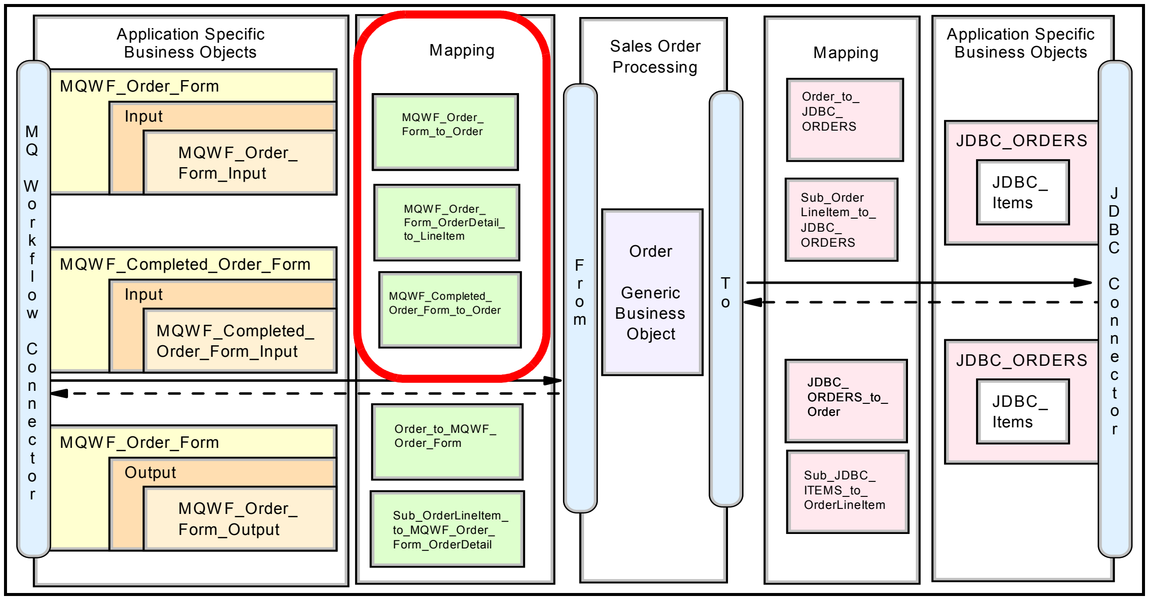


Figure 7-20 Mapping the workflow input objects to the GBO Order

MQWF_Order_Form_OrderDetail_to_OrderLineItem

This submap maps the information that is contained in the application-specific object MQWF_Order_Form_Detail to the generic object OrderLineItem. Both objects are child objects.

1. Start the Map Designer from System Manager or from the Start Programs menu and select **File** → **New Map**.

2. Select the ICL where the new map will be created (Figure 7-21). Select **OrderManagement** and click **Next**.

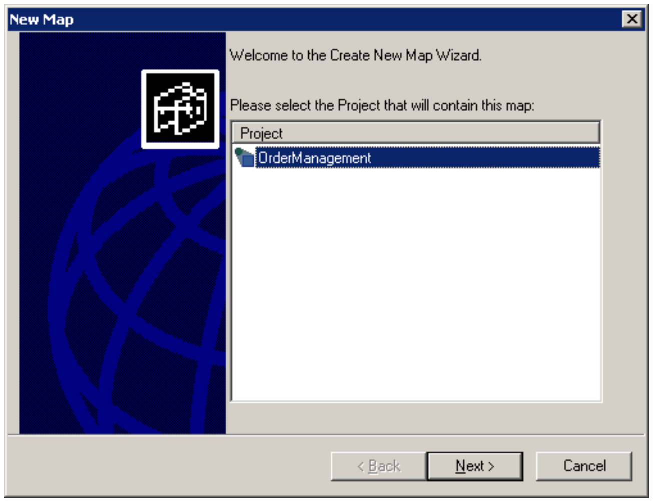
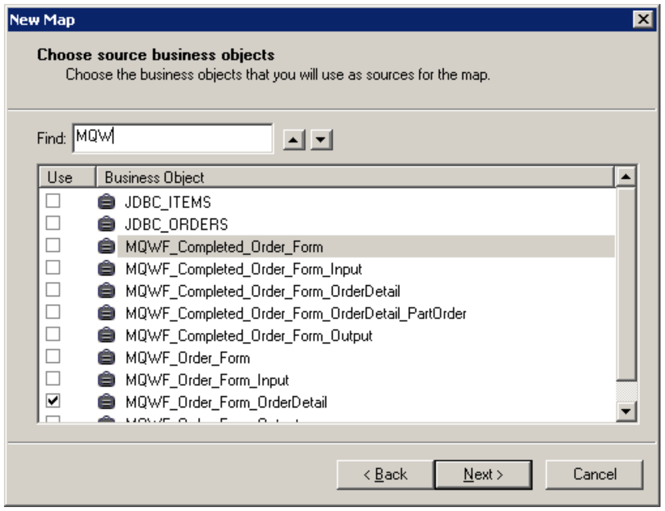

New Map
Welcome to the Create New Map Wizard.
Please select the Project that will contain this map:
Project
OrderManagement
< Back
Next >
Cancel

Figure 7-21 Select the ICL

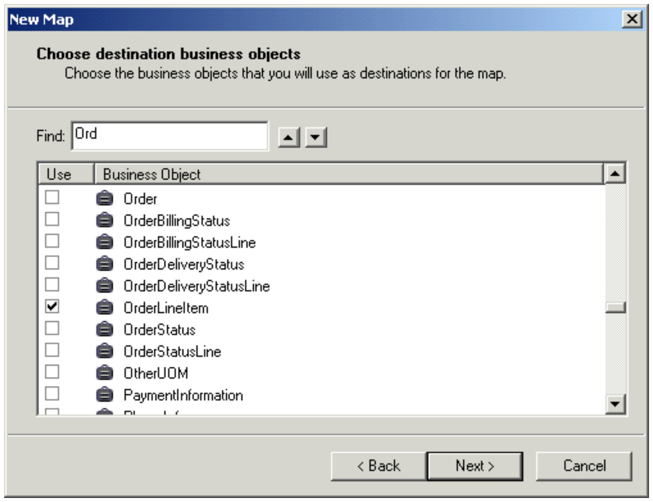
3. Select the source business object, **MQWF_Order_Form_OrderDetail**, as shown in Figure 7-22, and click **Next**.


New Map
Choose source business objects
Choose the business objects that you will use as sources for the map.
Find: MQW
< Back
Next >
Cancel

Use	Business Object
☐	JDBC_ITEMS
☐	JDBC_ORDERS
☐	MQWF_Completed_Order_Form
☐	MQWF_Completed_Order_Form_Input
☐	MQWF_Completed_Order_Form_OrderDetail
☐	MQWF_Completed_Order_Form_OrderDetail_PartOrder
☐	MQWF_Completed_Order_Form_Output
☐	MQWF_Order_Form
☐	MQWF_Order_Form_Input
✓	MQWF_Order_Form_OrderDetail

Figure 7-22 Select source business object

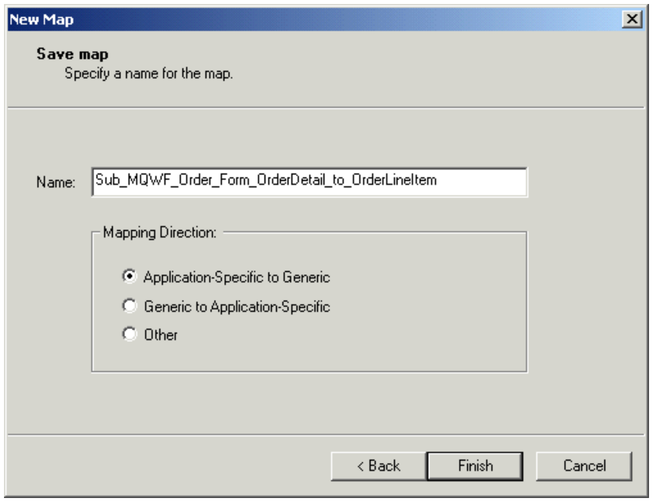
4. Select the destination business object, **OrderLineItem**, as shown in Figure 7-23, and click **Next**.


New Map
Choose destination business objects
Choose the business objects that you will use as destinations for the map.
Find: Ord
< Back
Next >
Cancel

Use	Business Object
☐	Order
☐	OrderBillingStatus
☐	OrderBillingStatusLine
☐	OrderDeliveryStatus
☐	OrderDeliveryStatusLine
✓	OrderLineItem
☐	OrderStatus
☐	OrderStatusLine
☐	OtherUOM
☐	PaymentInformation

Figure 7-23   Select destination business object

5. Enter `Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem` as the map name, and select the map direction as **Application-Specific to Generic**, as shown in Figure 7-24. Click **Finish**.



Name: Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem

Mapping Direction:
[✓ Application-Specific to Generic]
Generic to Application-Specific
Other

Figure 7-24   Select the map direction

The Map Designer now shows the Diagram tab, where you can drag and drop elements from the source object to the target object. Table 7-3 shows the mapping specification.

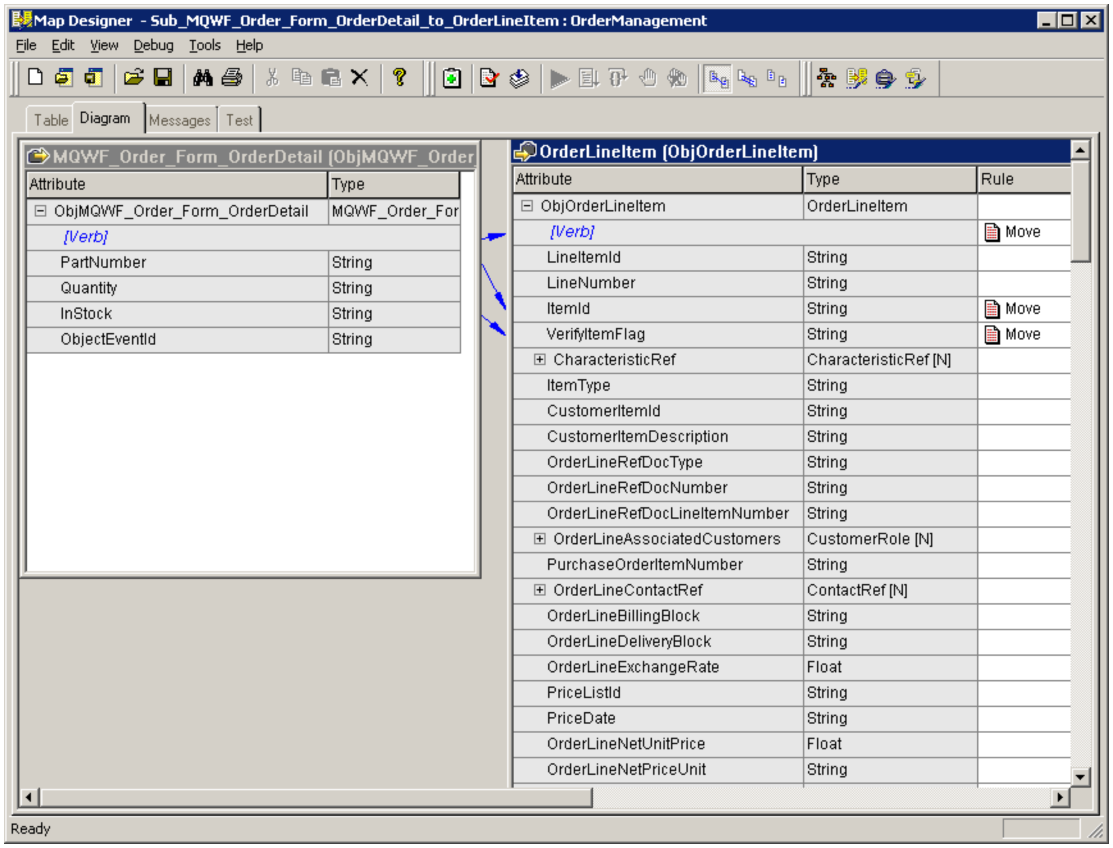
Table 7-3 Mapping in Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem

Source	Destination
Verb	Verb
Quantity	QtyRequired
PartNumber	ItemId
InStock	VerifyItemFlag

Attention: While doing the mapping, make sure to set the rule to **Move**.

Tip: If the **Move** mapping is your most common action, make that the default drag and drop action by selecting **View** → **Preferences** → **Key Mapping** and changing the characteristics to match your preferences.

Figure 7-25 shows the Map Designer after all mapping has been finished. Save and compile the map.


Map Designer - Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test
Ready

MQWF_Order_Form_OrderDetail (ObjMQWF_Order

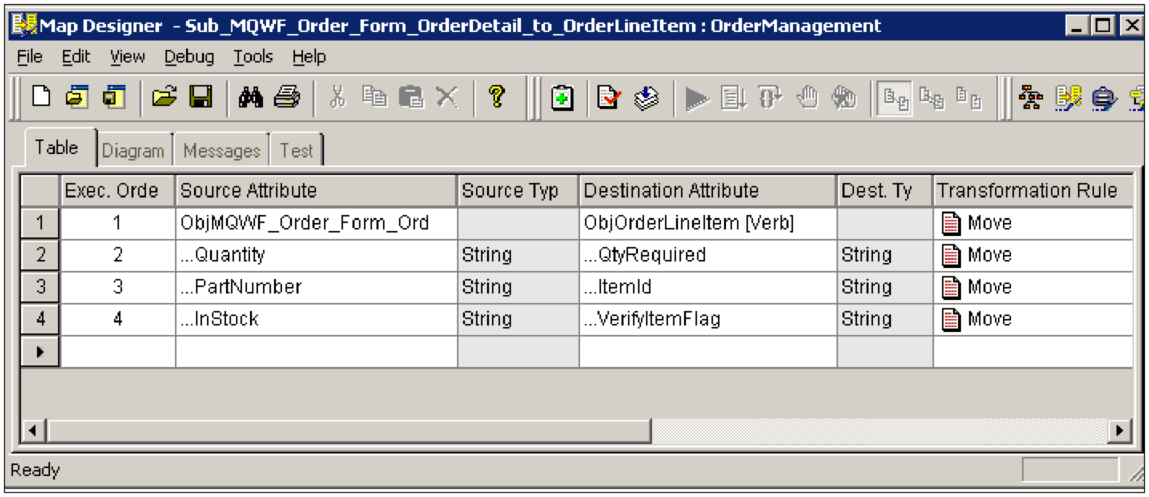
Attribute	Type
⊟ ObjMQWF_Order_Form_OrderDetail	MQWF_Order_For
[Verb]	
PartNumber	String
Quantity	String
InStock	String
ObjectEventId	String

OrderLineItem (ObjOrderLineItem)

Attribute	Type	Rule
⊟ ObjOrderLineItem	OrderLineItem	
[Verb]		Move
LineItemId	String	
LineNumber	String	
ItemId	String	Move
VerifyItemFlag	String	Move
⊞ CharacteristicRef	CharacteristicRef [N]	
ItemType	String	
CustomerItemId	String	
CustomerItemDescription	String	
OrderLineRefDocType	String	
OrderLineRefDocNumber	String	
OrderLineRefDocLineItemNumber	String	
⊞ OrderLineAssociatedCustomers	CustomerRole [N]	
PurchaseOrderItemNumber	String	
⊞ OrderLineContactRef	ContactRef [N]	
OrderLineBillingBlock	String	
OrderLineDeliveryBlock	String	
OrderLineExchangeRate	Float	
PriceListId	String	
PriceDate	String	
OrderLineNetUnitPrice	Float	
OrderLineNetPriceUnit	String	

Figure 7-25 Completed submap

Due to space constraints, this figure shows only the first three mappings. The Table tab shows all the mappings.


Map Designer - Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test
Ready

	Exec. Orde	Source Attribute	Source Typ	Destination Attribute	Dest. Ty	Transformation Rule
1	1	ObjMQWF_Order_Form_Ord		ObjOrderLineItem [Verb]		Move
2	2	...Quantity	String	...QtyRequired	String	Move
3	3	...PartNumber	String	...ItemId	String	Move
4	4	...InStock	String	...VerifyItemFlag	String	Move

Figure 7-26 The Table tab of the Map Designer

MQWF_Order_Form_to_Order

This maps the information that is contained in the application-specific object MQWF_Order_Form to the generic object Order. This map uses the submap Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem.

Note: The process for creating a new map is always the same. Therefore, the information in this and later sections about mapping contains only the information that is required to build the map, not detailed instructions. Use the instructions in the previous section to continue creating new maps.

The information for creating the map is:

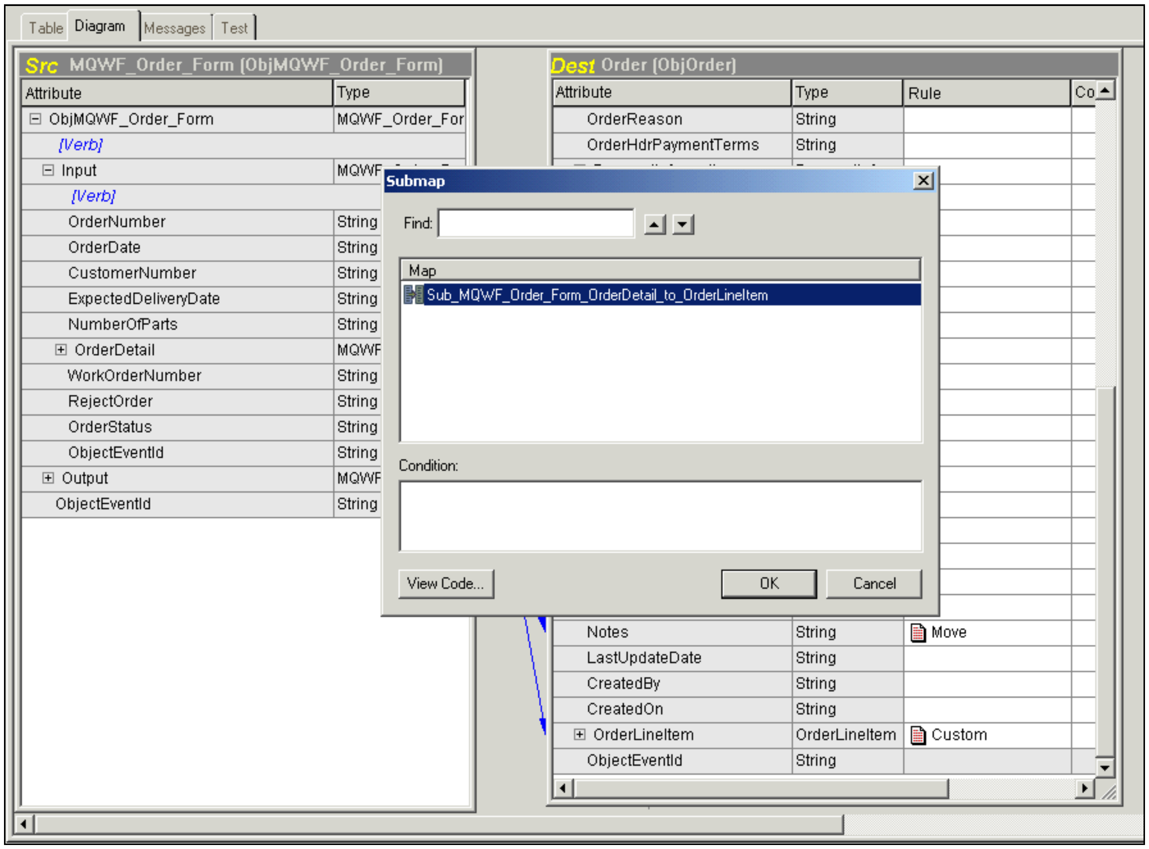
Source business object	`MQWF_Order_Form`
Destination business object	`Order`
Map name	`MQWF_Order_Form_to_Order`
Mapping direction	`Application-Specific to Generic`

Table 7-4 shows the mapping specification.

Table 7-4 Mapping in MQWF_Order_Form_to_Order

Source	**Destination**
Verb	Verb
OrderNumber	OrderNumber
OrderDate	OrderDate
CustomerNumber	CustomerId
ExpectedDeliveryDate	PODate
NumberOfParts	Notes
OrderDetail	OrderLineItem
WorkOrderNumber	PONumber
RejectOrder	OrderType
OrderStatus	OrderStatus

When mapping the OrderDetail to OrderLineItem, set the mapping rule to **Submap**. A new window (Figure 7-27 on page 283) opens so you can select the submap.



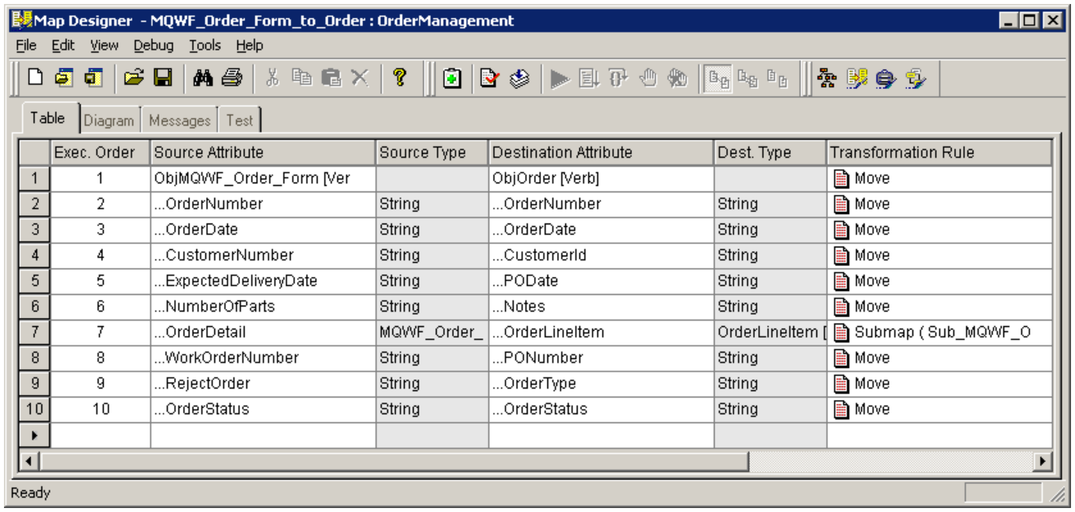
Attribute	Type
ObjMQWF_Order_Form	MQWF_Order_For
[Verb]	
Input	MQWF
[Verb]	
OrderNumber	String
OrderDate	String
CustomerNumber	String
ExpectedDeliveryDate	String
NumberOfParts	String
OrderDetail	MQWF
WorkOrderNumber	String
RejectOrder	String
OrderStatus	String
ObjectEventId	String
Output	MQWF
ObjectEventId	String

Attribute	Type	Rule	Co
OrderReason	String		
OrderHdrPaymentTerms	String		
Notes	String	Move	
LastUpdateDate	String		
CreatedBy	String		
CreatedOn	String		
OrderLineItem	OrderLineItem	Custom	
ObjectEventId	String		

Map
Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem

Figure 7-27 Select the submap

Figure 7-28 shows the Table tab of the Map Designer after all mapping is complete. Save and compile the map.


Map Designer - MQWF_Order_Form_to_Order : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test
Ready

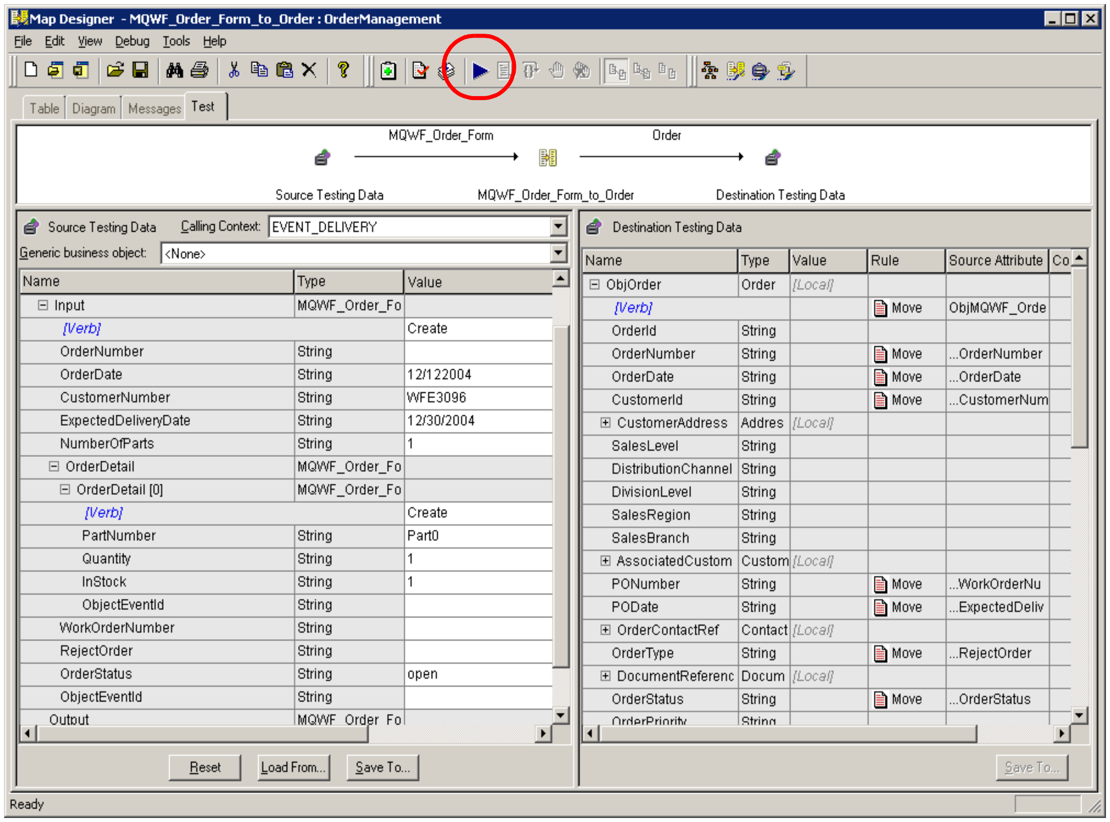
	Exec. Order	Source Attribute	Source Type	Destination Attribute	Dest. Type	Transformation Rule
1	1	ObjMQWF_Order_Form [Ver		ObjOrder [Verb]		Move
2	2	...OrderNumber	String	...OrderNumber	String	Move
3	3	...OrderDate	String	...OrderDate	String	Move
4	4	...CustomerNumber	String	...CustomerId	String	Move
5	5	...ExpectedDeliveryDate	String	...PODate	String	Move
6	6	...NumberOfParts	String	...Notes	String	Move
7	7	...OrderDetail	MQWF_Order_	...OrderLineItem	OrderLineItem [	Submap (Sub_MQWF_O
8	8	...WorkOrderNumber	String	...PONumber	String	Move
9	9	...RejectOrder	String	...OrderType	String	Move
10	10	...OrderStatus	String	...OrderStatus	String	Move

Figure 7-28 Map the attributes

Testing the map

Before proceeding, test the maps as part of a standard unit testing development process. To test the map, click the corresponding tab in the Map Designer.

Select **EVENT_DELIVERY** as the Calling Context in the Source Testing Data pane. You might also need to add an instance for child business objects. Right-click the child business object (**Input of OrderItem**) and select **Add Instance**. Add also one or more instances of OrderItem. Enter test values for each element to test the map.



Source Testing Data

Calling Context: EVENT_DELIVERY

Generic business object: <None>

Name	Type	Value
⊟ Input	MQWF_Order_Fo	
[Verb]		Create
OrderNumber	String	
OrderDate	String	12/122004
CustomerNumber	String	WFE3096
ExpectedDeliveryDate	String	12/30/2004
NumberOfParts	String	1
⊟ OrderDetail	MQWF_Order_Fo	
⊟ OrderDetail [0]	MQWF_Order_Fo	
[Verb]		Create
PartNumber	String	Part0
Quantity	String	1
InStock	String	1
ObjectEventId	String	
WorkOrderNumber	String	
RejectOrder	String	
OrderStatus	String	open
ObjectEventId	String	
Output	MQWF_Order_Fo	

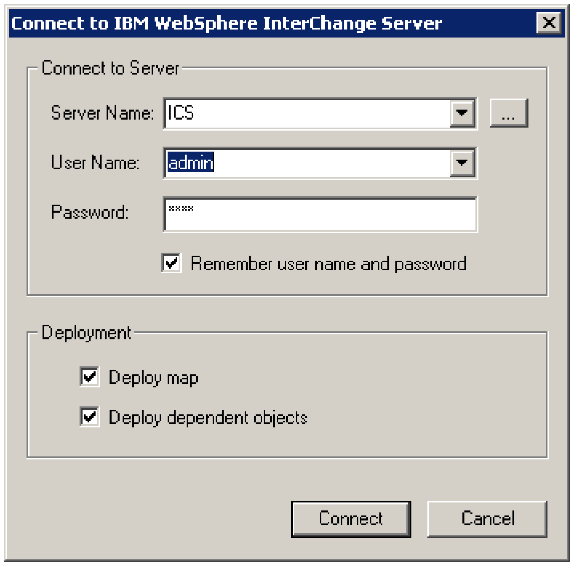
Destination Testing Data

Name	Type	Value	Rule	Source Attribute	Co
⊟ ObjOrder	Order	[Local]			
[Verb]			Move	ObjMQWF_Orde	
OrderId	String				
OrderNumber	String		Move	...OrderNumber	
OrderDate	String		Move	...OrderDate	
CustomerId	String		Move	...CustomerNum	
⊞ CustomerAddress	Addres	[Local]			
SalesLevel	String				
DistributionChannel	String				
DivisionLevel	String				
SalesRegion	String				
SalesBranch	String				
⊞ AssociatedCustom	Custom	[Local]			
PONumber	String		Move	...WorkOrderNu	
PODate	String		Move	...ExpectedDeliv	
⊞ OrderContactRef	Contact	[Local]			
OrderType	String		Move	...RejectOrder	
⊞ DocumentReferenc	Docum	[Local]			
OrderStatus	String		Move	...OrderStatus	
OrderPriority	String				

Figure 7-29 Testing the map

To test the map, click the **Test** icon circled in Figure 7-29. A new window to connect to the InterChange Server opens.

Enter the options to connect to the InterChange Server and select the options **Deploy map** and **Deploy dependent objects,** as shown in Figure 7-30. The second option would deploy the submap and the application-specific object related with this map.


Connect to IBM WebSphere InterChange Server

Connect to Server

Server Name: ICS

User Name: admin

Password: ****

[✓] Remember user name and password

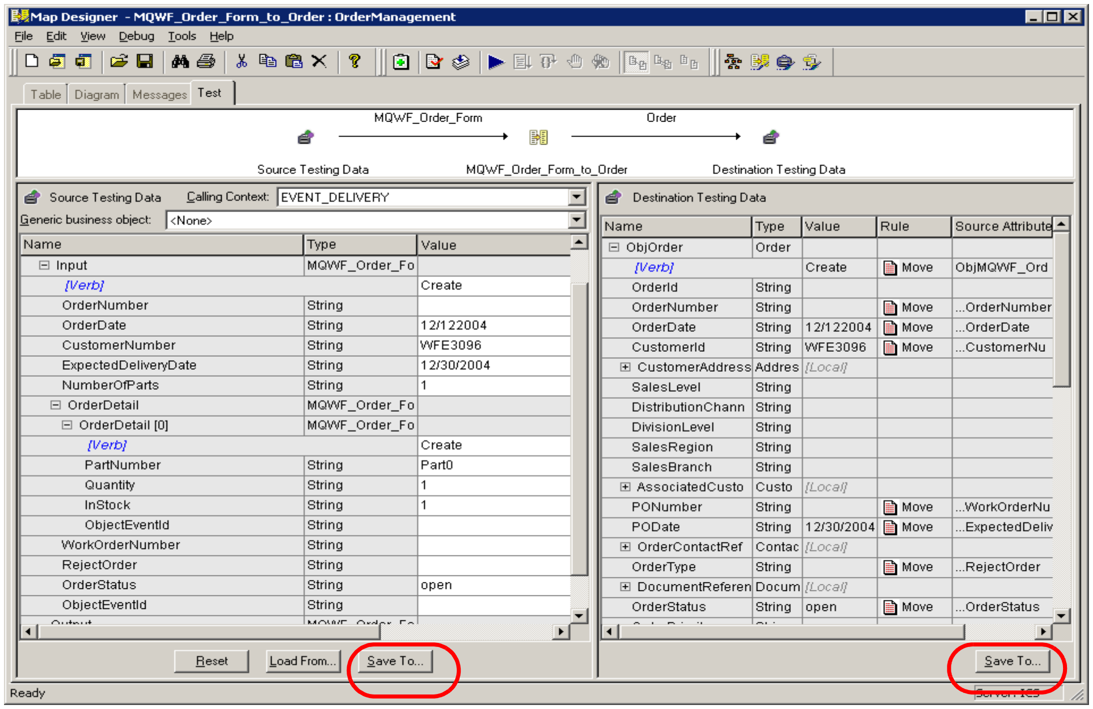
Deployment

[✓] Deploy map

[✓] Deploy dependent objects

Figure 7-30 Test the maps

Figure 7-31 shows the result of applying the map to our input data.



Source Testing Data

Calling Context: EVENT_DELIVERY

Generic business object: <None>

Name	Type	Value
⊟ Input	MQWF_Order_Fo	
[Verb]		Create
OrderNumber	String	
OrderDate	String	12/122004
CustomerNumber	String	WFE3096
ExpectedDeliveryDate	String	12/30/2004
NumberOfParts	String	1
⊟ OrderDetail	MQWF_Order_Fo	
⊟ OrderDetail [0]	MQWF_Order_Fo	
[Verb]		Create
PartNumber	String	Part0
Quantity	String	1
InStock	String	1
ObjectEventId	String	
WorkOrderNumber	String	
RejectOrder	String	
OrderStatus	String	open
ObjectEventId	String	

Destination Testing Data

Name	Type	Value	Rule	Source Attribute
⊟ ObjOrder	Order			
[Verb]		Create	Move	ObjMQWF_Ord
OrderId	String			
OrderNumber	String		Move	...OrderNumber
OrderDate	String	12/122004	Move	...OrderDate
CustomerId	String	WFE3096	Move	...CustomerNu
⊞ CustomerAddress	Addres	*[Local]*		
SalesLevel	String			
DistributionChann	String			
DivisionLevel	String			
SalesRegion	String			
SalesBranch	String			
⊞ AssociatedCusto	Custo	*[Local]*		
PONumber	String		Move	...WorkOrderNu
PODate	String	12/30/2004	Move	...ExpectedDeliv
⊞ OrderContactRef	Contac	*[Local]*		
OrderType	String		Move	...RejectOrder
⊞ DocumentReferen	Docum	*[Local]*		
OrderStatus	String	open	Move	...OrderStatus

Figure 7-31 Finished testing maps

Save both business objects to file for use in later unit testing.

MQWF_Completed_Order_Form_to_Order

This maps the information that is contained in the application-specific object MQWF_Completed_Order_Form to the generic object Order. This map is to cater for the MQWorkflow Delete_Order and Confirm_Order activities.

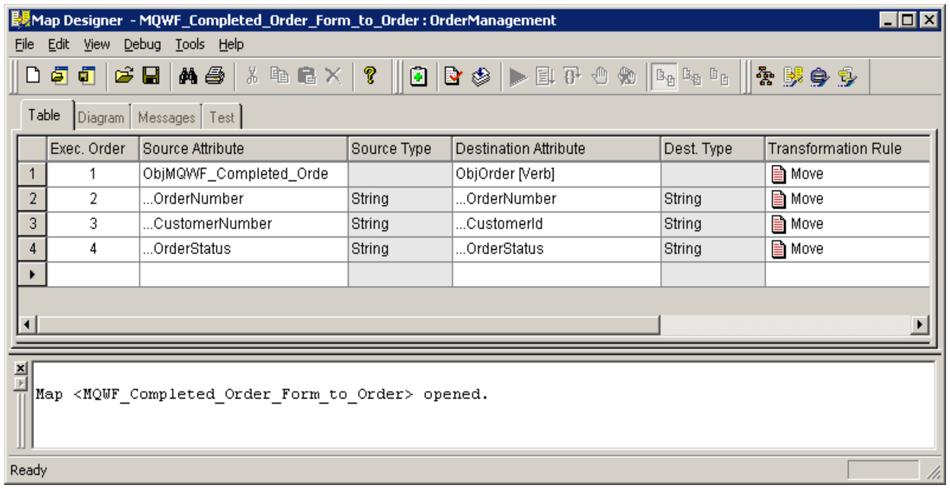
To add MQWF_Completed_Order_Form, follow the instructions in 7.6.1, “From ASBO MQWF_Order_Form to GBO Order” on page 276. Use the data in this list as well as Table 7-5 on page 288.

Source business object	`MQWF_Completed_Order_Form`
Destination business object	`Order`
Map name	`MQWF_Completed_Order_Form_to_Order`
Mapping direction	`Application-Specific to Generic`

Table 7-5 Mapping in MQWF_Completed_Order_Form_to_Order

Source	Destination
Verb	Verb
OrderNumber	OrderNumber
CustomerNumber	CustomerId
OrderStatus	OrderStatus

The resulting map is shown in Figure 7-32.


Map Designer - MQWF_Completed_Order_Form_to_Order : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test
Ready

	Exec. Order	Source Attribute	Source Type	Destination Attribute	Dest. Type	Transformation Rule
1	1	ObjMQWF_Completed_Orde		ObjOrder [Verb]		Move
2	2	...OrderNumber	String	...OrderNumber	String	Move
3	3	...CustomerNumber	String	...CustomerId	String	Move
4	4	...OrderStatus	String	...OrderStatus	String	Move

```
Map <MQWF_Completed_Order_Form_to_Order> opened.
```

Figure 7-32 Completed map

7.6.2 From the GBO order to the ASBO JDBC_ORDERS

This section discusses the development of the main and submap to transform the Order generic business object to the ASBO JDBC. Figure 7-33 shows these maps within the overall solution.

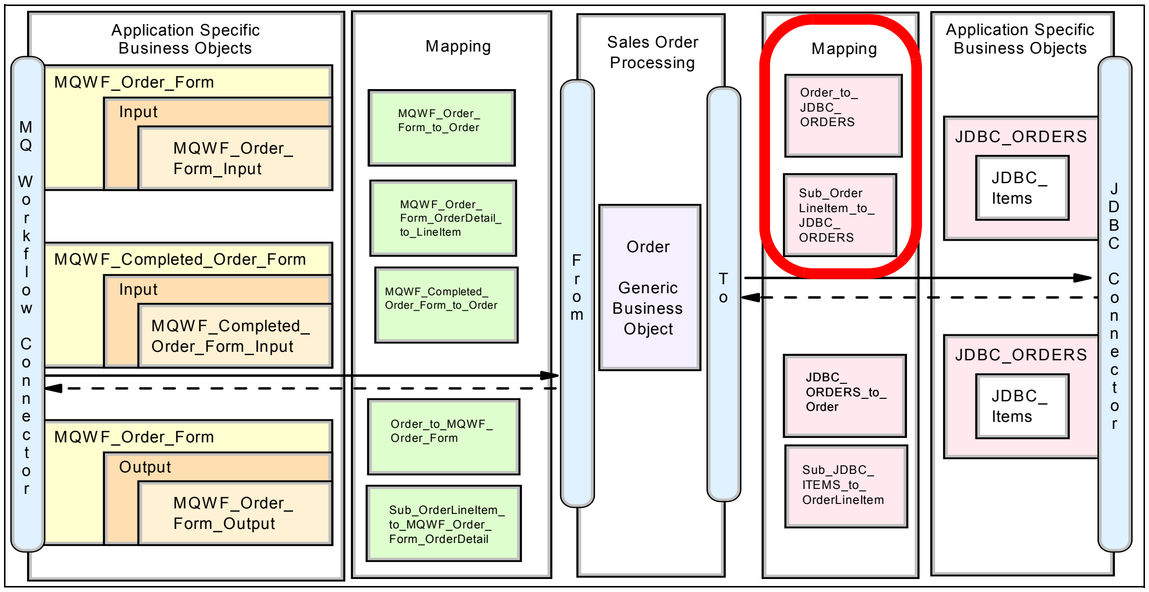


Figure 7-33 Mapping the Order business object to JDBC_Orders

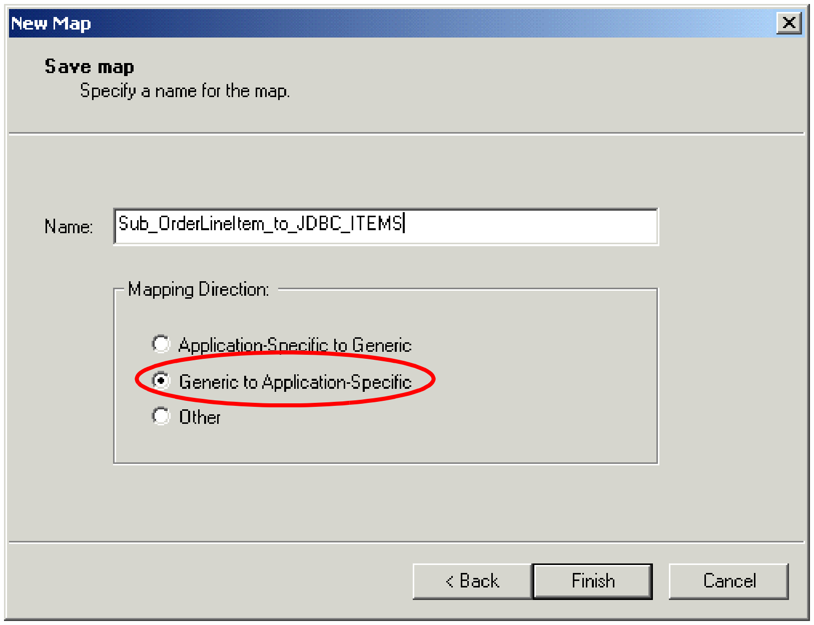
Sub_OrderLineItem_to_JDBC_ITEMS

This submap maps the information that is contained in the generic object OrderLineItem to the application-specific object JDBC_ITEMS.

To add Sub_OrderLineItem_to_JDBC-ITEMS, follow the instructions in 7.6.1, “From ASBO MQWF_Order_Form to GBO Order” on page 276. Use the data in this list as well as Table 7-6 on page 290.

Source business object	`OrderLineItem`
Destination business object	`JDBC_ITEMS`
Map name	`Sub_OrderLineItem_to_JDBC_ITEMS`
Mapping direction	`Generic to Application-Specific`

Make sure to select the correct mapping direction (**Generic to Application Specific**), as shown in Figure 7-34.



Name: Sub_OrderLineItem_to_JDBC_ITEMS

Mapping Direction:
- Application-Specific to Generic
- [✓ Generic to Application-Specific]
- Other

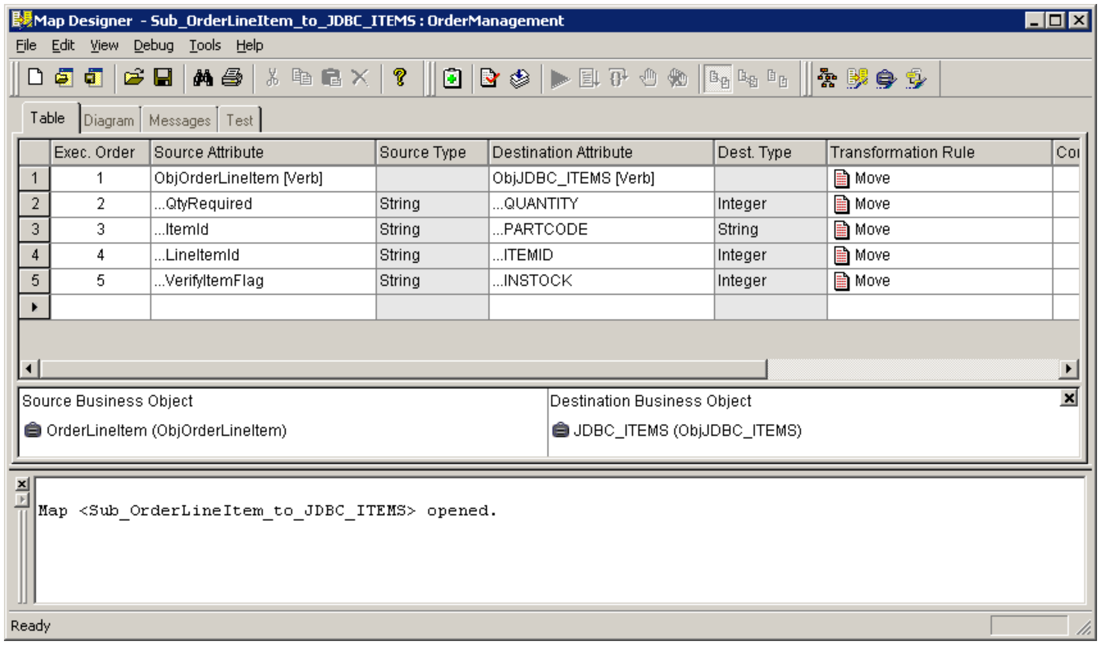
Figure 7-34 Direction of a map

The mapping specification can be seen in Table 7-6.

Table 7-6 Mapping in Sub_OrderLineItem_to_JDBC_ITEMS

Source	**Destination**
Verb	Verb
QtyRequired	QUANTITY
ItemId	PARTCODE
LineItemId	ITEMID
VerifyItemFlag	INSTOCK

Figure 7-35 shows the Map Designer after all of the mapping has been done. Save and compile the submap.


Map Designer - Sub_OrderLineItem_to_JDBC_ITEMS : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test

	Exec. Order	Source Attribute	Source Type	Destination Attribute	Dest. Type	Transformation Rule	Co
1	1	ObjOrderLineItem [Verb]		ObjJDBC_ITEMS [Verb]		Move	
2	2	...QtyRequired	String	...QUANTITY	Integer	Move	
3	3	...ItemId	String	...PARTCODE	String	Move	
4	4	...LineItemId	String	...ITEMID	Integer	Move	
5	5	...VerifyItemFlag	String	...INSTOCK	Integer	Move	

Source Business Object	Destination Business Object
OrderLineItem (ObjOrderLineItem)	JDBC_ITEMS (ObjJDBC_ITEMS)

```
Map <Sub_OrderLineItem_to_JDBC_ITEMS> opened.
```

Ready

Figure 7-35 Map the attributes

Order_to_JDBC_ORDERS

This map maps the information contained in the generic object Order to the application-specific object JDBC_ORDERS. This map uses the submap Sub_OrderLineItem_to_JDBC_ITEMS that we created in a previous section.

To create Order_to_JDBC_ORDERS, follow the instructions in 7.6.1, "From ASBO MQWF_Order_Form to GBO Order" on page 276. Use the data in this list as well as Table 7-7 on page 292.

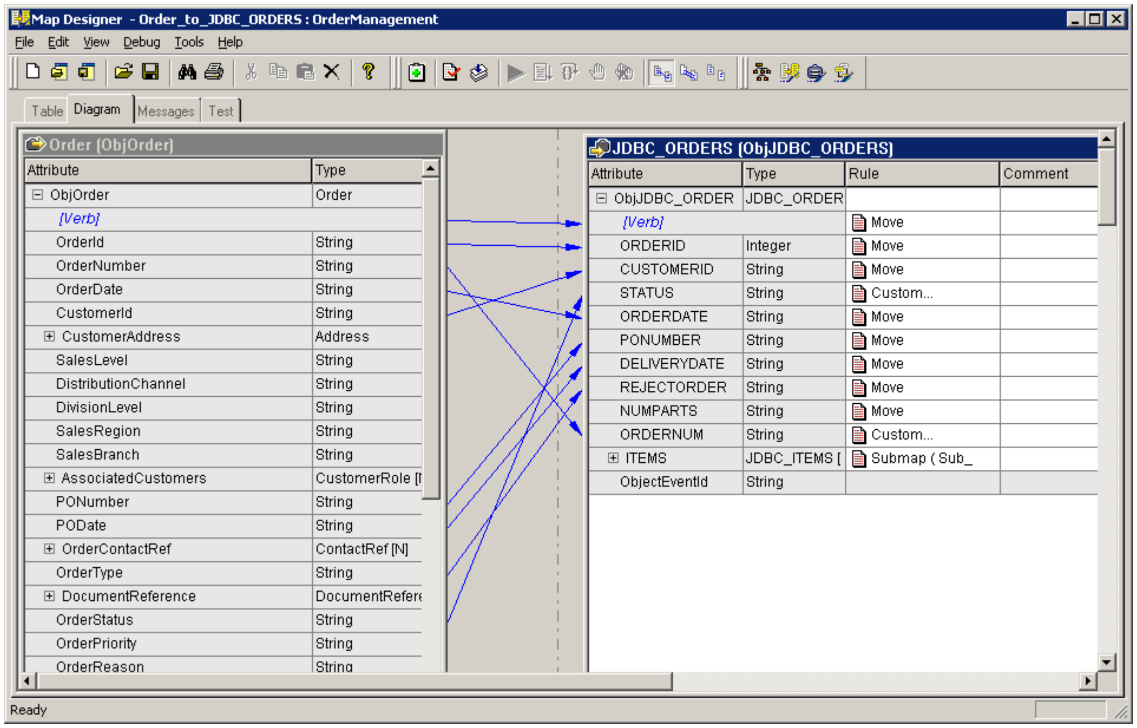
Source business object	`Order`
Destination business object	`JDBC_ORDERS`
Map name	`Order_to_JDBC_ORDERS`
Mapping direction	`Generic to Application-Specific`

The maps can be seen in Table 7-7.

Table 7-7 Mapping in Order_to_JDBC_ORDERS

Source	**Destination**
Verb	Verb
CustomerId	CUSTOMERID
OrderId	ORDERNUMBER
OrderStatus	STATUS
OrderDate	ORDERDATE
PONumber	PONUMBER
Notes	NUMPARTS
OrderType	REJECTORDER
PODate	DELIVERYDATE
OrderLineItem	ITEMS

Figure 7-36 on page 293 shows the Map Designer after all of the mapping has been completed. Save and compile the map. Note the different color for the rule Custom for attribute STATUS, indicating that custom code has been written for this mapping.



Map Designer - Order_to_JDBC_ORDERS : OrderManagement

File Edit View Debug Tools Help

Table | Diagram | Messages | Test

Order (ObjOrder)

Attribute	Type
⊟ ObjOrder	Order
[Verb]	
OrderId	String
OrderNumber	String
OrderDate	String
CustomerId	String
⊞ CustomerAddress	Address
SalesLevel	String
DistributionChannel	String
DivisionLevel	String
SalesRegion	String
SalesBranch	String
⊞ AssociatedCustomers	CustomerRole [I
PONumber	String
PODate	String
⊞ OrderContactRef	ContactRef [N]
OrderType	String
⊞ DocumentReference	DocumentRefere
OrderStatus	String
OrderPriority	String
OrderReason	String

JDBC_ORDERS (ObjJDBC_ORDERS)

Attribute	Type	Rule	Comment
⊟ ObjJDBC_ORDER	JDBC_ORDER		
[Verb]		Move	
ORDERID	Integer	Move	
CUSTOMERID	String	Move	
STATUS	String	Custom...	
ORDERDATE	String	Move	
PONUMBER	String	Move	
DELIVERYDATE	String	Move	
REJECTORDER	String	Move	
NUMPARTS	String	Move	
ORDERNUM	String	Custom...	
⊞ ITEMS	JDBC_ITEMS [	Submap (Sub_	
ObjectEventId	String		

Ready

Figure 7-36 Mapping the attributes

The STATUS field in the destination business object has a particular behavior depending on the Order business object OrderStatus value. If the OrderStatus is empty or null, the new STATUS field will be set to NEW. If the OrderStatus contains non empty value, the new STATUS field will be set to the OrderStatus value.

To create this behavior, we create a custom rule:

1. Specify **Custom** as the rule for the STATUS attribute. The custom rule can be created by entering Java code or with a graphical interface. In our scenario, we use graphical interface to create the custom rule. To enter the code, right-click in the rule and select **Open**.

2. The graphical editor is opened by double-clicking a custom rule. Figure 7-37 shows the GUI interface with graphics that create the custom rule.

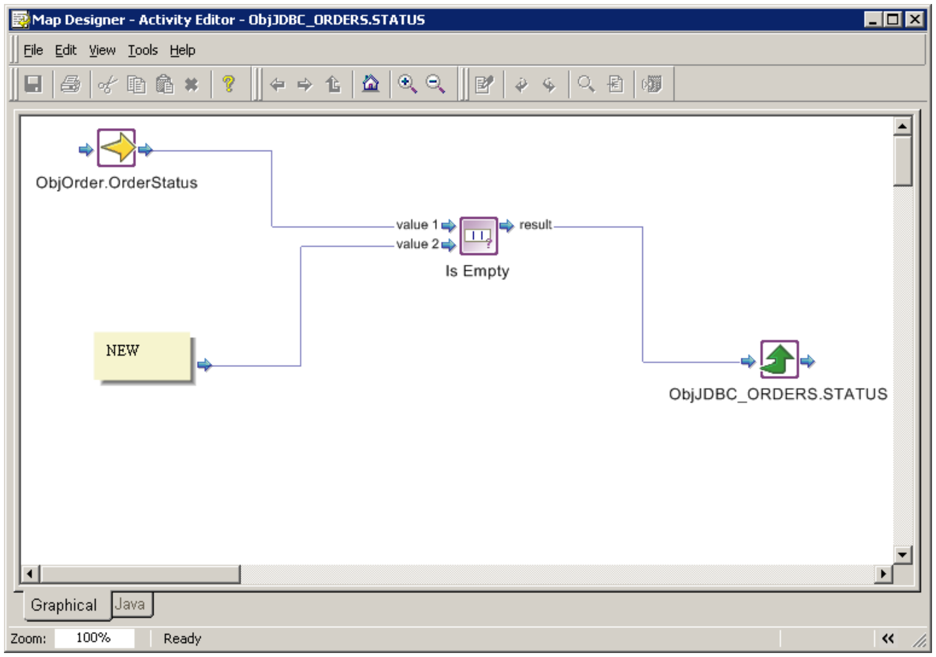


Figure 7-37 Creating a custom rule

The ORDERNUM field destination business object follows a custom mapping rule. If the OrderNumber is not empty (has a value), the new ORDERNUM field will be set to OrderNumber value. If the OrderNumber is empty, the new ORDERNUM field will be set CustomerId appended with current date and time stamp.

Figure 7-38 shows the GUI interface with graphics for the ORDERNUM mapping rule.

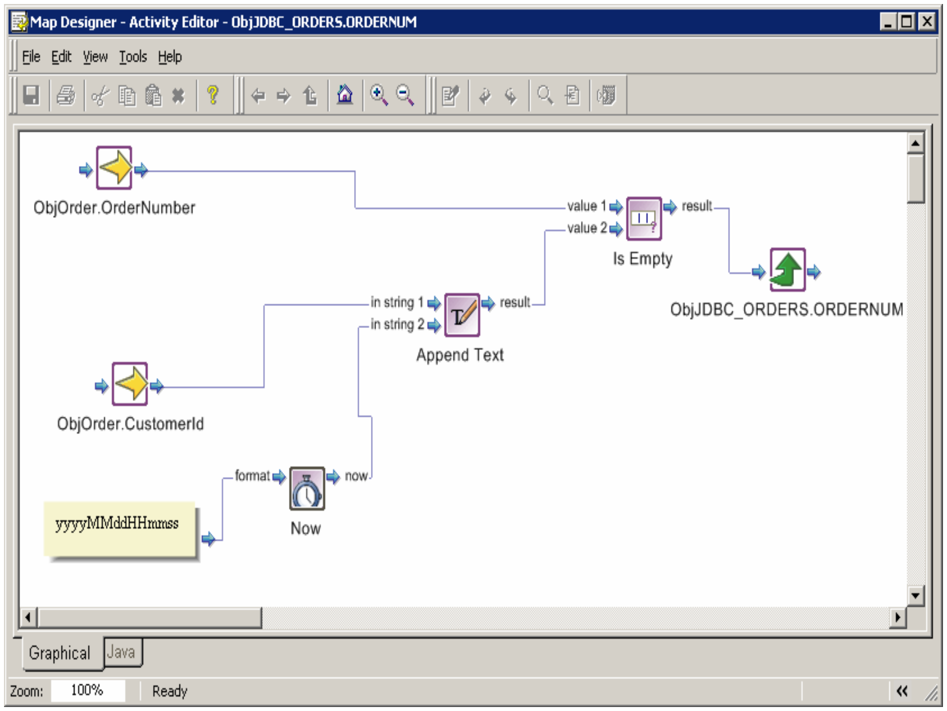


Figure 7-38   Custom rule map for ORDERNUM field

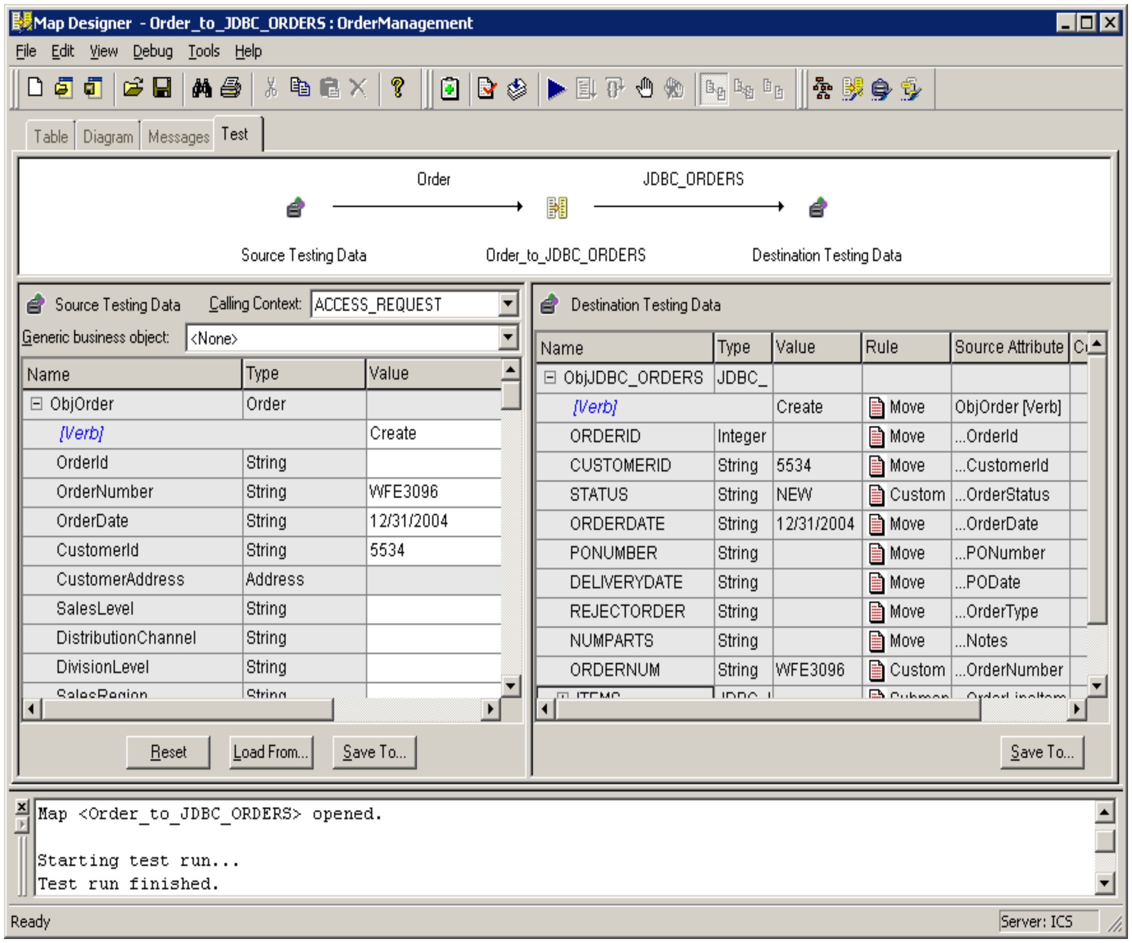
Testing the map

The final task before finishing a map is to test the map. To test the map, click the corresponding tab in the Map Designer.

We can use the resulting business object from our previous map testing as our source destination data. To use the file we saved, select the source business object, click **Load From** to browse for the file, and select **OK**. Now we have our input data.

Select **ACCESS_REQUEST** as the Calling Context in the Source Testing Data pane, and test the map.

In Figure 7-39, we can see the result of applying the map to our input data.



Source Testing Data Calling Context: ACCESS_REQUEST

Generic business object: <None>

Name	Type	Value
⊟ ObjOrder	Order	
[Verb]		Create
OrderId	String	
OrderNumber	String	WFE3096
OrderDate	String	12/31/2004
CustomerId	String	5534
CustomerAddress	Address	
SalesLevel	String	
DistributionChannel	String	
DivisionLevel	String	
SalesRegion	String	

Destination Testing Data

Name	Type	Value	Rule	Source Attribute
⊟ ObjJDBC_ORDERS	JDBC_			
[Verb]		Create	Move	ObjOrder [Verb]
ORDERID	Integer		Move	...OrderId
CUSTOMERID	String	5534	Move	...CustomerId
STATUS	String	NEW	Custom	...OrderStatus
ORDERDATE	String	12/31/2004	Move	...OrderDate
PONUMBER	String		Move	...PONumber
DELIVERYDATE	String		Move	...PODate
REJECTORDER	String		Move	...OrderType
NUMPARTS	String		Move	...Notes
ORDERNUM	String	WFE3096	Custom	...OrderNumber

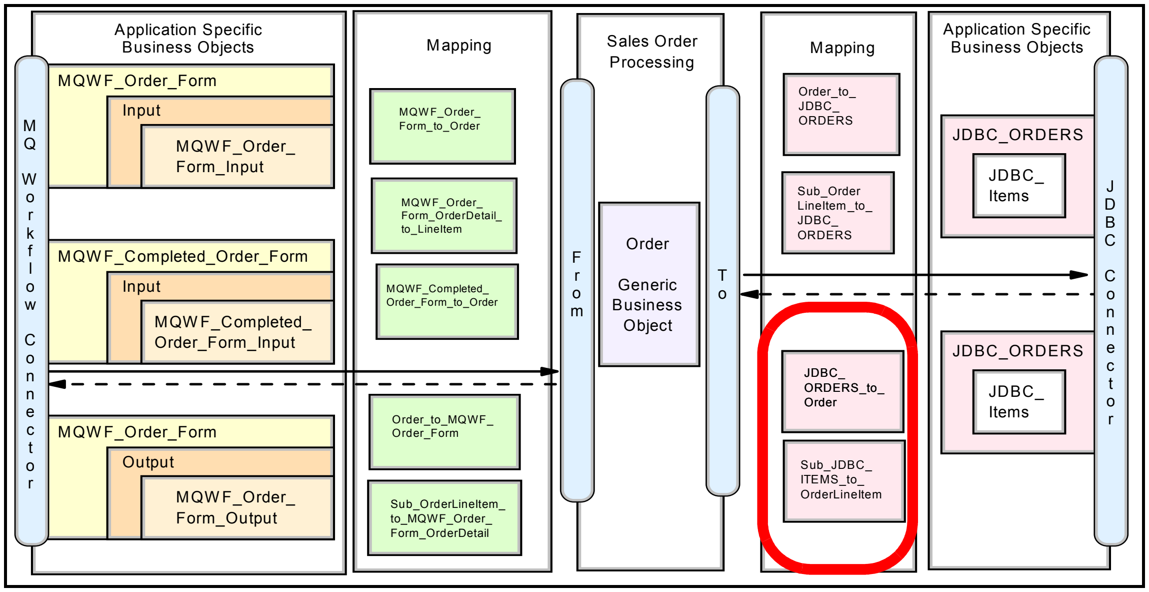
```
Map <Order_to_JDBC_ORDERS> opened.

Starting test run...
Test run finished.
```

Figure 7-39 Finished testing maps

Save both business objects to a file to use them in later unit testing.

7.6.3 From ASBO JDBC_ORDERS to GBO Order

This section discusses the development of the main and submap to transform the JDBC ASBO to the generic business object Order. Figure 7-40 shows these maps within the overall solution.



Application Specific Business Objects
MQ Workflow Connector
MQWF_Order_Form
Input
MQWF_Order_Form_Input
MQWF_Completed_Order_Form
Input
MQWF_Completed_Order_Form_Input
MQWF_Order_Form
Output
MQWF_Order_Form_Output
Mapping
MQWF_Order_Form_to_Order
MQWF_Order_Form_OrderDetail_to_LineItem
MQWF_Completed_Order_Form_to_Order
Order_to_MQWF_Order_Form
Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail
Sales Order Processing
From
Order Generic Business Object
To
Mapping
Order_to_JDBC_ORDERS
Sub_Order LineItem_to_JDBC_ORDERS
JDBC_ORDERS_to_Order
Sub_JDBC_ITEMS_to_OrderLineItem
Application Specific Business Objects
JDBC_ORDERS
JDBC_Items
JDBC_ORDERS
JDBC_Items
JDBC Connector

Figure 7-40 Mapping the business object JDBC_Orders to Order

Sub_JDBC_ITEMS_to_OrderLineItem

This submap maps the information contained in the application-specific object JDBC_ITEMS to the generic object OrderLineItem.

To create Sub_JDBC_ITEMS_to_OrderLineItem, follow the instructions in 7.6.1, “From ASBO MQWF_Order_Form to GBO Order” on page 276. Use the data in this list as well as Table 7-8:

Source business object	`JDBC_ITEMS`
Destination business object	`OrderLineItem`
Map name	`Sub_JDBC_ITEMS_to_OrderLineItem`
Mapping direction	`Application-Specific to Generic`

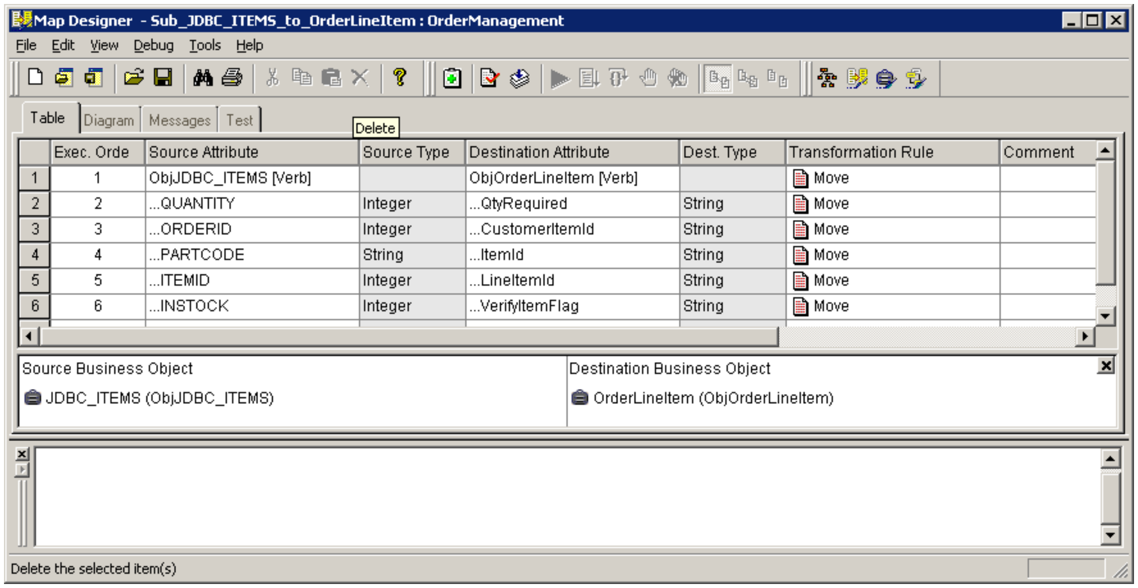
The map specification is shown in Table 7-8.

Table 7-8 Mapping in Sub_JDBC_ITEMS_to_OrderLineItem

Source	Destination
Verb	Verb

Source	Destination
QUANTITY	QtyRequired
ORDERID	CustomerItemId
PARTCODE	ItemId
ITEMID	LineItemId
INSTOCK	VerityItemFlag

Figure 7-41 shows the Map Designer for Sub_JDBC_ITEMS_to_OrderLineItem


Map Designer - Sub_JDBC_ITEMS_to_OrderLineItem : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test
Delete

	Exec. Orde	Source Attribute	Source Type	Destination Attribute	Dest. Type	Transformation Rule	Comment
1	1	ObjJDBC_ITEMS [Verb]		ObjOrderLineItem [Verb]		Move	
2	2	...QUANTITY	Integer	...QtyRequired	String	Move	
3	3	...ORDERID	Integer	...CustomerItemId	String	Move	
4	4	...PARTCODE	String	...ItemId	String	Move	
5	5	...ITEMID	Integer	...LineItemId	String	Move	
6	6	...INSTOCK	Integer	...VerifyItemFlag	String	Move	

Source Business Object	Destination Business Object
JDBC_ITEMS (ObjJDBC_ITEMS)	OrderLineItem (ObjOrderLineItem)

Delete the selected item(s)

Figure 7-41 Completed map

JDBC_ORDERS_to_Order

This map maps the information that is contained in the application-specific object JDBC_ORDERS to the generic object Order. This map uses the submap Sub_JDBC_ITEMS_to_OrderLineItem that was created before.

To create JDBC_ORDERS_to_Order, follow the instructions in 7.6.1, “From ASBO MQWF_Order_Form to GBO Order” on page 276. Use the data in this list as well as Table 7-9 on page 299:

Source business object	`JDBC_ORDERS`
Destination business object	`Order`
Map name	`JDBC_ORDERS_to_Order`

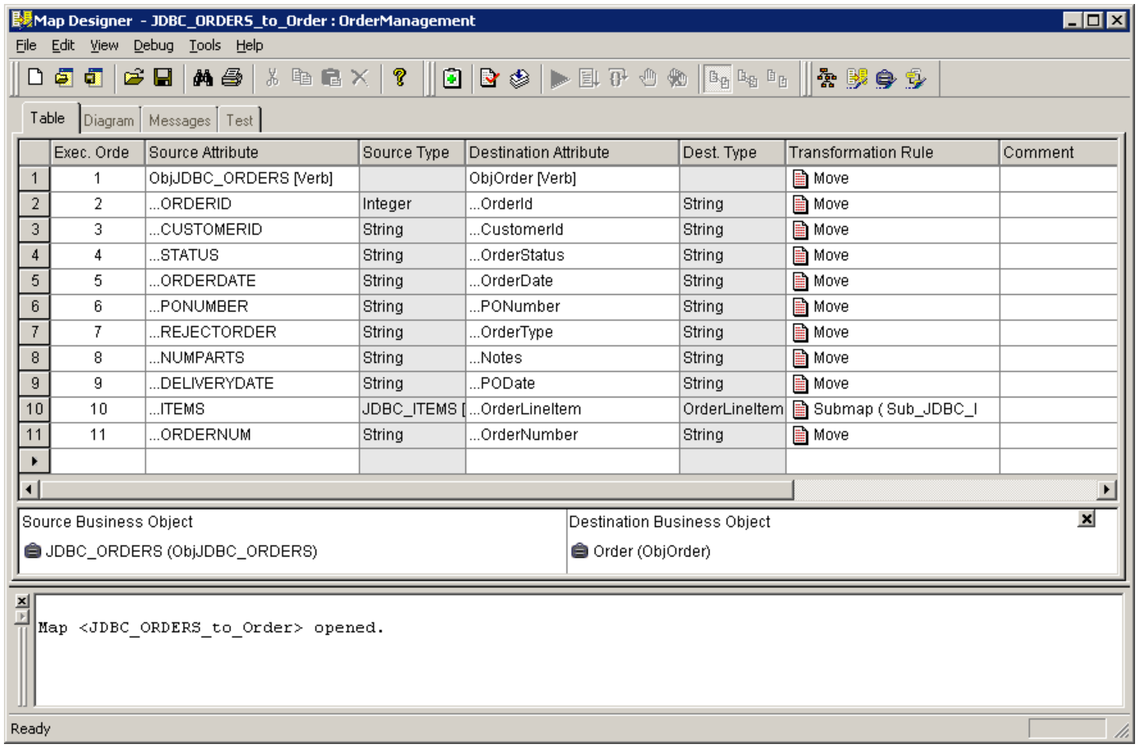
Mapping direction	`Application-Specific to Generic`

The maps can be seen in Table 7-9.

Table 7-9 Mapping in Order_to_JDBC_ORDERS

Source	Destination
Verb	Verb
ORDERID	OrderId
CUSTOMERID	CustomerId
STATUS	OrderStatus
ORDERDATE	OrderDate
PONUMBER	PONumber
REJECTORDER	RejectOrder
NUMPARTS	Notes
DELIVERYDATE	PODate
ITEMS	OrderLineItem
ORDERNUM	OrderNumber

Figure 7-42 shows the Map Designer for JDBC_ORDERS_to_Order.



Map Designer - JDBC_ORDERS_to_Order : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test

	Exec. Orde	Source Attribute	Source Type	Destination Attribute	Dest. Type	Transformation Rule	Comment
1	1	ObjJDBC_ORDERS [Verb]		ObjOrder [Verb]		Move	
2	2	...ORDERID	Integer	...OrderId	String	Move	
3	3	...CUSTOMERID	String	...CustomerId	String	Move	
4	4	...STATUS	String	...OrderStatus	String	Move	
5	5	...ORDERDATE	String	...OrderDate	String	Move	
6	6	...PONUMBER	String	...PONumber	String	Move	
7	7	...REJECTORDER	String	...OrderType	String	Move	
8	8	...NUMPARTS	String	...Notes	String	Move	
9	9	...DELIVERYDATE	String	...PODate	String	Move	
10	10	...ITEMS	JDBC_ITEMS [	...OrderLineItem	OrderLineItem	Submap (Sub_JDBC_I	
11	11	...ORDERNUM	String	...OrderNumber	String	Move	

Source Business Object	Destination Business Object
JDBC_ORDERS (ObjJDBC_ORDERS)	Order (ObjOrder)

```
Map <JDBC_ORDERS_to_Order> opened.
```

Ready

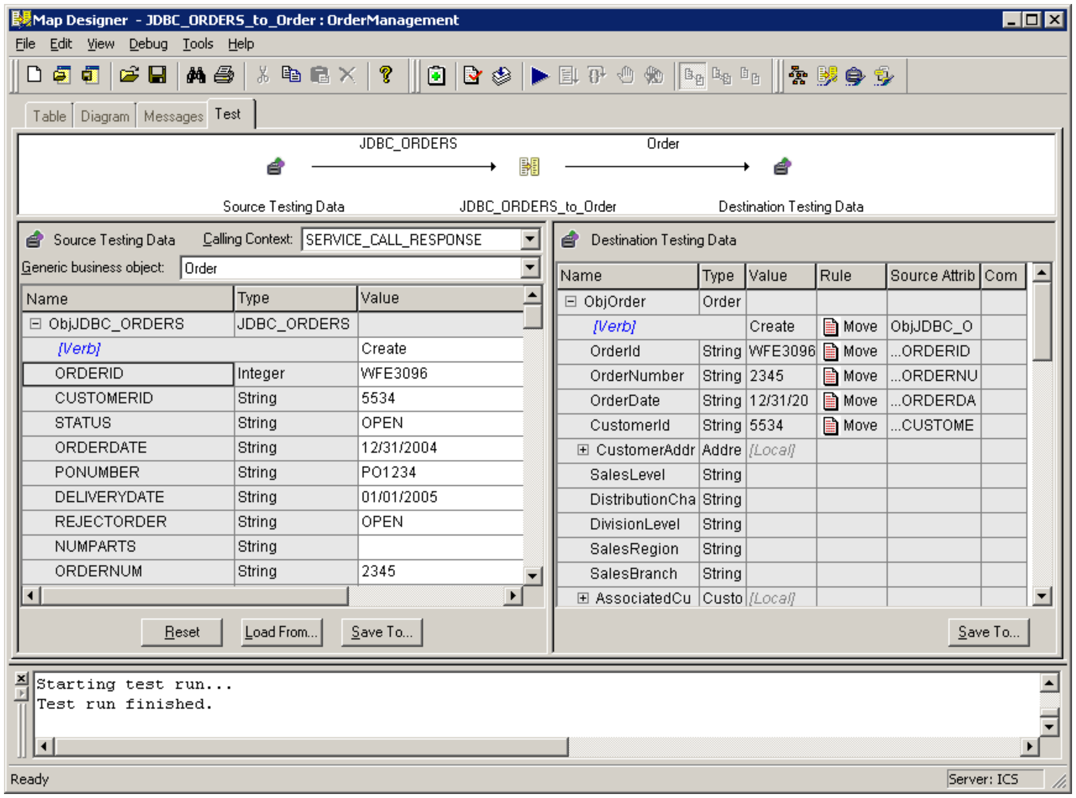
Figure 7-42 Map the attributes

Testing the map

To test the map, click the corresponding tab in the Map Designer.

Select **SERVICE_CALL_RESPONSE** as the Calling Context in the Source Testing Data pane, key in the input data and test the map.

Figure 7-43 shows the result of applying the map to our input data.



Source Testing Data — Calling Context: SERVICE_CALL_RESPONSE

Generic business object: Order

Name	Type	Value
⊟ ObjJDBC_ORDERS	JDBC_ORDERS	
[Verb]		Create
ORDERID	Integer	WFE3096
CUSTOMERID	String	5534
STATUS	String	OPEN
ORDERDATE	String	12/31/2004
PONUMBER	String	PO1234
DELIVERYDATE	String	01/01/2005
REJECTORDER	String	OPEN
NUMPARTS	String	
ORDERNUM	String	2345

Destination Testing Data

Name	Type	Value	Rule	Source Attrib	Com
⊟ ObjOrder	Order				
[Verb]		Create	Move	ObjJDBC_O	
OrderId	String	WFE3096	Move	...ORDERID	
OrderNumber	String	2345	Move	...ORDERNU	
OrderDate	String	12/31/20	Move	...ORDERDA	
CustomerId	String	5534	Move	...CUSTOME	
⊞ CustomerAddr	Addre	*[Local]*			
SalesLevel	String				
DistributionCha	String				
DivisionLevel	String				
SalesRegion	String				
SalesBranch	String				
⊞ AssociatedCu	Custo	*[Local]*			

```
Starting test run...
Test run finished.
```

Figure 7-43 Test the map

Save the destination business object to file.

7.6.4 From GBO Order to ASBO MQWF_Form_OrderDetail

This section discusses the development of the main and submap to transform the Order generic business object to the MQWF_Form_OrderDetail ASBO.

Figure 7-44 shows these maps within the overall solution.

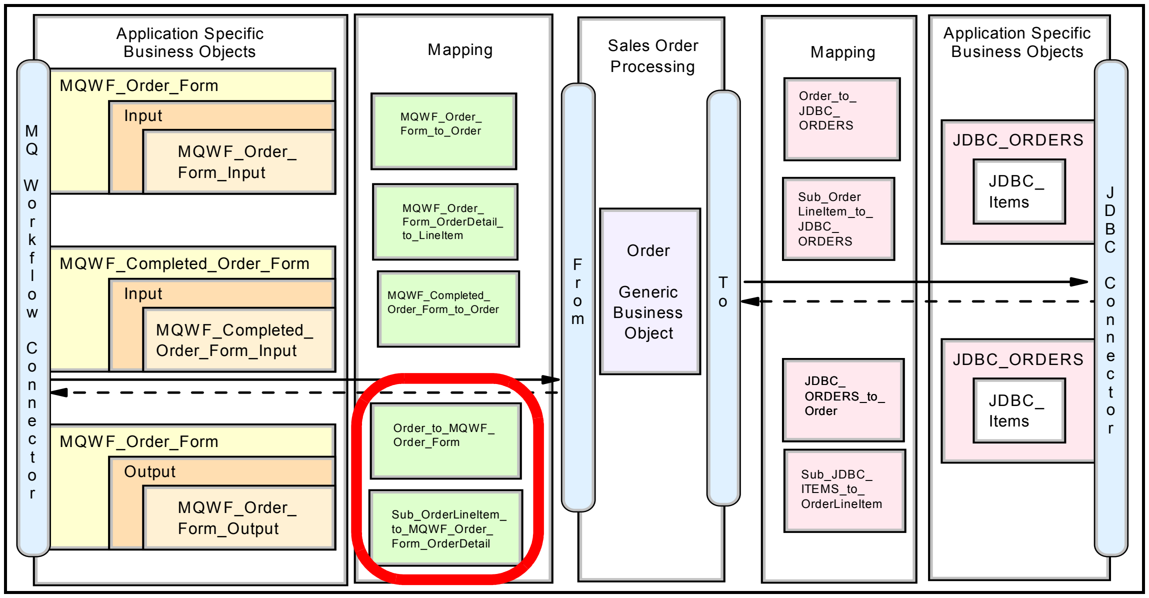

Application Specific Business Objects
MQ Workflow Connector
MQWF_Order_Form
Input
MQWF_Order_Form_Input
MQWF_Completed_Order_Form
Input
MQWF_Completed_Order_Form_Input
MQWF_Order_Form
Output
MQWF_Order_Form_Output
Mapping
MQWF_Order_Form_to_Order
MQWF_Order_Form_OrderDetail_to_LineItem
MQWF_Completed_Order_Form_to_Order
Order_to_MQWF_Order_Form
Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail
Sales Order Processing
From
Order Generic Business Object
To
Mapping
Order_to_JDBC_ORDERS
Sub_Order LineItem_to_JDBC_ORDERS
JDBC_ORDERS_to_Order
Sub_JDBC_ITEMS_to_OrderLineItem
Application Specific Business Objects
JDBC_ORDERS
JDBC_Items
JDBC_ORDERS
JDBC_Items
JDBC Connector

Figure 7-44 Mapping the generic business object Order to the ASBO MQWF_Form_OrderDetail

Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail

This submap maps the information that is contained in the OrderLineItem business object to the MQWF_Order_Form_OrderDetail business object.

To create Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail, follow the instructions in 7.6.1, "From ASBO MQWF_Order_Form to GBO Order" on page 276. Use the data in this list as well as Table 7-10 on page 303:

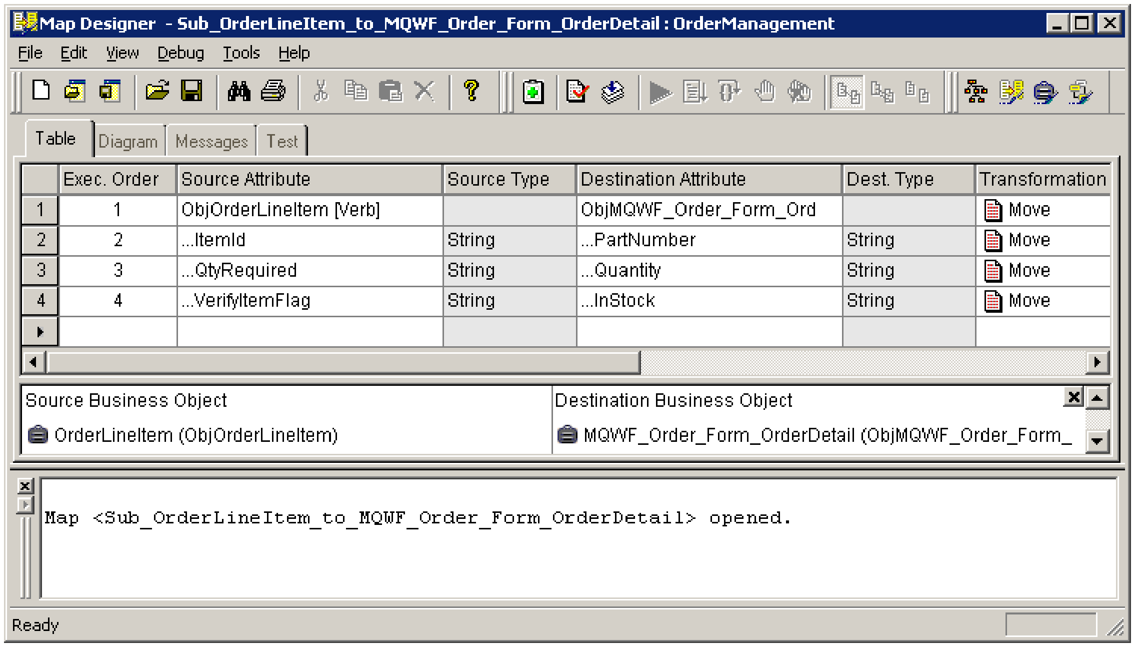
Source business object	`OrderLineItem`
Destination business object	`MQWF_Order_Form_OrderDetail`
Map name	`Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail`
Mapping direction	`Generic to Application-Specific`

The maps can be seen in Table 7-10.

Table 7-10 Mapping in Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail

Source	Destination
Verb	Verb
ItemId	PartNumber
QtyRequired	Quantity
VerifyItemFlag	InStock

Figure 7-45 shows the Map Designer.


Map Designer - Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test

	Exec. Order	Source Attribute	Source Type	Destination Attribute	Dest. Type	Transformation
1	1	ObjOrderLineItem [Verb]		ObjMQWF_Order_Form_Ord		Move
2	2	...ItemId	String	...PartNumber	String	Move
3	3	...QtyRequired	String	...Quantity	String	Move
4	4	...VerifyItemFlag	String	...InStock	String	Move

Source Business Object	Destination Business Object
OrderLineItem (ObjOrderLineItem)	MQWF_Order_Form_OrderDetail (ObjMQWF_Order_Form_

```
Map <Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail> opened.
```

Ready

Figure 7-45 Completed map

Order_to_MQWF_Order_Form

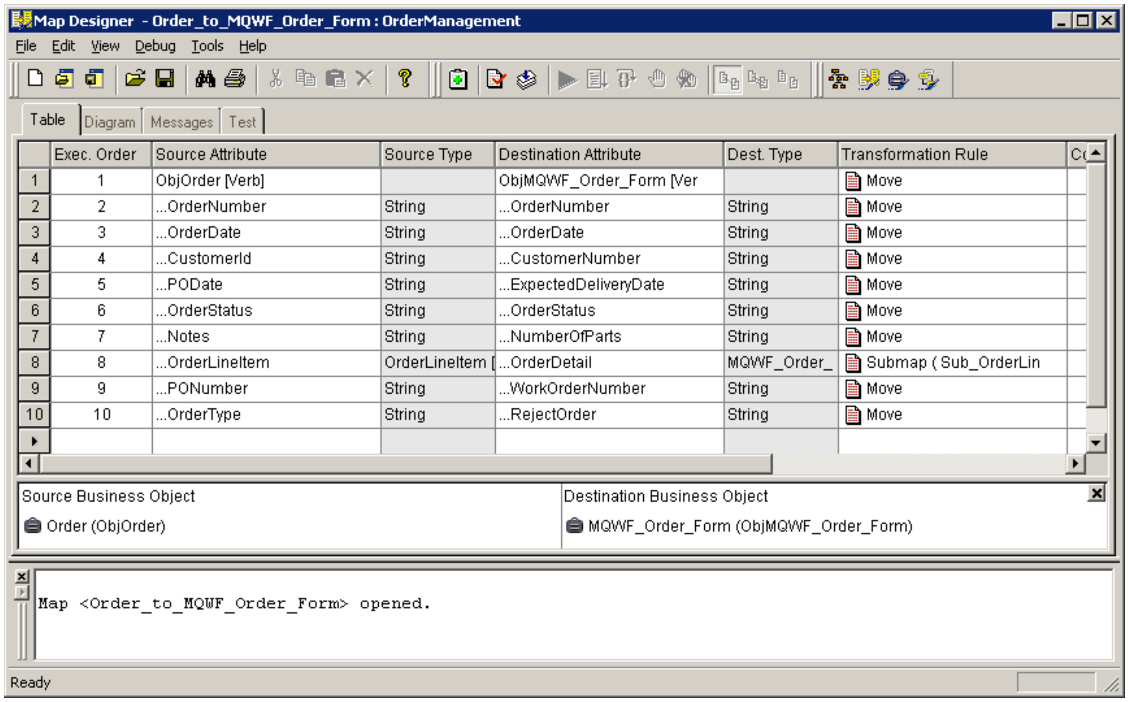
To create Order_to_MQWF_Order_Form, follow the instructions in 7.6.1, “From ASBO MQWF_Order_Form to GBO Order” on page 276. Use the data in this list as well as Table 7-11:

Source business object:	`Order`
Destination business object:	`MQWF_Order_Form`
Map name:	`Order_to_MQWF_Order_Form`
Mapping direction:	`Generic to Application-Specific`

Table 7-11 Mapping in Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail

Source	Destination
Verb	Verb
OrderNumber	OrderNumber
OrderDate	OrderDate
CustomerId	CustomerNumber
PODate	ExpectedDeliveryDate
OrderStatus	OrderStatus
Notes	NumberOfParts
OrderLineItem	OrderDetails
PONumber	WorkOrderNumber
OrderType	RejectOrder

Figure 7-46 shows the Map Designer after the mapping has been done.


Map Designer - Order_to_MQWF_Order_Form : OrderManagement
File Edit View Debug Tools Help
Table Diagram Messages Test

	Exec. Order	Source Attribute	Source Type	Destination Attribute	Dest. Type	Transformation Rule	Co
1	1	ObjOrder [Verb]		ObjMQWF_Order_Form [Ver		Move	
2	2	...OrderNumber	String	...OrderNumber	String	Move	
3	3	...OrderDate	String	...OrderDate	String	Move	
4	4	...CustomerId	String	...CustomerNumber	String	Move	
5	5	...PODate	String	...ExpectedDeliveryDate	String	Move	
6	6	...OrderStatus	String	...OrderStatus	String	Move	
7	7	...Notes	String	...NumberOfParts	String	Move	
8	8	...OrderLineItem	OrderLineItem [	...OrderDetail	MQWF_Order_	Submap (Sub_OrderLin	
9	9	...PONumber	String	...WorkOrderNumber	String	Move	
10	10	...OrderType	String	...RejectOrder	String	Move	

Source Business Object	Destination Business Object
Order (ObjOrder)	MQWF_Order_Form (ObjMQWF_Order_Form)

```
Map <Order_to_MQWF_Order_Form> opened.
```

Ready

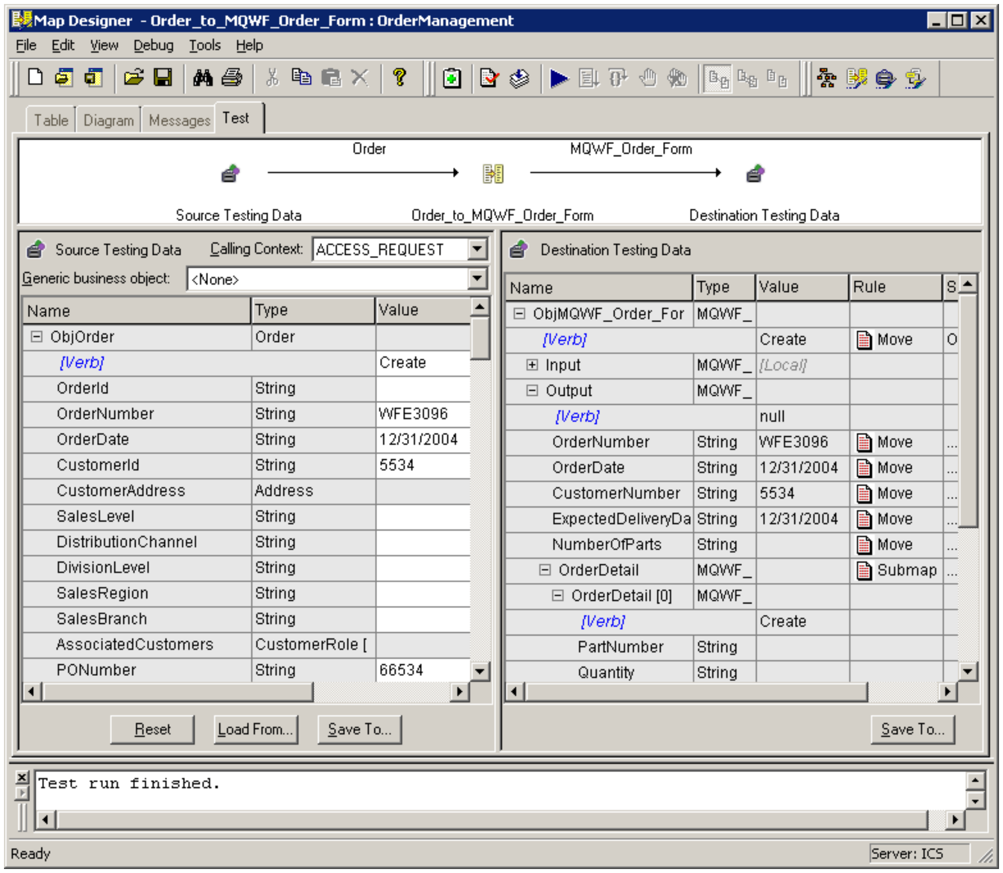
Figure 7-46 Completed map

Testing the map

To test the map, click the corresponding tab in the Map Designer.

We can use the resulting business object from our previous map testing as our source destination data. To use the file that we saved, select the source business object, click **Load From** to browse for the file, and select **OK.** Now we have our input data.

Figure 7-47 shows the result of applying the map to our input data.



Source Testing Data   Calling Context: ACCESS_REQUEST

Generic business object: <None>

Name	Type	Value
⊟ ObjOrder	Order	
[Verb]		Create
OrderId	String	
OrderNumber	String	WFE3096
OrderDate	String	12/31/2004
CustomerId	String	5534
CustomerAddress	Address	
SalesLevel	String	
DistributionChannel	String	
DivisionLevel	String	
SalesRegion	String	
SalesBranch	String	
AssociatedCustomers	CustomerRole [	
PONumber	String	66534

Destination Testing Data

Name	Type	Value	Rule	S
⊟ ObjMQWF_Order_For	MQWF_			
[Verb]		Create	Move	O
⊞ Input	MQWF_	*[Local]*		
⊟ Output	MQWF_			
[Verb]		null		
OrderNumber	String	WFE3096	Move	...
OrderDate	String	12/31/2004	Move	...
CustomerNumber	String	5534	Move	...
ExpectedDeliveryDa	String	12/31/2004	Move	...
NumberOfParts	String		Move	...
⊟ OrderDetail	MQWF_		Submap	...
⊟ OrderDetail [0]	MQWF_			
[Verb]		Create		
PartNumber	String			
Quantity	String			

```
Test run finished.
```

Figure 7-47 Test the map

Figure 7-48 shows all the maps that we have created for the collaboration.

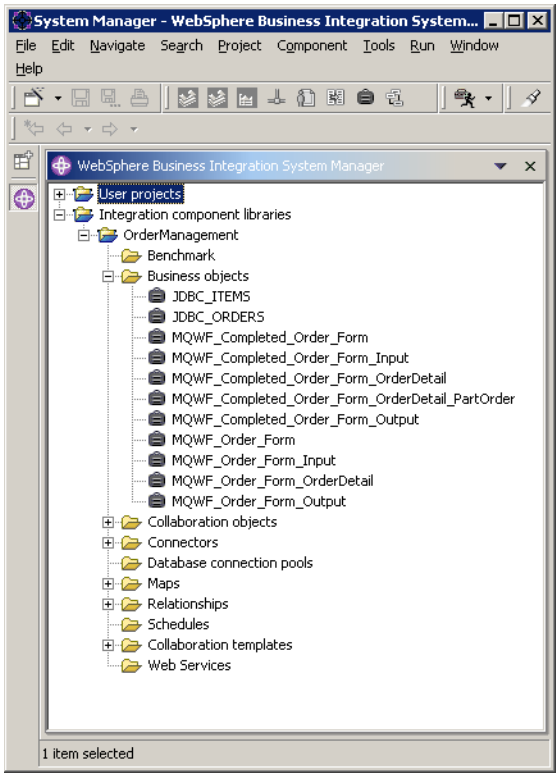

System Manager - WebSphere Business Integration System...
File Edit Navigate Search Project Component Tools Run Window Help
WebSphere Business Integration System Manager
User projects
Integration component libraries
OrderManagement
Benchmark
Business objects
JDBC_ITEMS
JDBC_ORDERS
MQWF_Completed_Order_Form
MQWF_Completed_Order_Form_Input
MQWF_Completed_Order_Form_OrderDetail
MQWF_Completed_Order_Form_OrderDetail_PartOrder
MQWF_Completed_Order_Form_Output
MQWF_Order_Form
MQWF_Order_Form_Input
MQWF_Order_Form_OrderDetail
MQWF_Order_Form_Output
Collaboration objects
Connectors
Database connection pools
Maps
Relationships
Schedules
Collaboration templates
Web Services
1 item selected

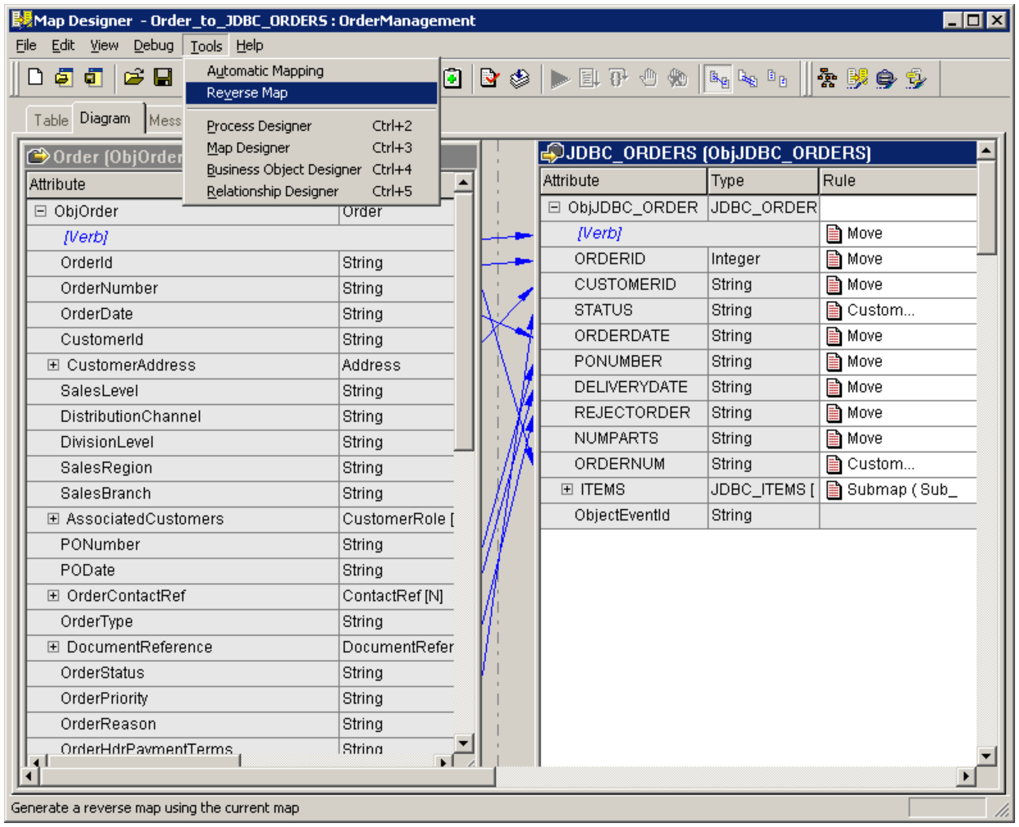
Figure 7-48 List of maps

Automatic and Reverse Mapping

The InterChange Server V4.3 Map Designer has two new features, called Automatic and Reverse Mapping. *Automatic* mapping allows for the mapping tool to compare the attributes in both the source and target business objects and automatically map between attributes that have matching names. There is also the ability to configure synonyms whereby you can associate ID in the source business object to Customer_ID in a target business object. *Reverse* mapping is exactly as it sounds, the ability to create a reverse map automatically.

In this section, we demonstrate a simple reverse mapping, using Order_to_JDBC_ORDERS map to generate JDBC_ORDERS_to_Order map.

Open the **Order_to_JDBC_ORDERS** map, then select **Tools** and **Reverse Mapping**.


Map Designer - Order_to_JDBC_ORDERS : OrderManagement
File
Edit
View
Debug
Tools
Help
Automatic Mapping
Reverse Map
Process Designer Ctrl+2
Map Designer Ctrl+3
Business Object Designer Ctrl+4
Relationship Designer Ctrl+5
Table
Diagram
Mess
Order (ObjOrder
JDBC_ORDERS (ObjJDBC_ORDERS)

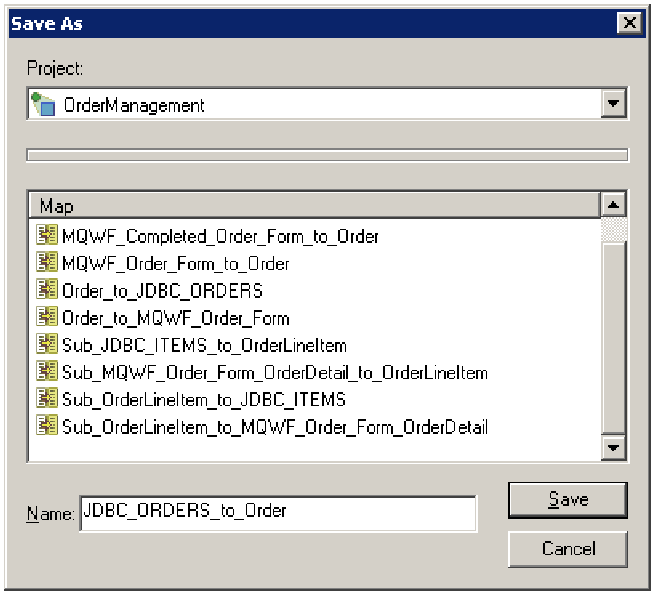
Attribute	
⊟ ObjOrder	Order
[Verb]	
OrderId	String
OrderNumber	String
OrderDate	String
CustomerId	String
⊞ CustomerAddress	Address
SalesLevel	String
DistributionChannel	String
DivisionLevel	String
SalesRegion	String
SalesBranch	String
⊞ AssociatedCustomers	CustomerRole [
PONumber	String
PODate	String
⊞ OrderContactRef	ContactRef [N]
OrderType	String
⊞ DocumentReference	DocumentRefer
OrderStatus	String
OrderPriority	String
OrderReason	String
OrderHdrPaymentTerms	String

Attribute	Type	Rule
⊟ ObjJDBC_ORDER	JDBC_ORDER	
[Verb]		Move
ORDERID	Integer	Move
CUSTOMERID	String	Move
STATUS	String	Custom...
ORDERDATE	String	Move
PONUMBER	String	Move
DELIVERYDATE	String	Move
REJECTORDER	String	Move
NUMPARTS	String	Move
ORDERNUM	String	Custom...
⊞ ITEMS	JDBC_ITEMS [	Submap (Sub_
ObjectEventId	String	

Generate a reverse map using the current map

Figure 7-49 Select reverse mapping

The map designer requests you to save the newly generated map.


Save As

Project:

OrderManagement

Map

MQWF_Completed_Order_Form_to_Order
MQWF_Order_Form_to_Order
Order_to_JDBC_ORDERS
Order_to_MQWF_Order_Form
Sub_JDBC_ITEMS_to_OrderLineItem
Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem
Sub_OrderLineItem_to_JDBC_ITEMS
Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail

Name: JDBC_ORDERS_to_Order

Figure 7-50 Saved the newly created map

At this time, you can see a warning message with the information in Example 7-5.

Example 7-5 Reverse mapping creation warning message

```
Starting Reverse Map...
Warning: Reverse transformation could not be performed for 'Custom'
tranformation from 'ObjOrder.OrderStatus'.
Warning: Reverse transformation could not be performed for 'Custom'
tranformation from 'ObjOrder.OrderNumber'.
Completed Reverse Map.
```

Note: It is important to note that the reverse mapping will not handle any Custom rule mapping and developer has to manually input the custom rule mapping if any.

7.7 Adapter configuration

In this section we describe the configuration of the two adapters that we use with our Order Management Collaboration. These adapters are the WebSphere MQ Workflow adapter and the JDBC adapter.

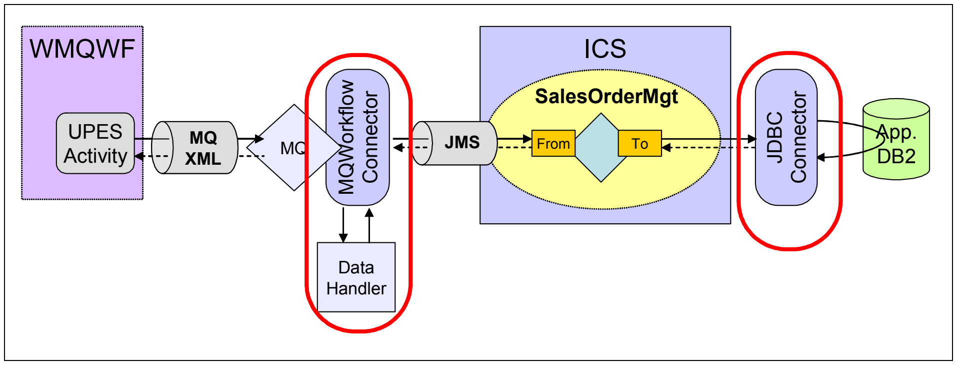


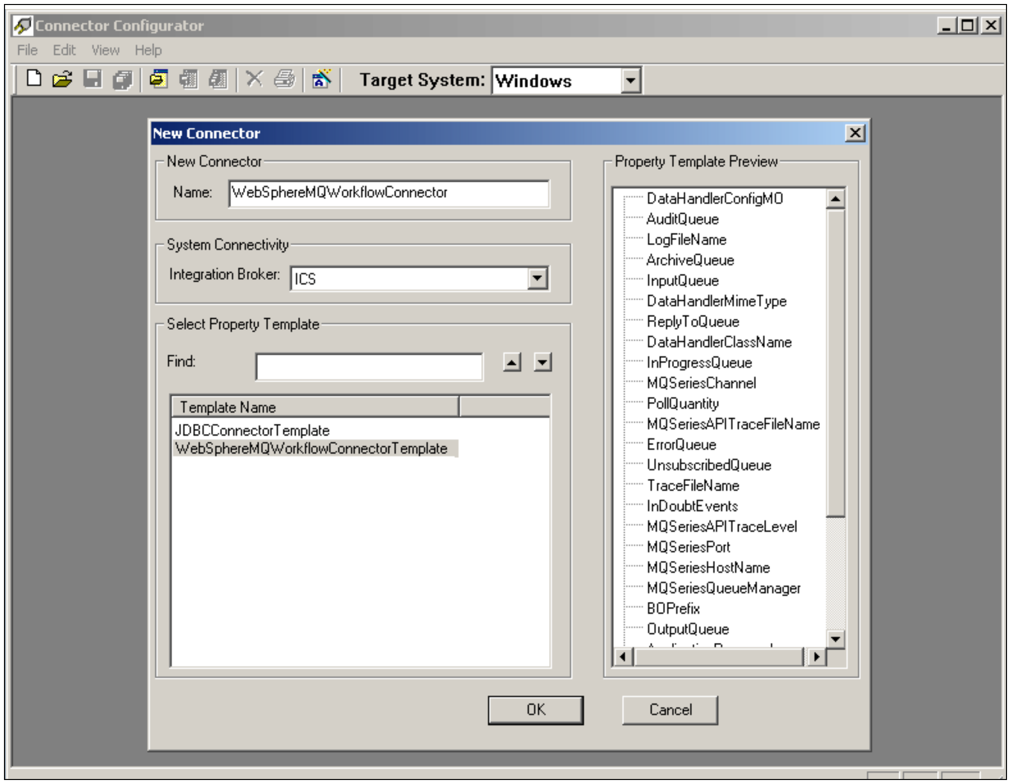
Figure 7-51 Use of adapters in solution

7.7.1 Importing the adapters into System Manager

In general, there are a few ways to start working with an adapter. Also, there are several different ways to define an adapter. The most common ways are as repository JAR files that you can import into the System Manager and as a template view within the Connector Configuration tool. The most productive alternative is to use the adapter templates as described in this section.

1. From within the System Manager, expand the Integration Component Library (ICL) you are working with and right-click the **Connector folder**.
2. From the list select **Create new connector**, which launches the Connector Configuration tool with the template selection window as shown in Figure 7-52 on page 311. Select the correct template, and type in the name of the new adapter, then select **OK**.

Note: If the adapter you want to work with is not in the adapter template list, you must import the adapter JAR file from the repository directory where you installed the WebSphere Business Integration Adapters. To do this, right-click the name of the ICL and select **Import from repository file** browse for the correct file name.


New Connector

Name: WebSphereMQWorkflowConnector

Integration Broker: ICS

Template Name
JDBCConnectorTemplate
WebSphereMQWorkflowConnectorTemplate

Property Template Preview
DataHandlerConfigMO
AuditQueue
LogFileName
ArchiveQueue
InputQueue
DataHandlerMimeType
ReplyToQueue
DataHandlerClassName
InProgressQueue
MQSeriesChannel
PollQuantity
MQSeriesAPITraceFileName
ErrorQueue
UnsubscribedQueue
TraceFileName
InDoubtEvents
MQSeriesAPITraceLevel
MQSeriesPort
MQSeriesHostName
MQSeriesQueueManager
BOPrefix
OutputQueue

Figure 7-52 Creating new adapter configuration with a template

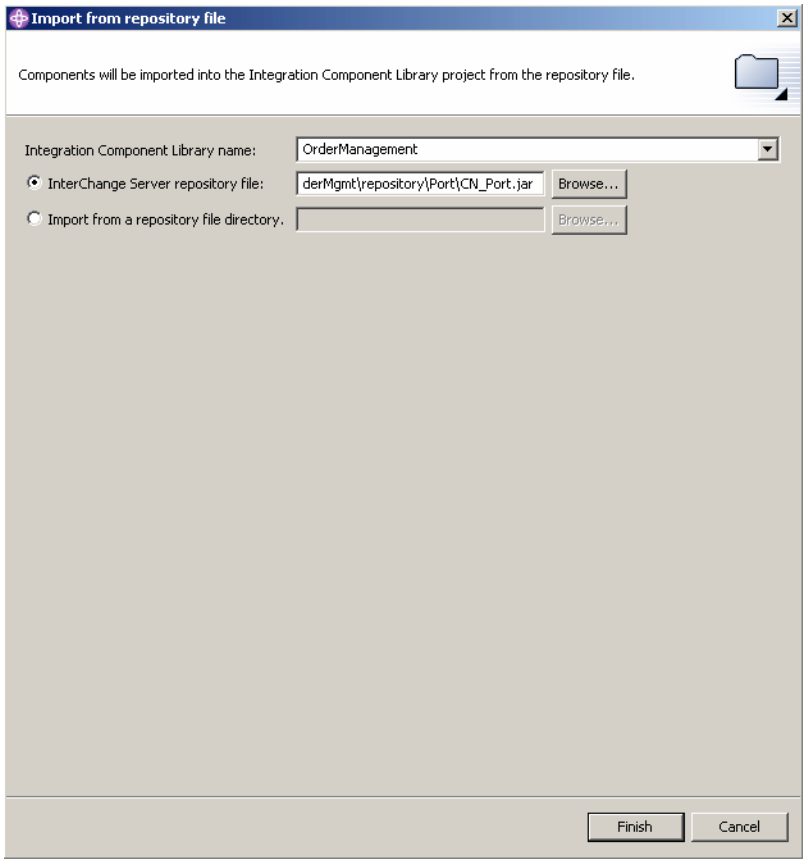
3. The normal Connector Configuration window opens, from which you can save the connector configuration to your ICL.
4. Repeat steps 1 on page 310 through 3 for each of the adapters you will use during the project. Repeat the steps to also create the configuration for the JDBC adapter.

We configure both the WebSphere MQ Workflow adapter and the JDBC adapter later in this chapter.

Later in the implementation when we create and configure our Sales Order Management Collaboration object, it will be a requirement to *bind* each of the collaboration ports to an adapter. For our implementation we will only be using the From and To ports, thus we will bind the remaining ports to the Port connector. To import the Port connector into our ICL, right click the ICL name and select **Import from repository file**. The needed JAR file for our implementation is found in this directory:

```
C:\IBM\IndustryCollabs\OrderMgmt\repository\Port\CN_Port.jar
```

See Figure 7-53.


Import from repository file

Components will be imported into the Integration Component Library project from the repository file.

Integration Component Library name: OrderManagement

(●) InterChange Server repository file: derMgmt\repository\Port\CN_Port.jar

( ) Import from a repository file directory.

Figure 7-53 Importing Port connector

Tip: You might have to refresh the view in order to see the imported objects in the System Manager.

7.7.2 WebSphere MQ Workflow adapter configuration

In this section, we show how to configure, start, and stop the WebSphere MQ Workflow adapter to work with WebSphere InterChange Server.

Configuring an adapter consists, in general, of the following steps:

1. Import any required meta objects.
2. Use the Connector Configurator to configure standard properties.

3. Use the Connector Configurator to configure application-specific properties.
4. Create any resources that the connector relies on, such as WebSphere MQ queues, event tables, and so forth.
5. Customize start-up scripts for the connector if required.

Importing required meta objects

The WebSphere MQ Workflow adapter requires the use of meta objects to store additional configuration information. These meta objects are actually business objects with predefined attributes that are specific to the operation of the WebSphere MQ Workflow adapter. These meta objects need to be imported into the System Manager for configuration and deployment along with the adapter.

To import the WebSphereMQWorkflow_MetaObjects.jar file, which holds the meta-objects, right click the ICL name and select **Import from repository file**. The file in our implementation was located at:

```
C:\IBM\WBIAdapters\repository\WebSphereMQWorkflow\WebSphereMQWorkflow_
MetaObjects.jar
```

For those adapters that use datahandlers it is also necessary to import the datahandler meta objects into the System Manager. The WebSphere MQ Workflow adapter requires the use of the XML datahandler, so we must import those meta objects before completing the configuration of the adapter. The datahandler meta objects are not stored in a JAR file, instead they are in TXT files. To import these files, follow these steps:

1. Expand the ICL that you are using within the System Manager, right-click the **Business Objects** folder, and select **Create New business object.**
2. Cancel the window to create a new object, that window is shown in Figure 7-54.

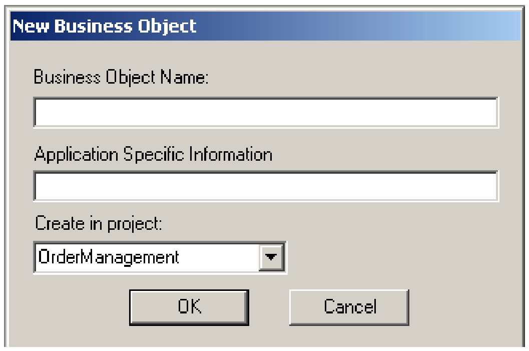
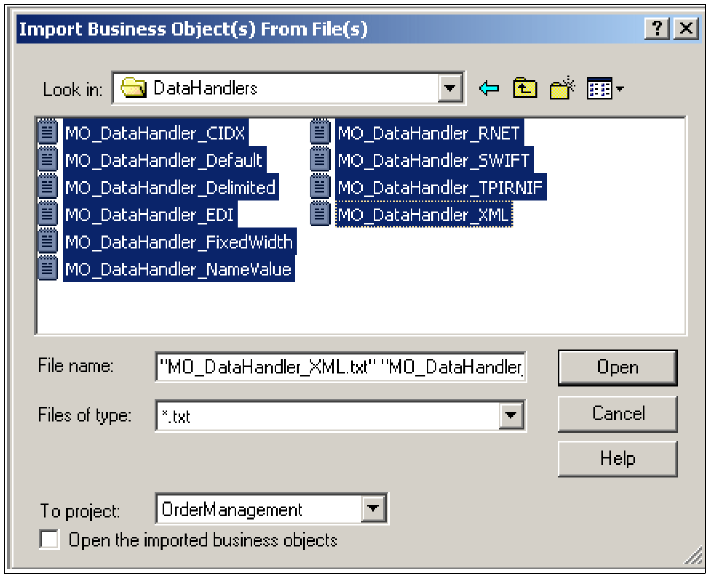


Figure 7-54 Cancel the New Business Object window

3. Select **File** → **Open From File** and select all the following TXT files (all in the list) from the directory:

   `C:\IBM\WebSphereICS\DataHandlers\repository\DataHandlersdirectory`

   To see the files, change the file type to *.txt, as shown in Figure 7-55. Make sure that the Open the imported business objects option is *not* selected.



MO_DataHandler_CIDX
MO_DataHandler_Default
MO_DataHandler_Delimited
MO_DataHandler_EDI
MO_DataHandler_FixedWidth
MO_DataHandler_NameValue
MO_DataHandler_RNET
MO_DataHandler_SWIFT
MO_DataHandler_TPIRNIF
MO_DataHandler_XML

File name: "MO_DataHandler_XML.txt" "MO_DataHandler

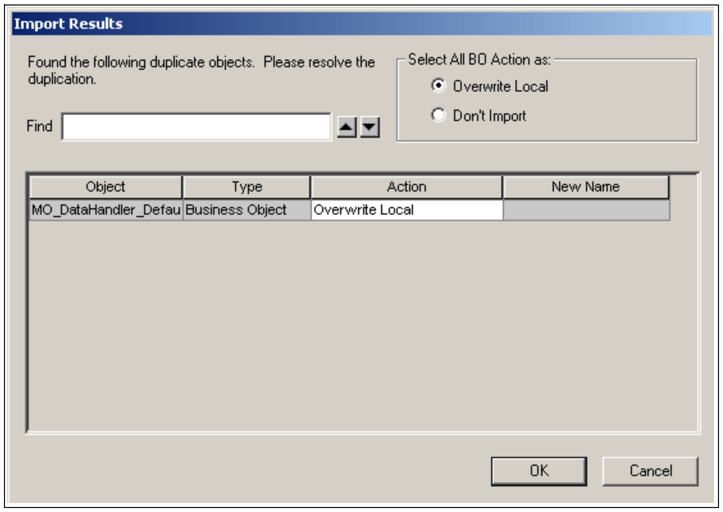
Files of type: *.txt

To project: OrderManagement

☐ Open the imported business objects

Figure 7-55 Select Datahandler MO files for import

4. During the import process, multiple Import Results windows will be displayed. For each of these windows, select **Overwrite Local** and click **OK**. One of these windows is shown in Figure 7-56.


Import Results

Found the following duplicate objects. Please resolve the duplication.

Find

Select All BO Action as: [✓ Overwrite Local] Don't Import

Object	Type	Action	New Name
MO_DataHandler_Defau	Business Object	Overwrite Local	

Figure 7-56 Overwrite Local objects

Configuring the connector object

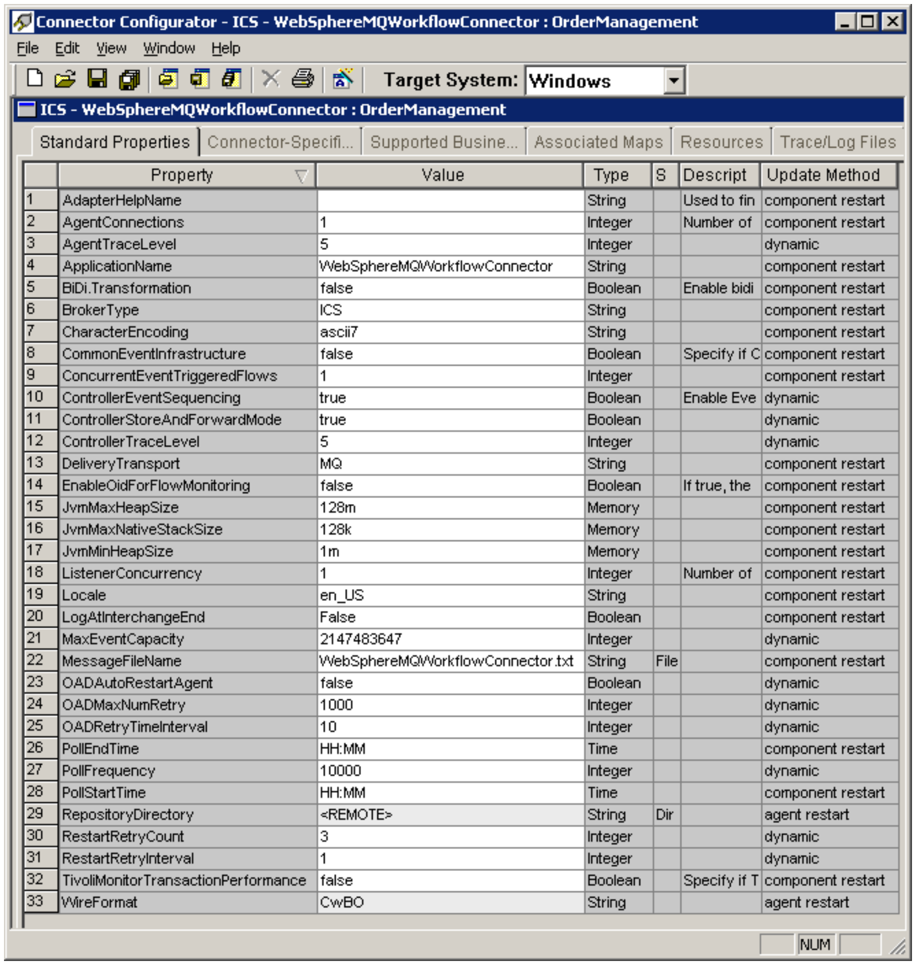
To start the configuration process, double-click the imported WebSphere MQ Workflow connector located in your ICL. This will start the Connector Configurator. First, select the **Standard Properties** tab.

Many parameters enable you to tune the behavior of the connector, but the most important ones are listed here:

- BrokerType: `ICS`
- DeliveryType: `MQ`
- ApplicationName: `WebSphereMQWorkflowConnector`

Note: Note that selecting **MQ** as the delivery type requires the completion of the **Messaging** tab.

The ApplicationName attribute is the real name of the connector. This will also be the name that you provide as a parameter in the start-up of the connector agent. When the connector agent connects to the InterChange Server, a connector controller must be available with the same name (similar to the way that the names of a sender channel and a receiver channel have to match when setting up communication between two queue managers).


Connector Configurator - ICS - WebSphereMQWorkflowConnector : OrderManagement
File Edit View Window Help
Target System: Windows
ICS - WebSphereMQWorkflowConnector : OrderManagement
Standard Properties
Connector-Specifi...
Supported Busine...
Associated Maps
Resources
Trace/Log Files
NUM

	Property	Value	Type	S	Descript	Update Method
1	AdapterHelpName		String		Used to fin	component restart
2	AgentConnections	1	Integer		Number of	component restart
3	AgentTraceLevel	5	Integer			dynamic
4	ApplicationName	WebSphereMQWorkflowConnector	String			component restart
5	BiDi.Transformation	false	Boolean		Enable bidi	component restart
6	BrokerType	ICS	String			component restart
7	CharacterEncoding	ascii7	String			component restart
8	CommonEventInfrastructure	false	Boolean		Specify if C	component restart
9	ConcurrentEventTriggeredFlows	1	Integer			component restart
10	ControllerEventSequencing	true	Boolean		Enable Eve	dynamic
11	ControllerStoreAndForwardMode	true	Boolean			dynamic
12	ControllerTraceLevel	5	Integer			dynamic
13	DeliveryTransport	MQ	String			component restart
14	EnableOidForFlowMonitoring	false	Boolean		If true, the	component restart
15	JvmMaxHeapSize	128m	Memory			component restart
16	JvmMaxNativeStackSize	128k	Memory			component restart
17	JvmMinHeapSize	1m	Memory			component restart
18	ListenerConcurrency	1	Integer		Number of	component restart
19	Locale	en_US	String			component restart
20	LogAtInterchangeEnd	False	Boolean			component restart
21	MaxEventCapacity	2147483647	Integer			dynamic
22	MessageFileName	WebSphereMQWorkflowConnector.txt	String	File		component restart
23	OADAutoRestartAgent	false	Boolean			dynamic
24	OADMaxNumRetry	1000	Integer			dynamic
25	OADRetryTimeInterval	10	Integer			dynamic
26	PollEndTime	HH:MM	Time			component restart
27	PollFrequency	10000	Integer			dynamic
28	PollStartTime	HH:MM	Time			component restart
29	RepositoryDirectory	<REMOTE>	String	Dir		agent restart
30	RestartRetryCount	3	Integer			dynamic
31	RestartRetryInterval	1	Integer			dynamic
32	TivoliMonitorTransactionPerformance	false	Boolean		Specify if T	component restart
33	WireFormat	CwBO	String			agent restart

Figure 7-57 Standard properties of the WebSphere MQ Workflow connector

Select the **Connector-Specific Properties** tab. One set of properties is related to the characteristics of the workflow server to which we want to connect. Provide or verify:

- ApplicationUserID and ApplicationPassword, which will be used by the connector to log on to the workflow server
- WorkflowSystemName and WorkflowSystemGroup, which identifies the workflow server in a workflow domain

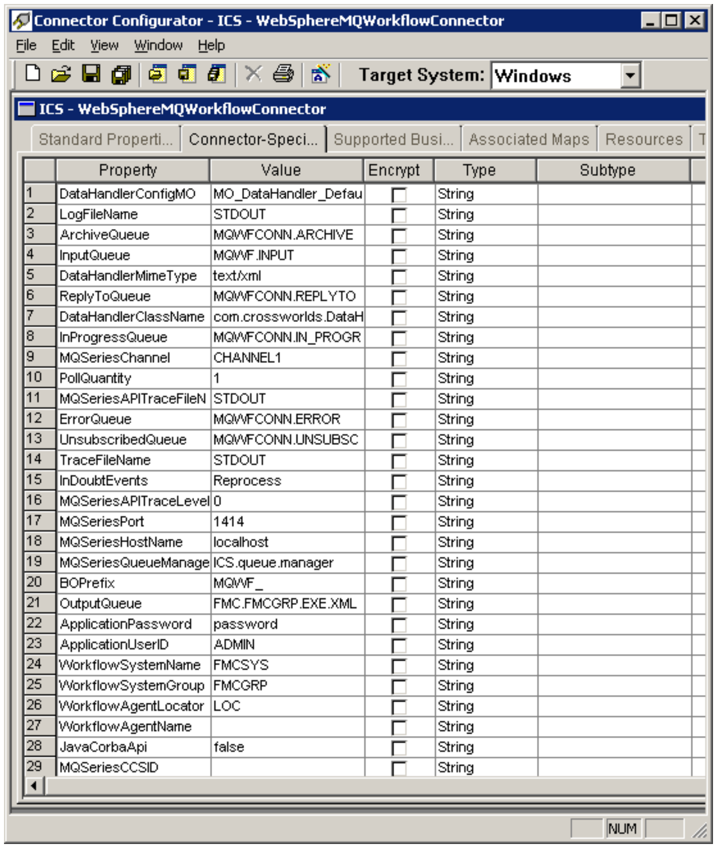
Another group of properties enables you to name the queues and the queue manager that are used by the adapter to receive and manage work that is being sent to and from the workflow server. There are queues to archive events, to report errors, to store work-in-progress, and so forth.

There are several options that you can implement when referring to the queues and queue manager:

- The adapter uses an MQ client connection to the workflow queue manager. This means that the queues that are listed on this tab of the configuration are to be added to the workflow queue manager WFQM.
- The adapter uses its own queue manager. This means that you must define this queue manager and create channels between this adapter queue manager and the workflow queue manager.
- The adapter uses the InterChange Server queue manager. This means that you must create channels between the InterChange Server queue manager and the workflow queue manager. However, it also means that the adapter is using only one queue manager to communicate with the workflow server and the InterChange Server. It also means that we must define a remote queue on the InterChange Server queue manager to point to the workflow input queue (FMC.FMCGRP.EXE.XML). This option is shown in Figure 7-58 on page 318. and is the approach we used in our scenario.

Attention: Please compare the IBM Software License Agreement and your implementation of your WebSphere MQ licenses to verify you are in compliance with using the WebSphere MQ Queue Manager to contain the configured queues for the WebSphere MQ Workflow adapter.

A last group of properties controls the parsing and generation of messages for the adapter. Workflow output messages are in XML. Therefore, we have values for DataHandlerMimeType and DataHandlerClassName that relate to the processing of text messages with an XML structure. The use of the property DataHandlerConfigMO is explained later.

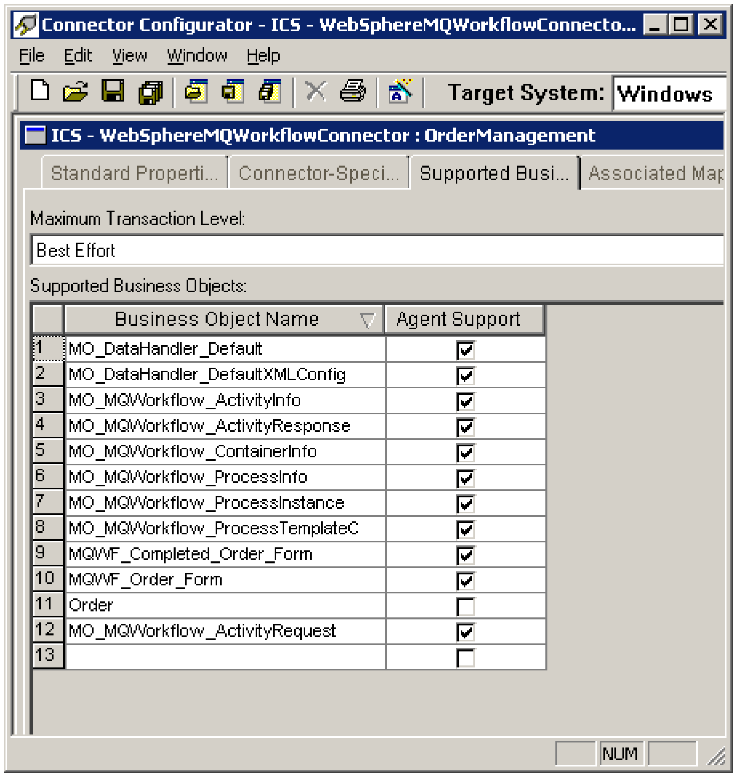

Connector Configurator - ICS - WebSphereMQWorkflowConnector
File Edit View Window Help
Target System: Windows
ICS - WebSphereMQWorkflowConnector
Standard Properti... Connector-Speci... Supported Busi... Associated Maps Resources T
NUM

	Property	Value	Encrypt	Type	Subtype
1	DataHandlerConfigMO	MO_DataHandler_Defau	☐	String	
2	LogFileName	STDOUT	☐	String	
3	ArchiveQueue	MQWFCONN.ARCHIVE	☐	String	
4	InputQueue	MQWF.INPUT	☐	String	
5	DataHandlerMimeType	text/xml	☐	String	
6	ReplyToQueue	MQWFCONN.REPLYTO	☐	String	
7	DataHandlerClassName	com.crossworlds.DataH	☐	String	
8	InProgressQueue	MQWFCONN.IN_PROGR	☐	String	
9	MQSeriesChannel	CHANNEL1	☐	String	
10	PollQuantity	1	☐	String	
11	MQSeriesAPITraceFileN	STDOUT	☐	String	
12	ErrorQueue	MQWFCONN.ERROR	☐	String	
13	UnsubscribedQueue	MQWFCONN.UNSUBSC	☐	String	
14	TraceFileName	STDOUT	☐	String	
15	InDoubtEvents	Reprocess	☐	String	
16	MQSeriesAPITraceLevel	0	☐	String	
17	MQSeriesPort	1414	☐	String	
18	MQSeriesHostName	localhost	☐	String	
19	MQSeriesQueueManage	ICS.queue.manager	☐	String	
20	BOPrefix	MQWF_	☐	String	
21	OutputQueue	FMC.FMCGRP.EXE.XML	☐	String	
22	ApplicationPassword	password	☐	String	
23	ApplicationUserID	ADMIN	☐	String	
24	WorkflowSystemName	FMCSYS	☐	String	
25	WorkflowSystemGroup	FMCGRP	☐	String	
26	WorkflowAgentLocator	LOC	☐	String	
27	WorkflowAgentName		☐	String	
28	JavaCorbaApi	false	☐	String	
29	MQSeriesCCSID		☐	String	

Figure 7-58 Connector specific properties

1. Click the **Supported Business Objects** tab and add **MQWF_Order_Form**, **MQWF_Completed_Order_Form** and **Order** business objects (Figure 7-59 on page 319). Add agent support for the business object **MQWF_Order_Form** and **MQWF_Completed_Order_Form**.

   Agent support is always needed for the application-specific business objects and for the meta-objects. Verify that the WebSphere MQ Workflow adapter contains all the supports all the business objects displayed in Figure 7-62 on page 327, if not, add them now as shown.


Connector Configurator - ICS - WebSphereMQWorkflowConnecto...
File Edit View Window Help
Target System: Windows
ICS - WebSphereMQWorkflowConnector : OrderManagement
Standard Properti... Connector-Speci... Supported Busi... Associated Map
NUM

Maximum Transaction Level:

Best Effort

Supported Business Objects:

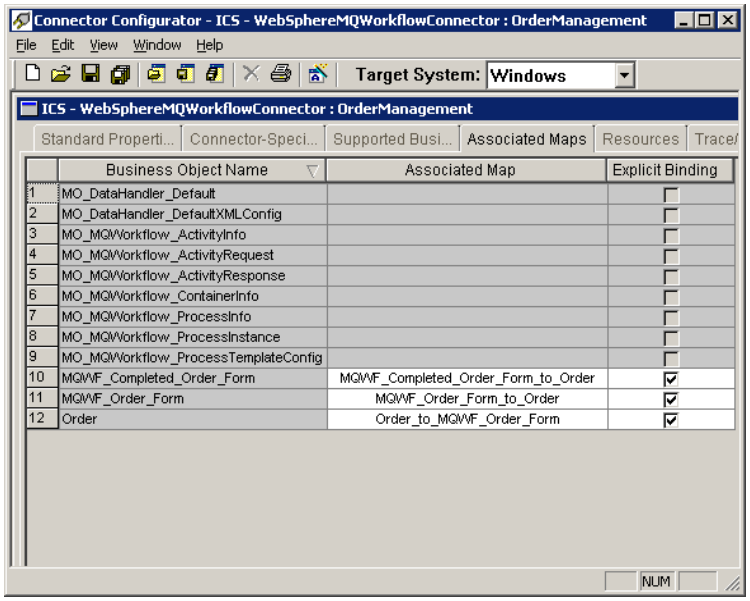
	Business Object Name	Agent Support
1	MO_DataHandler_Default	✓
2	MO_DataHandler_DefaultXMLConfig	✓
3	MO_MQWorkflow_ActivityInfo	✓
4	MO_MQWorkflow_ActivityResponse	✓
5	MO_MQWorkflow_ContainerInfo	✓
6	MO_MQWorkflow_ProcessInfo	✓
7	MO_MQWorkflow_ProcessInstance	✓
8	MO_MQWorkflow_ProcessTemplateC	✓
9	MQWF_Completed_Order_Form	✓
10	MQWF_Order_Form	✓
11	Order	☐
12	MO_MQWorkflow_ActivityRequest	✓
13		☐

Figure 7-59 Adding business objects to WebSphere MQ Workflow adapter

2. To associate maps with the business objects, click the **Associated Maps** tab and mark the **Explicit binding** check box for the MQWF_Order_Form, MQWF_Completed_Order, and Order business object. Select the corresponding maps, as shown in Figure 7-60 on page 320. These are the maps built in 7.6, "Create maps" on page 275. Note that we do not refer to the child business objects or to the submaps.

Note: If tab you cannot see the objects on the Associated Maps that you have added on the Supported Objects tab, save the configuration first to refresh the view.

If you cannot select the expected map, you might have to verify the mapping direction.


Connector Configurator - ICS - WebSphereMQWorkflowConnector : OrderManagement
File Edit View Window Help
ICS - WebSphereMQWorkflowConnector : OrderManagement
Standard Properti... Connector-Speci... Supported Busi... Associated Maps Resources Trace/
NUM

Target System: Windows

	Business Object Name	Associated Map	Explicit Binding
1	MO_DataHandler_Default		☐
2	MO_DataHandler_DefaultXMLConfig		☐
3	MO_MQWorkflow_ActivityInfo		☐
4	MO_MQWorkflow_ActivityRequest		☐
5	MO_MQWorkflow_ActivityResponse		☐
6	MO_MQWorkflow_ContainerInfo		☐
7	MO_MQWorkflow_ProcessInfo		☐
8	MO_MQWorkflow_ProcessInstance		☐
9	MO_MQWorkflow_ProcessTemplateConfig		☐
10	MQWF_Completed_Order_Form	MQWF_Completed_Order_Form_to_Order	✓
11	MQWF_Order_Form	MQWF_Order_Form_to_Order	✓
12	Order	Order_to_MQWF_Order_Form	✓

Figure 7-60 Associating maps to workflow connector

3. When all of the properties are configured, the connector should be saved to the project. To save the connector configuration, select **File** → **Save** → **To Project**. We save the connector configuration to a file as well by selecting select **File** → **Save** → **To File**. Provide a location and a name for the connector configuration file. For our implementation we used:

```
C:\IBM\WBIAdapters\connectors\WebSphereMQWorkflow\
WebSphereMQWorkflowConnector.cfg
```

Note: A configuration file is required if you are using JMS as the delivery transport. Without this file, the connector does not have sufficient information to connect to the InterChange Server at startup.

Runtime configuration

Two tasks must be completed at this stage:

- Several WebSphere MQ queues must be defined, including communication between ICS.queue.manager and the workflow queue manager WFQM.
- The agent component is started by a script that needs customization. This script also takes several parameters that must be reviewed and possibly changed.

WebSphere MQ configuration

In our scenario, communication between the InterChange Server queue manager and the MQ Workflow server queue manager is through an MQ channel. The MQ Workflow adapter is also using WebSphere MQ as the transport delivery mechanism.

The names of these queues are listed on the standard properties of the connector configuration (Figure 7-57 on page 316). A sample script to define these queues can be found in crossworlds_mq.tst file in the \WebSphereICS\mqseries folder. Create a copy of this file for back-up purposes and open the original in a text file editor.

The names of the queues used by the connector should adopt the following naming standard.

```
DEFINE QLOCAL(IC/SERVER_NAME/DestinationAdapter)
DEFINE QLOCAL(AP/DestinationAdapter/SERVER_NAME)
```

As a result, the queues defined in the crossworld_mq.tst has the following queue definitions in Example 7-6.

Example 7-6 Sample MQ definitions

```
DEFINE QLOCAL(IC/ICS/WebSphereMQWorkflowConnector)
DEFINE QLOCAL(AP/WebSphereMQWorkflowConnector/ICS)
```

In addition, there are other queues to be defined and listed on the Connector-Specific Properties tab (see Figure 7-58 on page 318). These queues are to be hosted by the InterChange Server queue manager in our implementation.

The crossworld_mq.tst contains the following MQ-defined commands for the following queues:

- Local queues:
  - MQWFCONN.ARCHIVE
  - MQWF.INPUT
  - MQWFCONN.REPLYTO
  - MQWFCONN.IN_PROGRESS
  - MQWFCONN.ERROR
  - MQWFCONN.UNSUBSCRIBED

  The queue commands are listed in Example 7-7 on page 322.

Example 7-7 MQ Workflow Adapter required queues

```
DEFINE QLOCAL(MQWFCONN.ARCHIVE)
DEFINE QLOCAL(MQWF.INPUT)
DEFINE QLOCAL(MQWFCONN.REPLYTO)
DEFINE QLOCAL(MQWFCONN.IN_PROGRESS)
DEFINE QLOCAL(MQWFCONN.ERROR)
DEFINE QLOCAL(MQWFCONN.UNSUBSCRIBED)
```

- Remote queue:
  - FMC.FMCGRP.EXE.XML

    This remote queue object is basically an alias for an object that resides on the queue manager WFQM. To define this object, you can use a command similar to Example 7-8:

Example 7-8 Remote queue

```
DEFINE QREMOTE(FMC.FMCGRP.EXE.XML) RNAME(FMC.FMCGRP.EXE.XML) RQMNAME(WFQM)
XMITQ(WBIWF)
```

- Transmission queue:

  The parameter XMITQ in the previous command refers to a transmission queue that must be added as well (Example 7-9):

Example 7-9 Transmission queue:

```
DEFINE QLOCAL(WBIWF) USAGE(XMITQ) TRIGGER TRIGTYPE(FIRST)
INITQ(SYSTEM.CHANNEL.INITQ) TRIGDATA(WBIICS.TO.WBIWF)
```

- Channels

  MQ channels are the objects that represent the way that communication will be set up at runtime. One queue manager should be a sender, and the other a receiver channel, both with matching names as in Example 7-10 and Example 7-11 on page 323.

Example 7-10 ICS.queue.manager channel definition

```
DEFINE CHL(WBIICS.TO.WBIWF) CHLTYPE(SDR) CONNAME(‘wbiwf(5010)’) XMITQ(WBIWF)
TRPTYPE(TCP)
DEFINE CHL(WBIWF.TO.WBIICS) CHLTYPE(RCVR)
```

Example 7-11 WFQM channel definition

```
DEFINE CHL(WBIICS.TO.WBIWF) CHLTYPE(RCVR)

DEFINE CHL(WBIWF.TO.WBIICS) CHLTYPE(SDR) CONNAME('wbiics(1414)') XMITQ(WBIICS)
TRPTYPE(TCP)

DEFINE QLOCAL(WBIICS) USAGE(XMITQ) TRIGGER TRIGTYPE(FIRST)
INITQ(SYSTEM.CHANNEL.INITQ) TRIGDATA(WBIWF.TO.WBIICS)
```

These objects together create full communication between the two queue managers. The objects are created in such a way that communication starts automatically whenever a message is created for the other queue manager.

For our implementation we added the definitions from Example 7-6 on page 321, Example 7-7 on page 322, Example 7-8 on page 322, Example 7-9 on page 322, and Example 7-10 on page 322 to our open crossworlds_mq.tst file and then saved and closed the file. To create the definitions within WebSphere MQ, select **Start** → **Programs** → **IBM WebSphere InterChange Server** → **IBM WebSphere MQ** → **Configure Queue Manager**.

An alternative to this classical MQ setup is to exploit the fact that the workflow queue manager WFQM is part of an MQ cluster. The InterChange Server queue manager could join this cluster.

To join the existing cluster FMCGRP, the following objects are required:

```
DEFINE CHL(TO.WFQM.TCP) CHLTYPE(CLUSSDR) CONNAME('wbiwf(5010)') TRPTYPE(TCP)
CLUSTER(FMCGRP)

DEFINE CHL(TO.WBIICS.TCP) CHLTYPE(CLUSRCVR) CONNAME(wbiics) TRPTYPE(TCP)
CLUSTER(FMCGRP)
```

Finally, make sure that the input queue of the connector is shared in the cluster:

```
ALTER QLOCAL(MQWF.INPUT) CLUSTER(FMCGR)
```

No objects have to be defined or altered on the workflow queue manager. Note that the cluster name FMCGRP was named during the creation of the workflow server. The names of the channels are fixed this time, because the workflow configuration object has created matching objects during initial setup of the workflow server.

Note: To allow for TCP/IP communication, an MQ listener task for TCP/IP must be created and started. For the workflow queue manager, this is created during initial setup. For the InterChange queue manager, this object does not get created by default. You can use IBM MQSeries services application to create and start such a listener task. Make sure that you create this object for the correct TCP/IP port, which is set to 1414 in the previous commands.

Start-up script and parameters

During the installation of the WebSphere Business Integration Adapters for WebSphere MQ Workflow, a shortcut has been created in the **Start** → **Programs** → **IBM WebSphere Business Integration Adapters** → **Adapters** → **Connectors** folder to start the connector agent. However, this shortcut needs modification to point to the configuration file we generated earlier with the Connector Configuration tool.

1. To modify this shortcut, open the properties of the shortcut used to start the connector agent and inspect the Target field. This command line usually does not reference the configuration file. Add it with the `-c` parameter, as shown in Example 7-12.

Example 7-12 Parameters for the start-up of the WebSphere MQ Workflow connector

```
C:\IBM\WBIAdapters\connectors\WebSphereMQWorkflow\start_WebSphereMQWorkflow.bat
WebSphereMQWorkflow ICS
-cC:\IBM\WBIAdapters\connectors\WebSphereMQWorkflow\WebSphereMQWorkflowConnecto
r.cfg
```

Note: Example 7-12 displays the Target information, a single line entry into the shortcut properties

The start-up script, to which this shortcut points, is in the C:\IBM\WBIAdapters\connectors\WebSphereMQWorkflow folder and is called start_WebSphereMQWorkflow.bat. Open this script file in a text editor to make any changes that are required.

The script sets an environment variable MQWF_JAVA_LIB to point to the Java support files of WebSphere MQ Workflow. The default value is:

```
set MQWF_JAVA_LIB="C:\Program Files\IBM WebSphere MQ Workflow\BIN\Java3320"
```

2. In our distributed environment, this folder does not exist on the development machine. The corresponding folder on the WebSphere MQ Workflow runtime server is called \WMQWF\BIN\Java3500. The different version number in the folder name is a consequence of using Version 3.5 for WebSphere MQ Workflow.

Copy this folder from the runtime server to the InterChange Server in an appropriate directory and adjust the value of MQWF_JAVA_LIB accordingly. This folder should contain two jar files that are named in the script:

```
set CONN_LIB=%MQWF_JAVA_LIB%\fmcojagt.jar;%MQWF_JAVA_LIB%\fmcojapi.jar
```

3. Save the changes to the script.

This completes the configuration of the WebSphere MQ Workflow connector. Before we can start it, we deploy it to the server, which is covered in 7.11, “Deploy user project” on page 343.

7.7.3 JDBC adapter configuration

The JDBC adapter enables the exchange of business objects with any database that provides a JDBC driver that implements the JDBC 2.0 specification. The connector connects to the database using the JDBC connect mechanism and establishes a connection pool with the database. This connection pool is used in any interaction with the database.

1. To start the configuration process, double-click the imported JDBC connector to start the Connector Configurator. Select first the tab Standard Properties.

   Similar to what we described for the WebSphere MQ Workflow connector, the most important properties are:

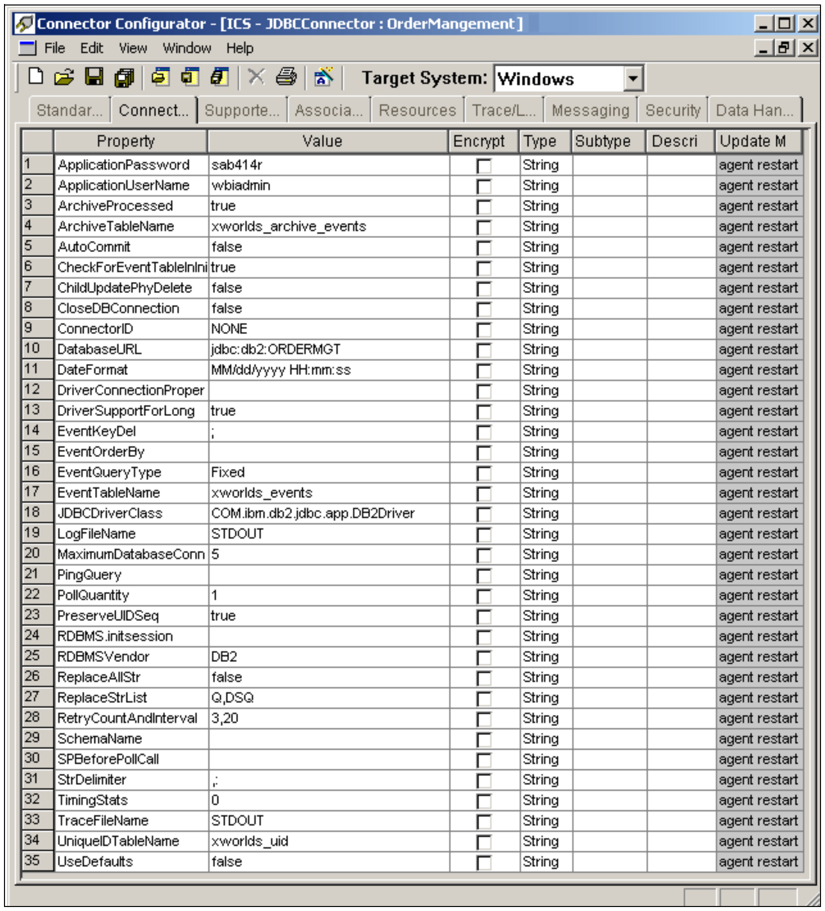
   – BrokerType: `ICS`
   – DeliveryType: `MQ`
   – jms.MessageBrokerName: `ICS.queue.manager`

2. On the **Connector-Specific properties** tab, we configure the values that control database access. Table 7-12 shows these properties.

Table 7-12 JDBC connector properties

Property	**Value**
ApplicationPassword	sab414r
ApplicationUserName	wbiadmin
DatabaseURL	jdbc:db2:ORDERMGT
JDBCDriverClass	COM.ibm.db2.jdbc.app.DB2Driver
RDBMSVendor	DB2

For our scenario we used the values listed in Table 7-12, as shown in Figure 7-61.


Connector Configurator - [ICS - JDBCConnector : OrderMangement]

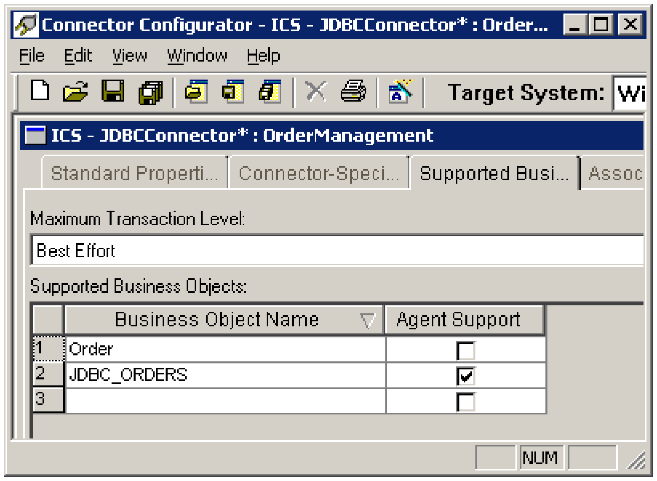
Target System: Windows

	Property	Value	Encrypt	Type	Subtype	Descri	Update M
1	ApplicationPassword	sab414r	☐	String			agent restart
2	ApplicationUserName	wbiadmin	☐	String			agent restart
3	ArchiveProcessed	true	☐	String			agent restart
4	ArchiveTableName	xworlds_archive_events	☐	String			agent restart
5	AutoCommit	false	☐	String			agent restart
6	CheckForEventTableInIni	true	☐	String			agent restart
7	ChildUpdatePhyDelete	false	☐	String			agent restart
8	CloseDBConnection	false	☐	String			agent restart
9	ConnectorID	NONE	☐	String			agent restart
10	DatabaseURL	jdbc:db2:ORDERMGT	☐	String			agent restart
11	DateFormat	MM/dd/yyyy HH:mm:ss	☐	String			agent restart
12	DriverConnectionProper		☐	String			agent restart
13	DriverSupportForLong	true	☐	String			agent restart
14	EventKeyDel	;	☐	String			agent restart
15	EventOrderBy		☐	String			agent restart
16	EventQueryType	Fixed	☐	String			agent restart
17	EventTableName	xworlds_events	☐	String			agent restart
18	JDBCDriverClass	COM.ibm.db2.jdbc.app.DB2Driver	☐	String			agent restart
19	LogFileName	STDOUT	☐	String			agent restart
20	MaximumDatabaseConn	5	☐	String			agent restart
21	PingQuery		☐	String			agent restart
22	PollQuantity	1	☐	String			agent restart
23	PreserveUIDSeq	true	☐	String			agent restart
24	RDBMS.initsession		☐	String			agent restart
25	RDBMSVendor	DB2	☐	String			agent restart
26	ReplaceAllStr	false	☐	String			agent restart
27	ReplaceStrList	Q,DSQ	☐	String			agent restart
28	RetryCountAndInterval	3,20	☐	String			agent restart
29	SchemaName		☐	String			agent restart
30	SPBeforePollCall		☐	String			agent restart
31	StrDelimiter	,;	☐	String			agent restart
32	TimingStats	0	☐	String			agent restart
33	TraceFileName	STDOUT	☐	String			agent restart
34	UniqueIDTableName	xworlds_uid	☐	String			agent restart
35	UseDefaults	false	☐	String			agent restart

Figure 7-61 Configuring JDBC connector

3. Click the **Supported Business Objects** tab and add the **JDBC_ORDERS** and **Order** business objects (Figure 7-62 on page 327). Add **Agent Support** for JDBC_ORDERS.

   Save the JDBC adapter configuration to the project.


Connector Configurator - ICS - JDBCConnector* : Order...
File Edit View Window Help
Target System: Wi
ICS - JDBCConnector* : OrderManagement
Standard Properti... Connector-Speci... Supported Busi... Assoc
NUM

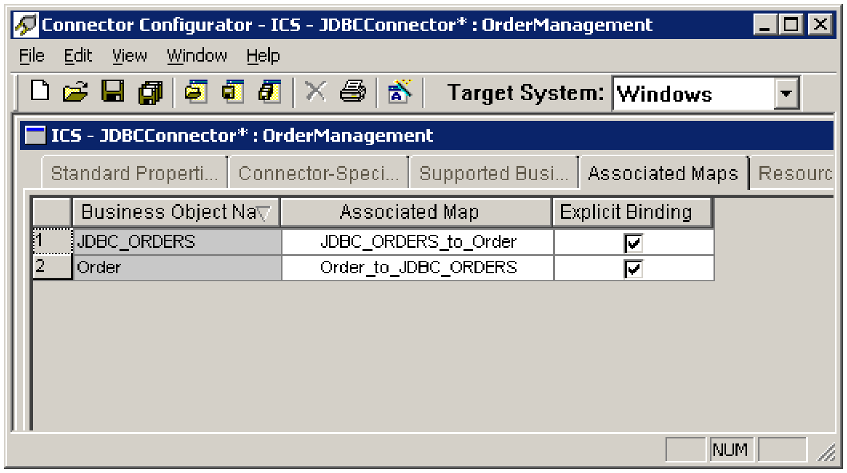
Maximum Transaction Level:
Best Effort

Supported Business Objects:

	Business Object Name	Agent Support
1	Order	☐
2	JDBC_ORDERS	✓
3		☐

Figure 7-62 Adding business objects to JDBC connector

4. To associate maps with the business objects, select the **Associated Maps** tab and mark the **Explicit binding** check box for the business objects **JDBC_ORDERS** and **Order**. Select the corresponding maps, as shown in Figure 7-63.


Connector Configurator - ICS - JDBCConnector* : OrderManagement
File Edit View Window Help
Target System: Windows
ICS - JDBCConnector* : OrderManagement
Standard Properti... Connector-Speci... Supported Busi... Associated Maps Resourc
NUM

	Business Object Na	Associated Map	Explicit Binding
1	JDBC_ORDERS	JDBC_ORDERS_to_Order	✓
2	Order	Order_to_JDBC_ORDERS	✓

Figure 7-63 Associating maps to JDBC connector

5. Save the configuration to the project and also to file.

Runtime configuration

In this section, we complete three tasks:

- Three tables, which the JDBC connector is using for event processing, must be defined.
- Several WebSphere MQ queues must be defined. These queues are used for communication between the connector agent and the InterChange Server.
- The agent component is started by a script that requires customization. This script also takes several parameters that must be reviewed and possibly changed.

Event table definition

The JDBC connector uses three tables in the application database for event management. In our scenario, the database is not an initiator of events so we are not required to define event and archive tables for the JDBC connector. However, for future scenarios in which events might have to be processed by the JDBC connector and the InterChange Server, it is a good idea to define these tables anyway.

Example 7-13 shows customized DDL for the archive events table. Note that it has the same schema name as the user ID that is used by the JDBC connector to interact with the application database.

Example 7-13 SQL DDL for storing archived events

```
CREATE TABLE WBIADMIN.XWORLDS_ARCHIVE_EVENTS
      ( EVENT_ID INTEGER  NOT NULL ,
      CONNECTOR_ID VARCHAR (40) ,
      OBJECT_KEY VARCHAR (80)  NOT NULL ,
      OBJECT_NAME VARCHAR (40)  NOT NULL ,
      OBJECT_VERB VARCHAR (40)  NOT NULL ,
      EVENT_PRIORITY INTEGER  NOT NULL ,
      EVENT_TIME TIMESTAMP ,
      ARCHIVE_TIME TIMESTAMP ,
      EVENT_STATUS INTEGER  NOT NULL ,
      EVENT_COMMENT VARCHAR (100)  ,
      CONSTRAINT CC1086984972799 PRIMARY KEY ( EVENT_ID)  ) ;
```

Example 7-14 shows SQL DDL for the events table. It is defined under the schema name of WBIADMIN.

Example 7-14 SQL DDL for storing events

```
CREATE TABLE WBIADMIN.XWORLDS_EVENTS
      ( EVENT_ID INTEGER  NOT NULL ,
      CONNECTOR_ID VARCHAR (40) ,
      OBJECT_KEY VARCHAR (80)  NOT NULL ,
      OBJECT_NAME VARCHAR (40)  NOT NULL ,
      OBJECT_VERB VARCHAR (40)  NOT NULL ,
      EVENT_PRIORITY INTEGER  NOT NULL ,
      EVENT_TIME TIMESTAMP ,
      EVENT_STATUS INTEGER  NOT NULL ,
      EVENT_COMMENT VARCHAR (100)  ,
      CONSTRAINT CC1086984972799 PRIMARY KEY ( EVENT_ID)  ) ;
```

Finally, Example 7-15 shows SQL DDL for the identifiers table.

Example 7-15 SQL DDL for the storing identifiers

```
CREATE TABLE WBIADMIN.XWORLDS_UID
      ( ID INTEGER  NOT NULL  ,
      CONSTRAINT CC1086986241113 PRIMARY KEY ( ID)  ) ;
```

WebSphere MQ configuration

Similar to the configuration for the WebSphere MQ Workflow connector, we create several queues that are named on the Standard Properties tab of the JDBC connector.

Opening the same crossworlds_mq.tst file that was used previously, add the definitions as shown in Example 7-16.

Example 7-16 Sample MQ definitions

```
DEFINE QLOCAL(IC/ICS/JDBCConnector)
DEFINE QLOCAL(AP/JDBCConnector/ICS)
```

To create the queues, select **Start** → **Programs** → **IBM WebSphere InterChange Server** → **IBM WebSphere MQ** → **Configure Queue Manager**.

Start-up script and parameters

During the installation of the WebSphere Business Integration Adapters for JDBC, a shortcut has been created in the **Start** → **Programs** → **IBM WebSphere Business Integration Adapters** → **Adapters** → **Connectors** folder to start the connector agent. However, this shortcut needs modification to point to the configuration file we generated earlier with the Connector Configuration tool.

To modify this shortcut, open the properties of the shortcut used to start the connector agent and inspect the Target field. This command line usually does not reference the configuration file. Add it with the `-c` parameter, as shown in Example 7-17.

Example 7-17 Parameters for the start-up of the JDBC connector

```
C:\IBM\WBIAdapters\connectors\JDBC\start_JDBC.bat JDBC ICS
-cC:\IBM\WBIAdapters\connectors\JDBC\JDBC.cfg
```

The start-up script, to which this shortcut points, is in the C:\IBM\WBIAdapters\connectors\JDBC folder and is called start_JDBC.bat. Open this script file in a text editor to make any changes that are required.

The script file contains a definition for an environment variable to point to the JDBC driver:

```
REM SET JDBCDRIVERPATH=
```

Change this to the location of the file db2java.zip:

```
SET JDBCDRIVERPATH=C:\IBM\SQLLIB\java\db2java.zip
```

This zip file contains the class that was named on the Connector-Specific Properties tab during the configuration of the JDBC connector.

Also in this script file, java.exe is launched with a long list of parameters. One of these parameters is -Djava.library.path, which is set to a concatenation of directories. Add the bin directory of DB2 to this concatenation, for example C:\IBM\SQLLIB\bin. Make sure that you do not add it in a way that adds a blank, which could have adverse effects.

The full and corrected command and its parameters are shown in Example 7-18.

Example 7-18 Starting the JDBC connector

```
%CWJAVA% -mx128m -ms64m -Djava.ext.dirs="%MQ_LIB%";%JRE_EXT_DIRS%
-Djava.library.path="%CROSSWORLDS%"\bin;%CONNDIR%;"%MQ_LIB%";%JRE_EXT_DIRS%;C:\
IBM\SQLLIB\BIN\ %ORB_PROPERTY% -Duser.home="%CROSSWORLDS%" -cp
%JCLASSES%;%CONNDIR%\%CONNJAR%; AppEndWrapper -l%CONNPACKAGENAME%
-n%CONNAME%Connector -s%SERVER%  %3 %4 %5
```

Port connector configuration

The configuration of the Port connector is relatively simple because it does not provide any interaction with an application.

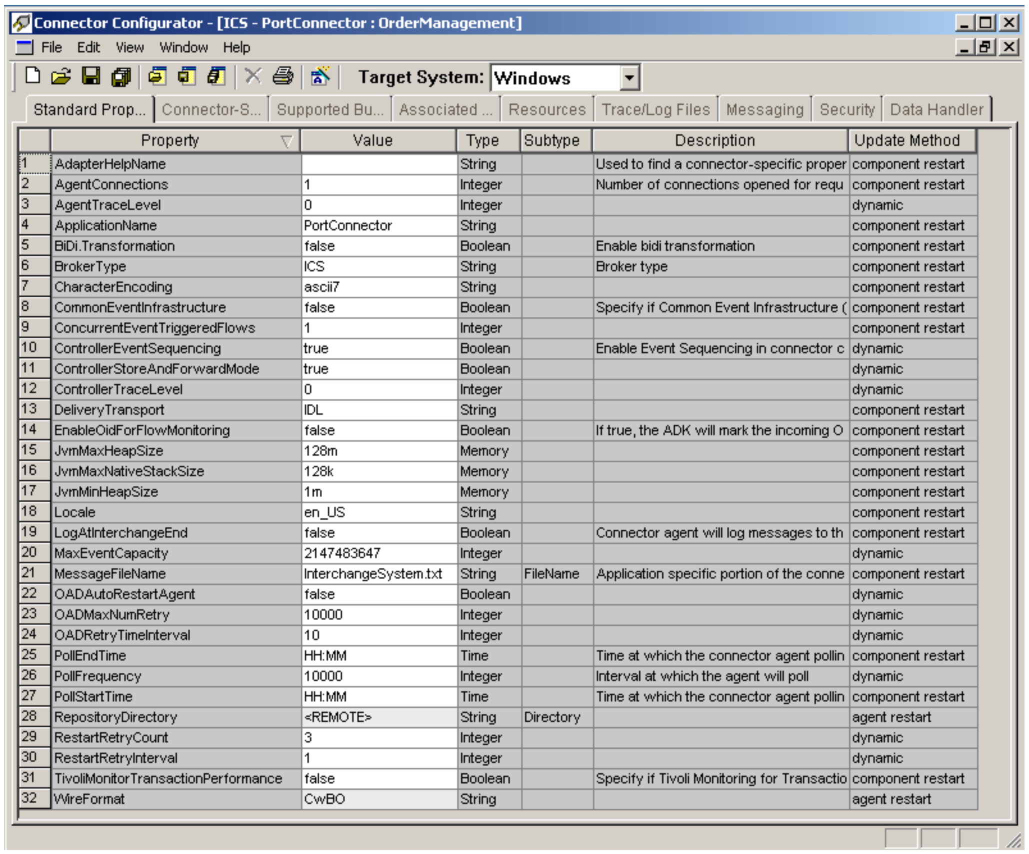
Similar to what we described for the WebSphere MQ Workflow connector, the most important properties are:

- BrokerType: `ICS`
- DeliveryType: `IDL`

> **Note:** Because the Port Connector will not actually be used for sending or receiving data, it is easier to configure the Delivery Type as IDL.

- jms.MessageBrokerName: `ICS.queue.manager`

See Figure 7-64 on page 332.

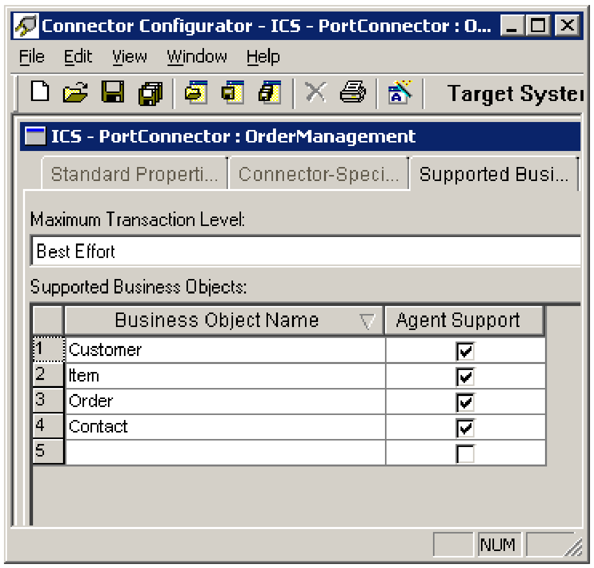

Connector Configurator - [ICS - PortConnector : OrderManagement]
File Edit View Window Help
Target System: Windows
Standard Prop... Connector-S... Supported Bu... Associated ... Resources Trace/Log Files Messaging Security Data Handler

	Property	Value	Type	Subtype	Description	Update Method
1	AdapterHelpName		String		Used to find a connector-specific proper	component restart
2	AgentConnections	1	Integer		Number of connections opened for requ	component restart
3	AgentTraceLevel	0	Integer			dynamic
4	ApplicationName	PortConnector	String			component restart
5	BiDi.Transformation	false	Boolean		Enable bidi transformation	component restart
6	BrokerType	ICS	String		Broker type	component restart
7	CharacterEncoding	ascii7	String			component restart
8	CommonEventInfrastructure	false	Boolean		Specify if Common Event Infrastructure (	component restart
9	ConcurrentEventTriggeredFlows	1	Integer			component restart
10	ControllerEventSequencing	true	Boolean		Enable Event Sequencing in connector c	dynamic
11	ControllerStoreAndForwardMode	true	Boolean			dynamic
12	ControllerTraceLevel	0	Integer			dynamic
13	DeliveryTransport	IDL	String			component restart
14	EnableOidForFlowMonitoring	false	Boolean		If true, the ADK will mark the incoming O	component restart
15	JvmMaxHeapSize	128m	Memory			component restart
16	JvmMaxNativeStackSize	128k	Memory			component restart
17	JvmMinHeapSize	1m	Memory			component restart
18	Locale	en_US	String			component restart
19	LogAtInterchangeEnd	false	Boolean		Connector agent will log messages to th	component restart
20	MaxEventCapacity	2147483647	Integer			dynamic
21	MessageFileName	InterchangeSystem.txt	String	FileName	Application specific portion of the conne	component restart
22	OADAutoRestartAgent	false	Boolean			dynamic
23	OADMaxNumRetry	10000	Integer			dynamic
24	OADRetryTimeInterval	10	Integer			dynamic
25	PollEndTime	HH:MM	Time		Time at which the connector agent pollin	component restart
26	PollFrequency	10000	Integer		Interval at which the agent will poll	dynamic
27	PollStartTime	HH:MM	Time		Time at which the connector agent pollin	component restart
28	RepositoryDirectory	<REMOTE>	String	Directory		agent restart
29	RestartRetryCount	3	Integer			dynamic
30	RestartRetryInterval	1	Integer			dynamic
31	TivoliMonitorTransactionPerformance	false	Boolean		Specify if Tivoli Monitoring for Transactio	component restart
32	WireFormat	CwBO	String			agent restart

Figure 7-64 Standard properties for the Port connector

For the Port connector, there are no connector-specific properties.

1. Select the **Supported Business Objects** tab (Figure 7-65 on page 333) and the following business objects:
   - **Order**
   - **Item**
   - **Customer**
   - **Contact**
2. Make sure to select **Agent Support** for all of them. These four business objects are the generic business objects that the collaboration sends to the ports that we are going to bind to the Port connector.


Connector Configurator - ICS - PortConnector : O...
File Edit View Window Help
Target Syster
ICS - PortConnector : OrderManagement
Standard Properti... Connector-Speci... Supported Busi...
NUM

Maximum Transaction Level: Best Effort

Supported Business Objects:

	Business Object Name	Agent Support
1	Customer	✓
2	Item	✓
3	Order	✓
4	Contact	✓
5		☐

Figure 7-65 Supported business objects for the Port connector

To see what objects are used by a certain port, look at the collaboration object or the collaboration template. Refer to Figure 7-73 on page 341 to find these objects.

3. Save the configuration to the project. You can also save it to a file, which could be used by the Test connector.

To import these meta objects into the workspace, follow these steps:

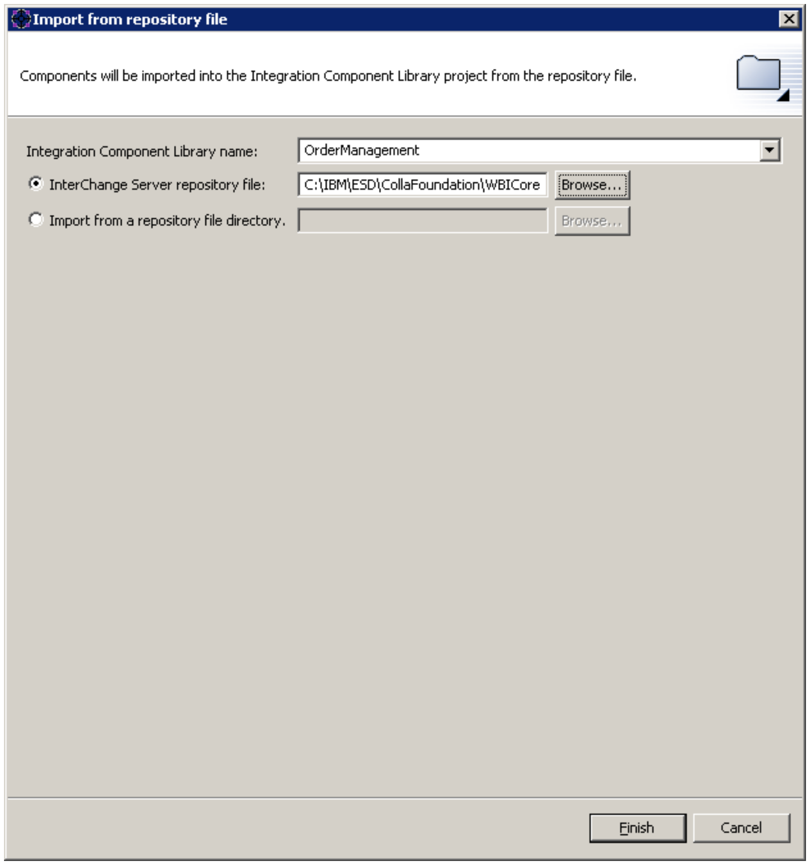
1. Open the **Business Object Designer** tool. When it is launched, select **File** → **Open** from file.
2. Point to the file in the \WebSphereICS\repository\DataHandler directory.
3. Repeat these steps until you have imported all of meta objects that you require.

7.8 Collaboration template

We used an Industry Solution Sales Order Management Collaboration in our implementation. This requires importing the collaboration into the System Manager.

1. To do this, right-click the ICL name and select **Import**.

2. In Figure 7-66, click **Browse** to locate the BIA_CT_SalesOrderProcessing.jar file in the C:\IBM\IndustryCollabs\OrderMgmt\WBICoreCollabs directory folder. Click **Finish**.


Import from repository file

Components will be imported into the Integration Component Library project from the repository file.

Integration Component Library name: OrderManagement

(●) InterChange Server repository file: C:\IBM\ESD\CollaFoundation\WBICore

( ) Import from a repository file directory.

Figure 7-66 Import the Sales Order Processing template

3. Now the SalesOrderProcessing template is loaded in the ICL, as shown in Figure 7-67.

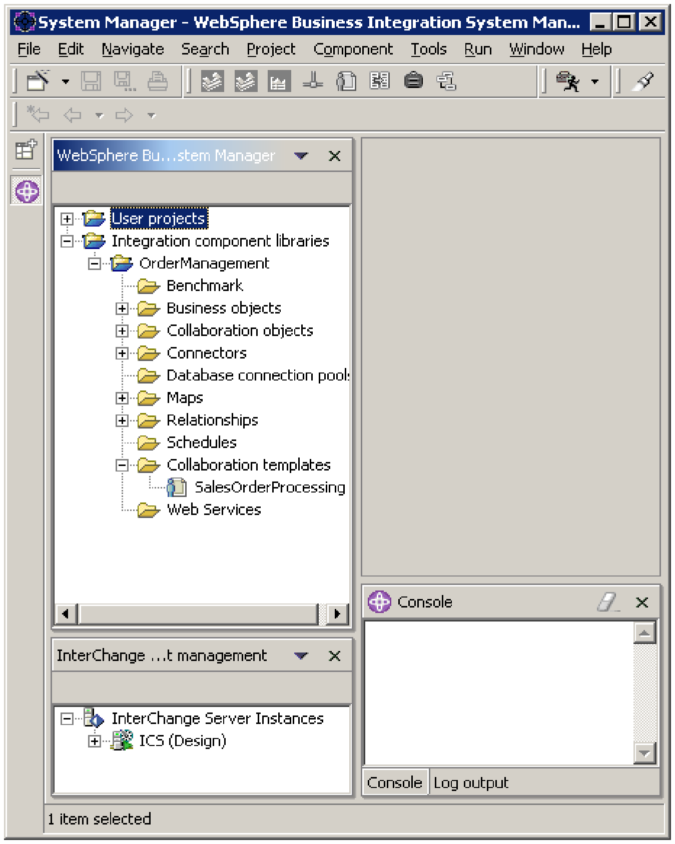
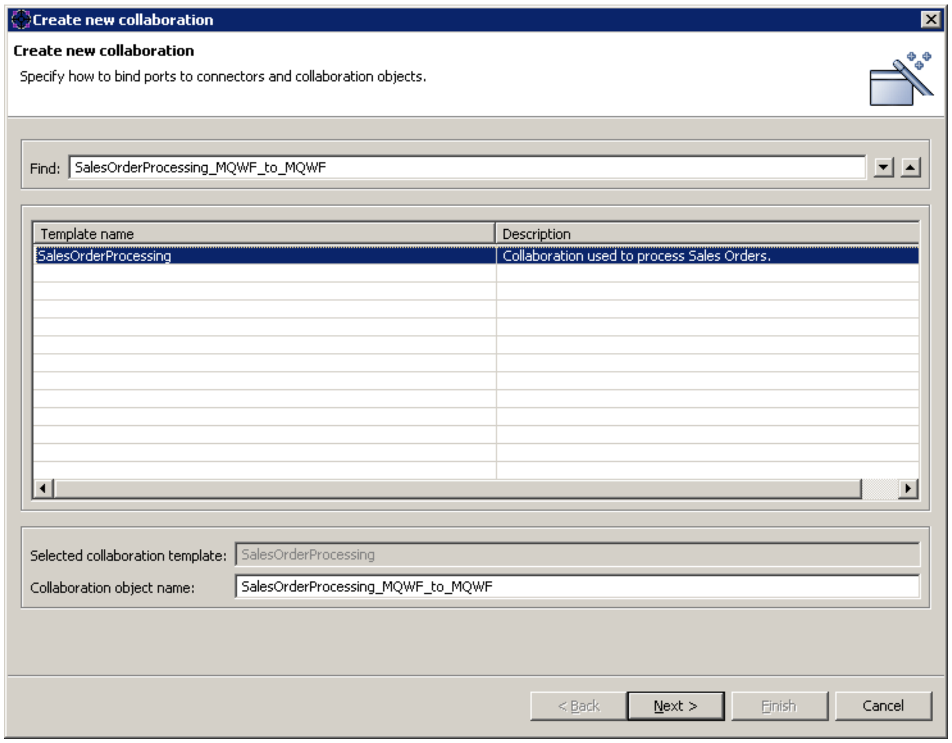

System Manager - WebSphere Business Integration System Man...
File Edit Navigate Search Project Component Tools Run Window Help
WebSphere Bu...stem Manager
User projects
Integration component libraries
OrderManagement
Benchmark
Business objects
Collaboration objects
Connectors
Database connection pool:
Maps
Relationships
Schedules
Collaboration templates
SalesOrderProcessing
Web Services
InterChange ...t management
InterChange Server Instances
ICS (Design)
Console
Console
Log output
1 item selected

Figure 7-67 Collaboration templates imported in System Manager

7.9 Collaboration object

The final step is to develop the collaboration object, which uses the SalesOrderProcessing template that we just imported into our ICL. To create the collaboration, follow these steps:

1. Right-click the **Collaboration Objects** folder inside the ICL and select **Create New Collaboration Object**.
2. Enter `SalesOrderProcessing_MQWF_to_MQWF` as the collaboration object name and select the **SalesOrderProcessing** collaboration object (Figure 7-68). Click **Next**.



Create new collaboration

Specify how to bind ports to connectors and collaboration objects.

Find: SalesOrderProcessing_MQWF_to_MQWF

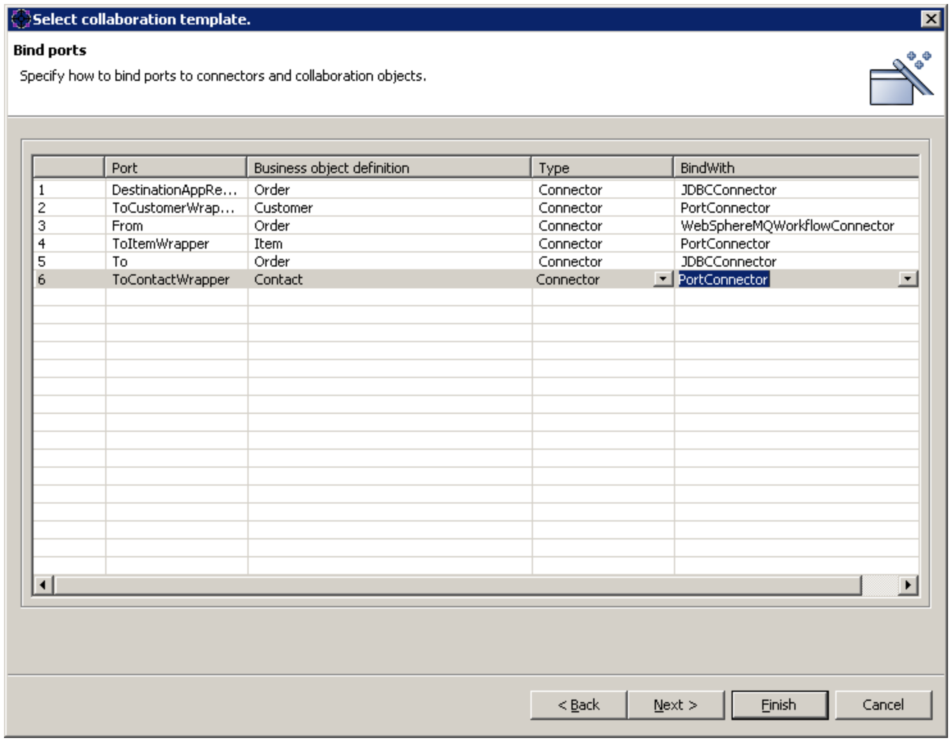
Template name	Description
SalesOrderProcessing	Collaboration used to process Sales Orders.

Selected collaboration template: SalesOrderProcessing

Collaboration object name: SalesOrderProcessing_MQWF_to_MQWF

Figure 7-68 Creating a new collaboration object: Step 1

3. Now we bind the different ports that are available in the SalesOrderProcessing collaboration to the adapters. SalesOrderProcessing uses six different ports, but we bind only two of them for our configuration. Figure 7-69 shows port bind details. After you have finished binding the port, click **Next**.



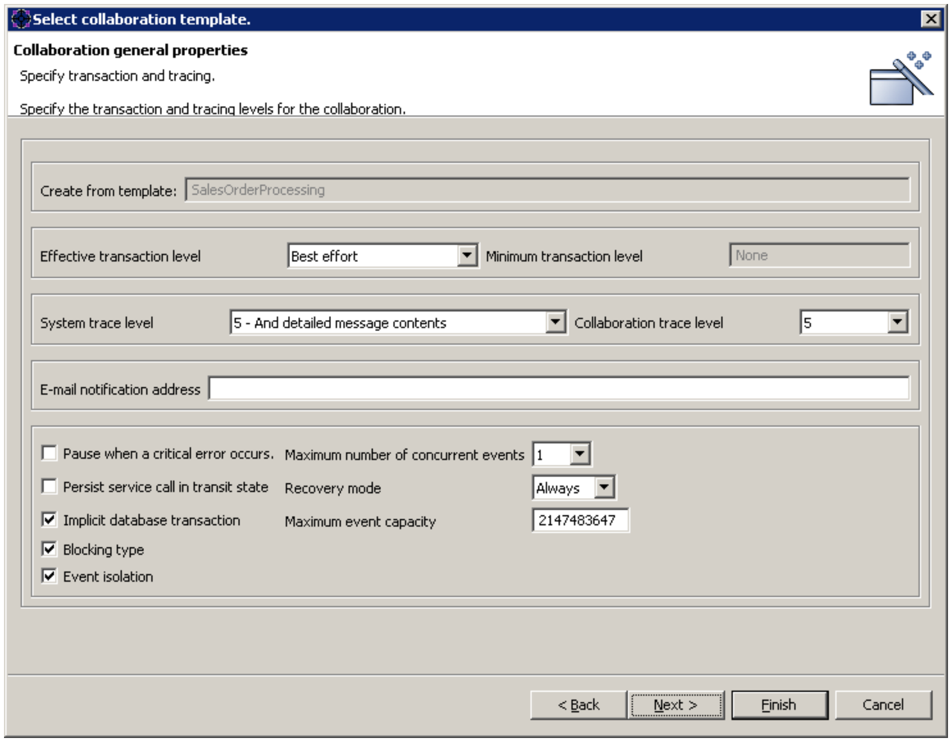
Bind ports

Specify how to bind ports to connectors and collaboration objects.

	Port	Business object definition	Type	BindWith
1	DestinationAppRe...	Order	Connector	JDBCConnector
2	ToCustomerWrap...	Customer	Connector	PortConnector
3	From	Order	Connector	WebSphereMQWorkflowConnector
4	ToItemWrapper	Item	Connector	PortConnector
5	To	Order	Connector	JDBCConnector
6	ToContactWrapper	Contact	Connector	PortConnector

Figure 7-69 Creating a new collaboration object: Step 2

4. The traces are the last configuration in our collaboration object. Set the System Trace Level and Collaboration trace level to **5**, as shown in Figure 7-70. Click **Finish**.


Select collaboration template.

Collaboration general properties

Specify transaction and tracing.

Specify the transaction and tracing levels for the collaboration.

Create from template: SalesOrderProcessing

Effective transaction level Best effort

Minimum transaction level None

System trace level 5 - And detailed message contents

Collaboration trace level 5

E-mail notification address:

- [ ] Pause when a critical error occurs.
- [ ] Persist service call in transit state
- [x] Implicit database transaction
- [x] Blocking type
- [x] Event isolation

Maximum number of concurrent events: 1

Recovery mode: Always

Maximum event capacity: 2147483647

Figure 7-70 Creating a new collaboration object: Step 3

Figure 7-71 shows the collaboration object displayed in the System Manager.

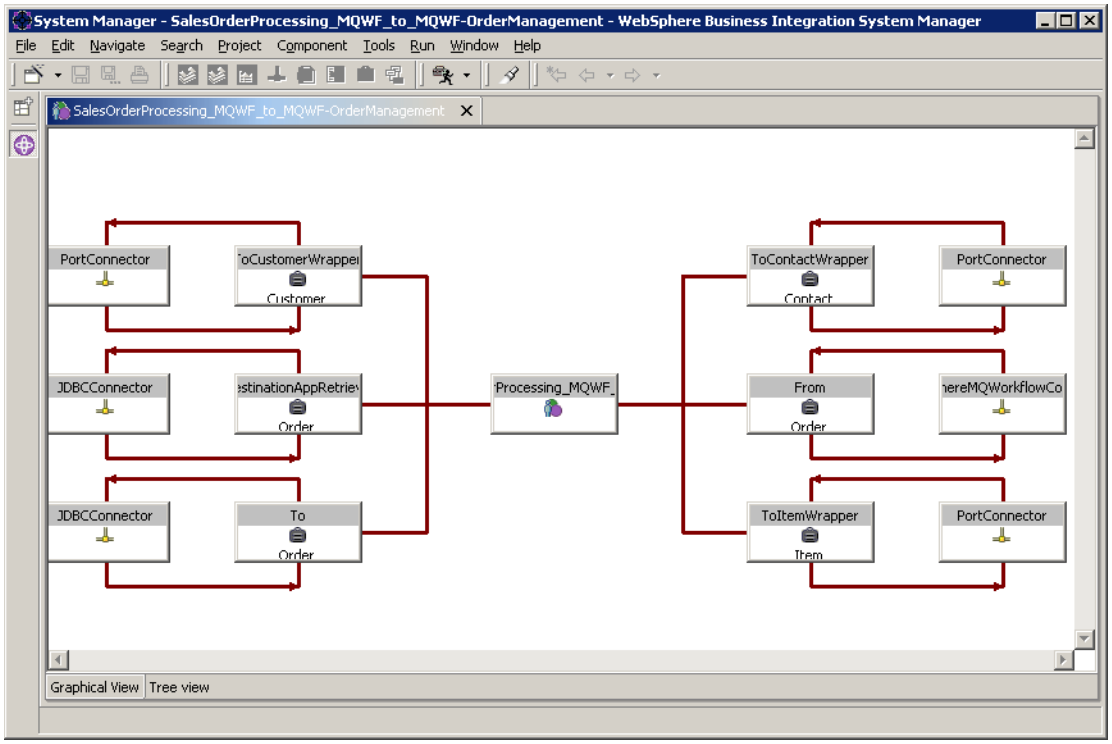

System Manager - SalesOrderProcessing_MQWF_to_MQWF-OrderManagement - WebSphere Business Integration System Manager
File Edit Navigate Search Project Component Tools Run Window Help
SalesOrderProcessing_MQWF_to_MQWF-OrderManagement
PortConnector
'oCustomerWrappe
Customer
JDBCConnector
estinationAppRetrie
Order
JDBCConnector
To
Order
Processing_MQWF_
ToContactWrapper
Contact
PortConnector
From
Order
ıereMQWorkflowCo
ToItemWrapper
Item
PortConnector
Graphical View
Tree view

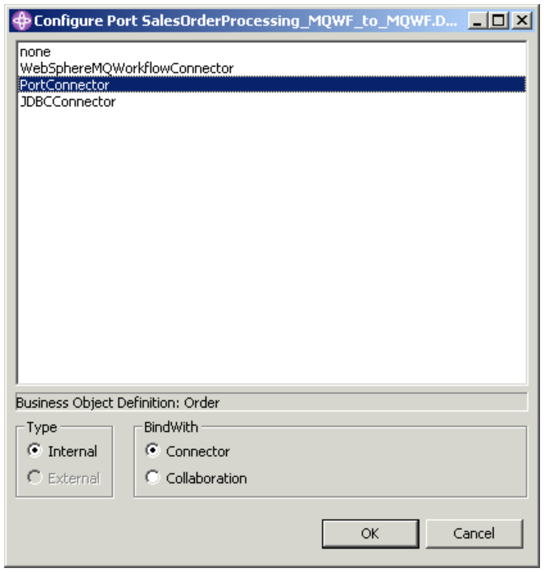
Figure 7-71 Collaboration object

There is another way to bind ports to connectors.

1. Switch to **Tree view** of the collaboration editor (bottom left corner). Right-click an unbound port and select **Bind Port**. A pop-up window (Figure 7-80 on page 348) shows the connectors that have the generic business object of that port in their list of supported business objects.

2. Select the Port connector and repeat this until all ports are connected to the port connector.

Tip: If an expected adapter name does not show as an option to bind to a port, check the business objects that the adapter supports.


Configure Port SalesOrderProcessing_MQWF_to_MQWF.D...

none
WebSphereMQWorkflowConnector
PortConnector
JDBCConnector

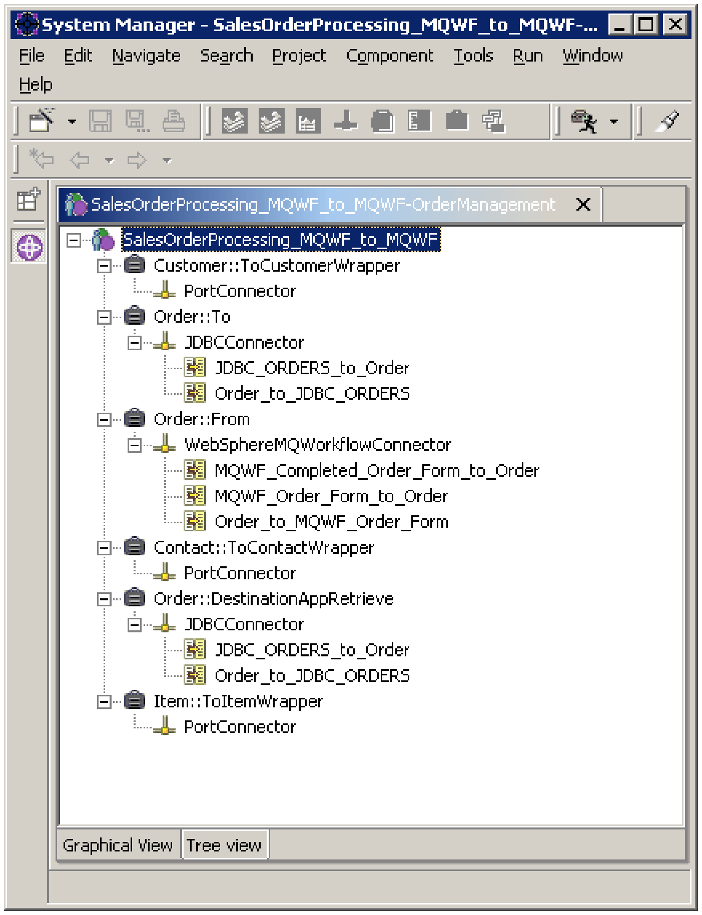
Business Object Definition: Order

Type: [✓ Internal] External

BindWith: [✓ Connector] Collaboration

Figure 7-72 Bind a port to a connector

When all ports are bound to a connector, the tree view of the collaboration looks similar to Figure 7-73.


System Manager - SalesOrderProcessing_MQWF_to_MQWF-...
File Edit Navigate Search Project Component Tools Run Window Help
SalesOrderProcessing_MQWF_to_MQWF-OrderManagement
Graphical View
Tree view

```
SalesOrderProcessing_MQWF_to_MQWF
    Customer::ToCustomerWrapper
        PortConnector
    Order::To
        JDBCConnector
            JDBC_ORDERS_to_Order
            Order_to_JDBC_ORDERS
    Order::From
        WebSphereMQWorkflowConnector
            MQWF_Completed_Order_Form_to_Order
            MQWF_Order_Form_to_Order
            Order_to_MQWF_Order_Form
    Contact::ToContactWrapper
        PortConnector
    Order::DestinationAppRetrieve
        JDBCConnector
            JDBC_ORDERS_to_Order
            Order_to_JDBC_ORDERS
    Item::ToItemWrapper
        PortConnector
```

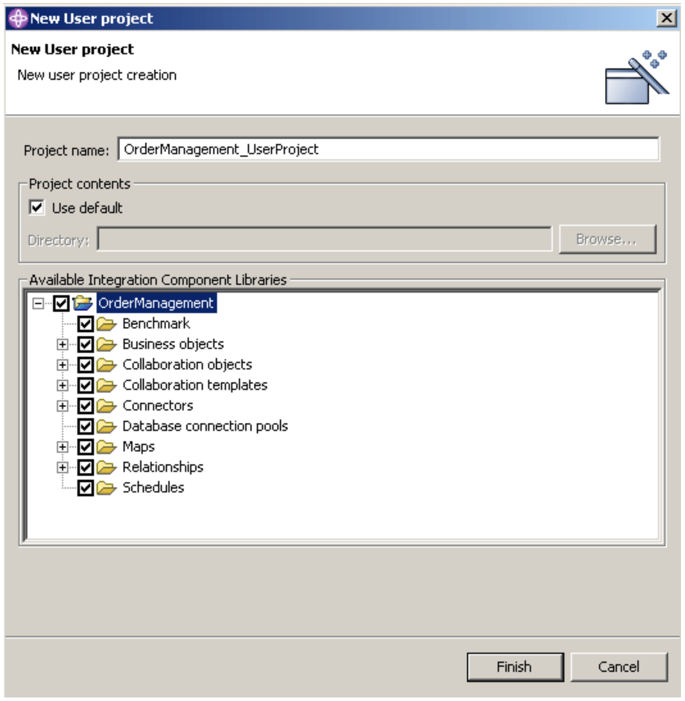
Figure 7-73 Tree view of a fully bound collaboration

7.10 Create a new user project

A common way to organize integration artifacts for the InterChange Server is to create a user project. While an Integration Component Library (ICL) is a file based repository for holding the actual artifacts, a user project is a virtual grouping of components from one or many ICLs. The components shown in a user project are only links to the actual components in the ICL. As a result, if you delete an artifact from the user project, it simply removes the link, while the physical object is still in the ICL.

From a user project one can export the grouped solution to a JAR file, validate the project, or even deploy the solution to the InterChange Server runtime.

1. Within the User Project folder in System Manager, right-click **InterChange Server Projects** and select **New InterChange Server Project**.
2. Provide a name for the user project, for example `OrderManagement`. To populate the user project, select all business objects, maps, and collaboration objects from the ICL OrderManagement. Also select the SalesOrderProcessing collaboration template and the three connectors, as shown in Figure 7-74. Click **Finish**.



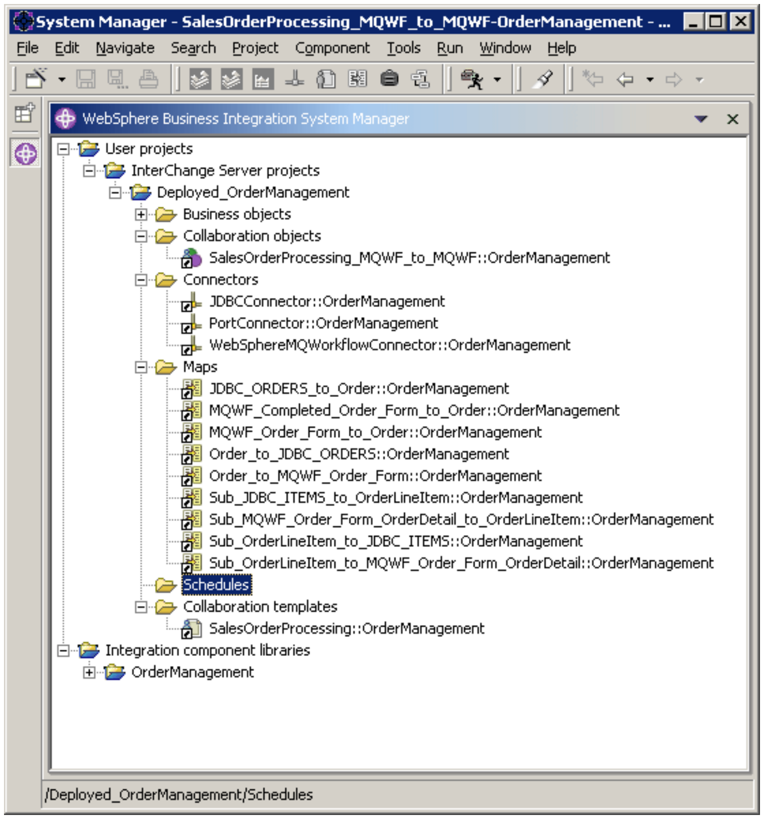
Project name: OrderManagement_UserProject

Project contents
☑ Use default
Directory:

Available Integration Component Libraries
☑ OrderManagement
- ☑ Benchmark
- ☑ Business objects
- ☑ Collaboration objects
- ☑ Collaboration templates
- ☑ Connectors
- ☑ Database connection pools
- ☑ Maps
- ☑ Relationships
- ☑ Schedules

Figure 7-74 Create new user project

Figure 7-75 shows the user project that we created with the objects that we selected.


System Manager - SalesOrderProcessing_MQWF_to_MQWF-OrderManagement - ...
File Edit Navigate Search Project Component Tools Run Window Help
WebSphere Business Integration System Manager

```
User projects
  InterChange Server projects
    Deployed_OrderManagement
      Business objects
      Collaboration objects
        SalesOrderProcessing_MQWF_to_MQWF::OrderManagement
      Connectors
        JDBCConnector::OrderManagement
        PortConnector::OrderManagement
        WebSphereMQWorkflowConnector::OrderManagement
      Maps
        JDBC_ORDERS_to_Order::OrderManagement
        MQWF_Completed_Order_Form_to_Order::OrderManagement
        MQWF_Order_Form_to_Order::OrderManagement
        Order_to_JDBC_ORDERS::OrderManagement
        Order_to_MQWF_Order_Form::OrderManagement
        Sub_JDBC_ITEMS_to_OrderLineItem::OrderManagement
        Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem::OrderManagement
        Sub_OrderLineItem_to_JDBC_ITEMS::OrderManagement
        Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail::OrderManagement
      Schedules
      Collaboration templates
        SalesOrderProcessing::OrderManagement
Integration component libraries
  OrderManagement
```

/Deployed_OrderManagement/Schedules

Figure 7-75 Populated User Project

7.11 Deploy user project

Now we can deploy our user project to the InterChange Server runtime. For this first-time deployment, the InterChange Server does not contain any active components. However, in the general case, you should review the Component Management View in the System Manager, shown in Figure 7-76 on page 344. In this view, you can stop connector controllers and collaboration objects before deploying components that would overwrite those active components.

Note: If you performed unit map testing in a previous section, there might be some artifacts left on the server from that testing. To remove all the components from the running server, you can right-click the server name in the InterChange Server Component Management view and select **Delete the repository**.

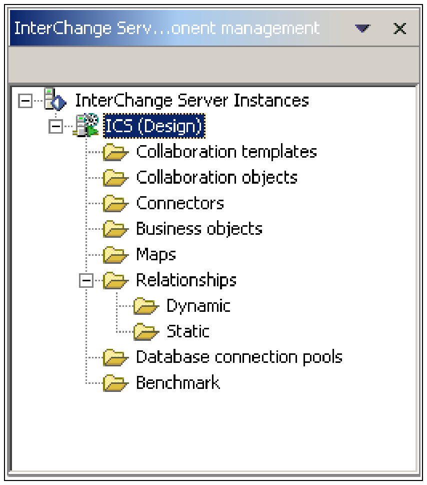


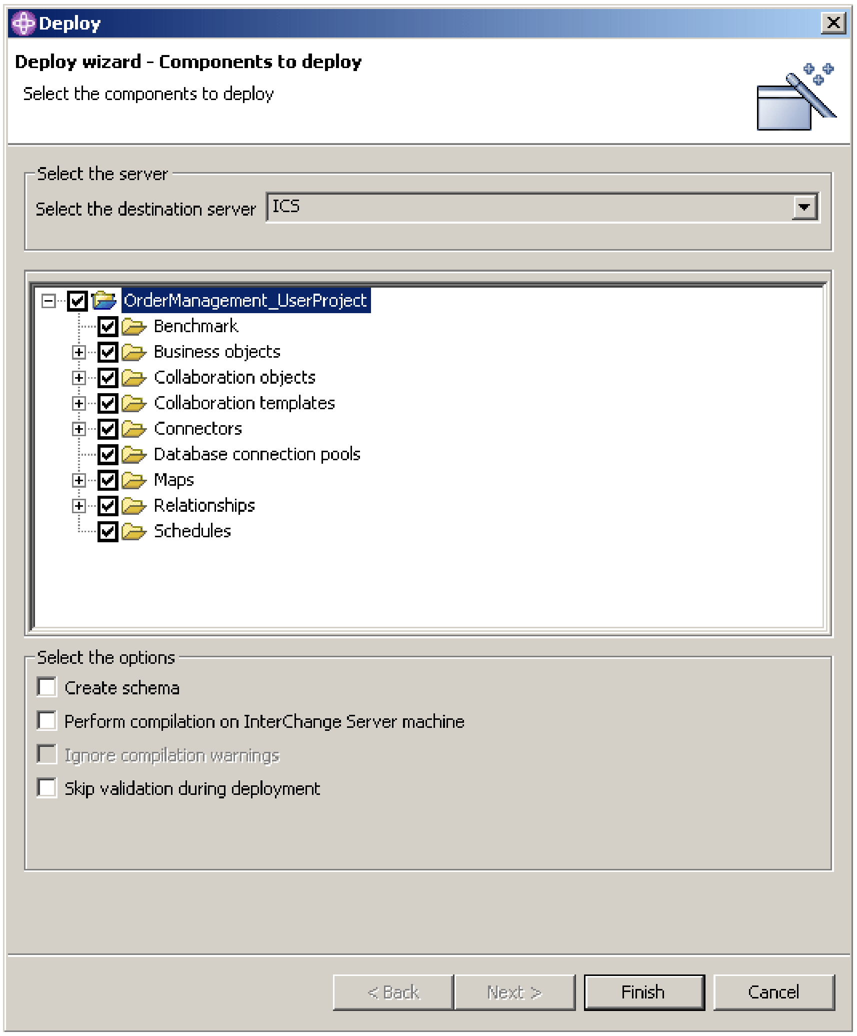
Figure 7-76 Component Management View

Note: An optional step to perform before deploying the user project, is to validate the user project. During validation, the InterChange Server will verify any dependencies between objects. If one object depends on another, then that last object must exist in the server or it must be part of the deployment process.

To perform this validation, right-click the user project and select **Validate user project**.

To deploy the user project there are two options.

- Right-click the user project name and select the **Deploy user project** option where you can select the target server and components from the user project to deploy, as shown in Figure 7-77. Click **Finish** when you are done.


Deploy

Deploy wizard - Components to deploy

Select the components to deploy

Select the server

Select the destination server: ICS

- ☑ OrderManagement_UserProject
  - ☑ Benchmark
  - ☑ Business objects
  - ☑ Collaboration objects
  - ☑ Collaboration templates
  - ☑ Connectors
  - ☑ Database connection pools
  - ☑ Maps
  - ☑ Relationships
  - ☑ Schedules

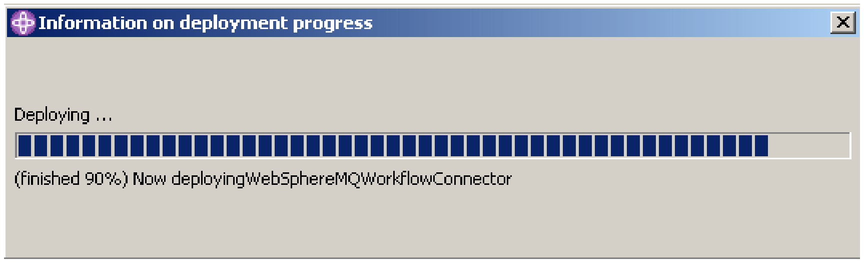
Select the options

- ☐ Create schema
- ☐ Perform compilation on InterChange Server machine
- ☐ Ignore compilation warnings
- ☐ Skip validation during deployment

Figure 7-77 Using the Deploy Wizard to deploy a user project

- The second deployment option is to simply click and drag the user project name down to the ICS server name in the Component Management view and drop the project on the server.

With either of these two options a deployment bar is displayed showing the current status of the deployment procedure (Figure 7-78).

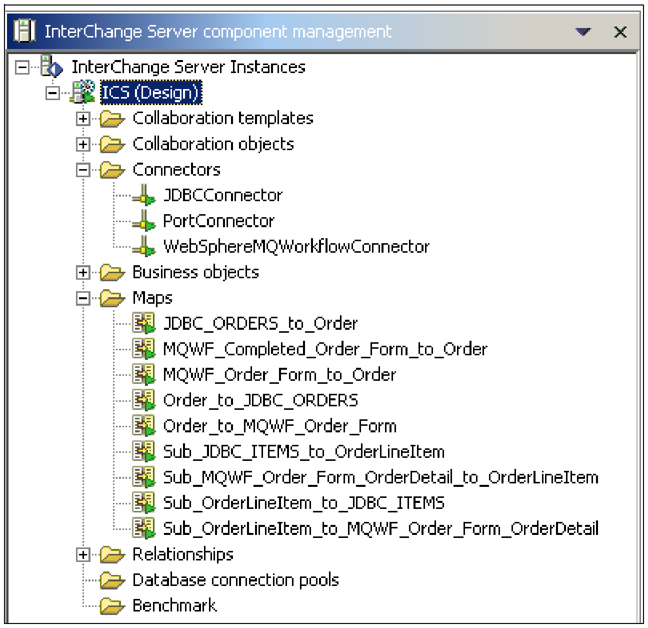

Information on deployment progress

Deploying ...

(finished 90%) Now deployingWebSphereMQWorkflowConnector

Figure 7-78 Deployment status bar

New in WebSphere InterChange Server v4.3 is the ability to deploy components without having to restart the InterChange Server runtime. You can see the active components in Figure 7-79.


InterChange Server component management

```
InterChange Server Instances
  ICS (Design)
    Collaboration templates
    Collaboration objects
    Connectors
      JDBCConnector
      PortConnector
      WebSphereMQWorkflowConnector
    Business objects
    Maps
      JDBC_ORDERS_to_Order
      MQWF_Completed_Order_Form_to_Order
      MQWF_Order_Form_to_Order
      Order_to_JDBC_ORDERS
      Order_to_MQWF_Order_Form
      Sub_JDBC_ITEMS_to_OrderLineItem
      Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem
      Sub_OrderLineItem_to_JDBC_ITEMS
      Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail
    Relationships
    Database connection pools
    Benchmark
```

Figure 7-79 Active InterChange Server components

7.12 Runtime validation of infrastructure

As shown in Figure 7-79 on page 346 the components are now active within the InterChange Server runtime. What is left to test for runtime component validation are the two individual adapters, the WebSphere MQ Workflow adapter and the JDBC adapter.

Starting the adapters

To start these adapters, click the short cut for each:

- **Start → Programs → IBM WebSphere Business Integration Adapters → Adapters → Connectors → WebSphere MQ Workflow Connector**
- **Start → Programs → IBM WebSphere Business Integration Adapters → Adapters → Connectors → JDBC Connector**

Each connector runs by default in its own command window where, by default, it displays logging and tracing to standard out. Depending on your settings for agent trace level, these log messages can be numerous and new log messages can be generated almost constantly.

One way to look at these log messages without scrolling away almost immediately is to click in the command window. The title bar shows the name of the connector prefixed with the word `Select`. By doing this, the scrolling stops.

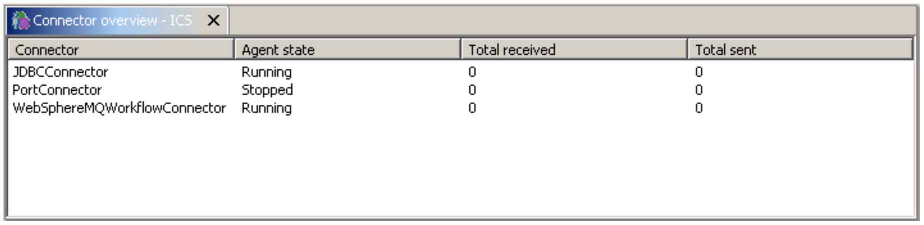
Important: Be aware that when you click in the command window, the connector pauses as well. As a result, you one must be careful when doing this in a running system because it can have adverse effects.

At this time, no application messages are expected because we just started the adapters for the first time, so look for the ready message. You can see this message at the bottom of the standard out window and is displayed as:

```
[Mesg: Connector Agent state is active.]
```

Instead of scrolling hundreds of log messages, you could expand the InterChange Server entry in the Component Management View in the System Manager. This shows the different connectors, maps, and collaborations, each prefixed with a little status icon (Figure 7-79 on page 346).

For connectors, this view merely shows the status of the connector controller. To view the status of the connector agents, right-click the **Connectors** folder and select **Overview**. This opens a new view in the System Manager (Figure 7-80).



Connector	Agent state	Total received	Total sent
JDBCConnector	Running	0	0
PortConnector	Stopped	0	0
WebSphereMQWorkflowConnector	Running	0	0

Figure 7-80 Status of connector agents

In this view, the agent state is running for the JDBC connector and the WebSphere MQ Workflow connector. Note that the Port connector is in a stopped state, which is normal.

Troubleshooting a stopped connector agent

If one of the connector agents does not show as running, the first step is to discover whether the connector controller is running correctly. This means verifying the status icon in front of the connector in the Component Management View:

- Green for running
- Red for stopped
- Two black bars for paused

Obviously, if the connector controller is not working correctly, the connector agent will not be able to communicate with it. Thus, the first task is to see why the connector controller is not starting correctly. You can try to restart it and then verify immediately the output of the InterChange Server. The most common problem is a setup problem with the underlying queues. For example, the queues might be defined in the wrong case or the names might not match due to a spelling mistake. Verify whether the queue manager is running. Errors at this level are usually caused by wrong information about the Standard Properties tab of the Connector Configurator.

After errors at the controller level are fixed and the connector controller is running correctly, you can then focus on the agent itself. Two broad categories of problems can occur.

- The connector agent cannot communicate with the controller due to a mismatch between the configuration file and the actual configuration. The most important settings in the configuration file are the settings about communication with the connector controller, such as queue names and queue manager names. When communication is established, the remaining configuration parameters are pushed from the server to the agent.

- You might have forgotten to save your latest changes to the configuration file and instead saved them only to the project. Alternatively, you might have pointed to the wrong configuration file. Check that the configuration file contains your latest changes.

When communication happens between controller and agent, the agent receives its runtime parameters, which can be verified in the standard out window. When the agent has received its runtime parameters, it uses them to connect to the application. For the WebSphere MQ Workflow connector, the term *application* means: Can it communicate with the configured queue manager that is defined with interaction with the WebSphere MQ Workflow server? For the JDBC connector, it means: Can it connect to the database and can it find its event tables? Thus, if errors occur at this level, they are caused by errors on the Connector-Specific Properties tab of the Connector Configurator.

7.13 Runtime validation of integration solution

To validate our integration solution, it is a good practice to perform component testing, unit testing, and end-to-end testing.

Component testing includes validating that the individual adapters successfully start (which involves connecting with the target application) as discussed in 7.12, “Runtime validation of infrastructure” on page 347 and using the Map Designer’s test environment to assure the intended data transformations are occurring, discussed in , “Testing the map” on page 300.

Unit testing involves testing the integration interfaces, namely the automated process defined by the collaboration and the interaction between the collaboration and the source and target systems. The most efficient fashion in which to perform unit testing with the collaboration is to first use the Integrated Test Environment. Then to test interaction between one actual source or target system and the collaboration a Virtual Test Connector is often used. These two approaches are described in detail in this section.

End-to-end testing is usually done with the adapters having live connections to development source and target systems, where those source and target systems are providing production type of data events and responses.

7.13.1 Unit testing with the Integrated Test Environment

The order and content of the information in this section was based upon the “Using the Integrated Test Environment” chapter in the *System Implementation Guide* that is located at this Web site:

`http://publib.boulder.ibm.com/infocenter/wbihelp/index.jsp`

To perform a test in the Integrated Test Environment, you must do a number of tasks such as register InterChange Server as a test server, create a test unit, deploy the components in the interface to the server, start the server, emulate the connectors in the interface, and exchange business objects between the connectors. The following characteristics describe the use of Integrated Test Environment:

- ► You only have to perform some of the tasks a single time. For instance, you only have to create a test unit for an interface once.
- ► You have to perform some tasks multiple times. For instance, you might test how an interface responds when you change the value in a particular attribute, so you will have to send business object requests for the interface multiple times.
- ► You can perform some tasks in multiple ways. For instance, you can deploy components to the server before you prepare the test unit, or you can deploy all of the components for a test unit by using the Task Manager view, or you can deploy single components by using the Test Unit view.

Note: New in WebSphere InterChange Server V4.3.0 is the ability to use the Integrated Test Environment to connect to a remotely running InterChange Server.

The next portion of this section describes the sequence which you would typically follow to perform a test. This section incorporates most of the subtasks and involves most of the interface elements. This section also recommends the most efficient and effective techniques in situations where there are multiple ways of accomplishing a subtask.

1. Although you can deploy components to an InterChange Server instance by using Integrated Test Environment, it is recommended that you perform all deployment activities beforehand for the following reasons:
   - – You avoid having to compile maps and collaboration templates as part of the testing process.
   - – You can start the components prior to the testing stage as well; components must be deployed before they can be started. When you deploy connectors you must restart the server to start them, and almost

any interface involves a connector, so it is typically not efficient to deploy the components for an interface as part of the testing process.

  – If you have to test several interfaces you can do a single deployment prior to testing, rather than having to make sure you oversee the proper deployment for each interface during testing.

2. Ensure that all of the components required to test the interface are in an active state.

   To start components, use one of the following interfaces:

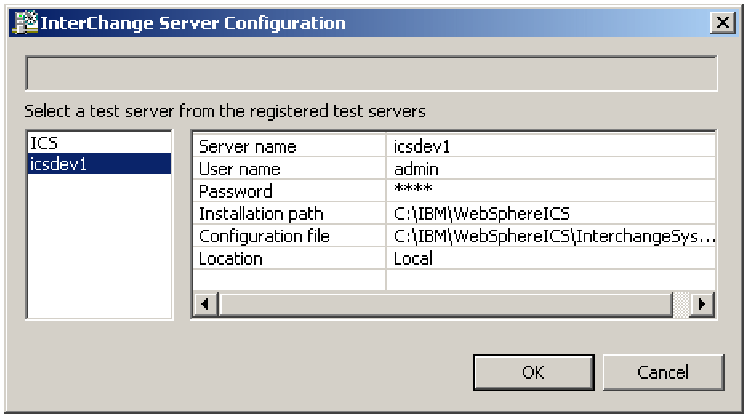
   – System Monitor, as described earlier in this book
   – InterChange Server Component Management view,
   – Managing component states using the Test Unit view in the Integrated Test Environment

3. Register the InterChange Server you want to test with as a Test Server as described in.

4. Start the Integrated Test Environment perspective as described in

5. Select the server you want to test with as described in Selecting a server configuration.

   Select **Integrated Test Environment** → **Test Server Configuration** from the menu bar of the Integrated Test Environment. Select the test server (Figure 7-81) to which to connect and click **OK**.


InterChange Server Configuration
Select a test server from the registered test servers
ICS
icsdev1
OK
Cancel

Server name	icsdev1
User name	admin
Password	*****
Installation path	C:\IBM\WebSphereICS
Configuration file	C:\IBM\WebSphereICS\InterchangeSys...
Location	Local

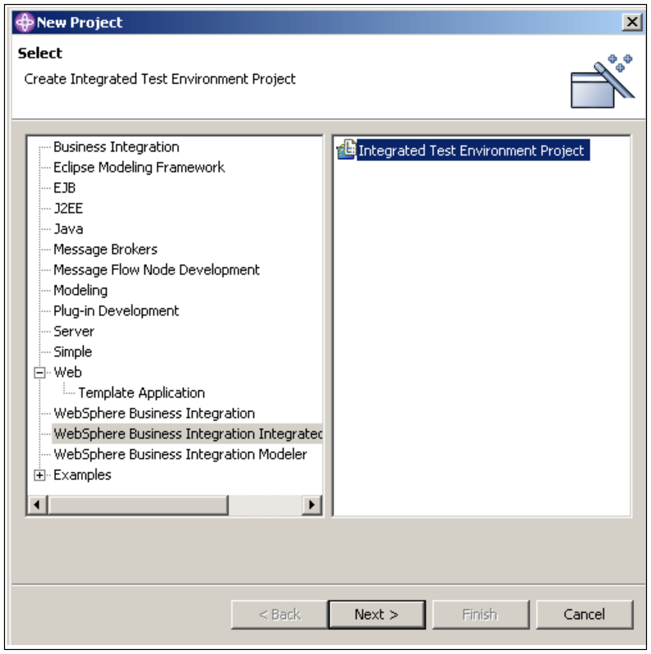
Figure 7-81 Testing Server Configuration for Integration Test Environment

Tip: If the server instance you want to use is not listed in the dialog, try deleting it from the Server Instances view and re-registering it.

6. Create a test project to contain the test unit.

   Do the following to create a test project to store the individual test units you will create:

   a. Select **File** → **New** → **Integrated Test Environment Project** from the menu bar (Figure 7-82).

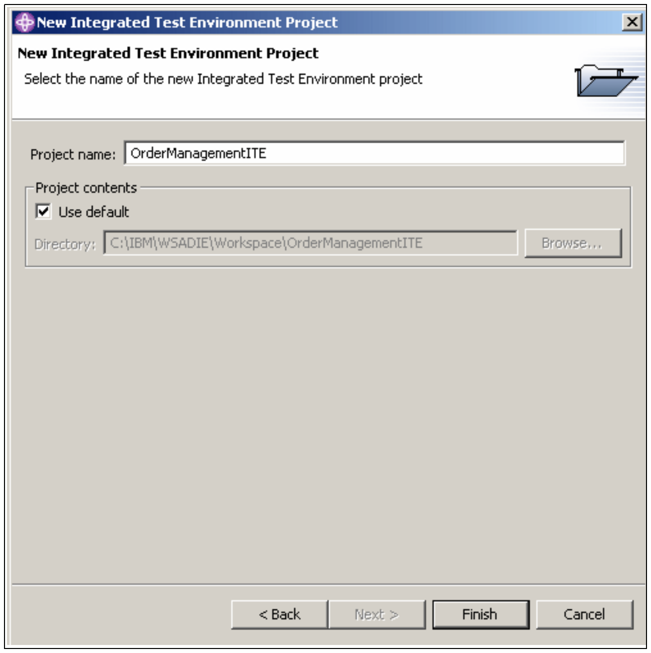


New Project
Select
Create Integrated Test Environment Project
Business Integration
Eclipse Modeling Framework
EJB
J2EE
Java
Message Brokers
Message Flow Node Development
Modeling
Plug-in Development
Server
Simple
Web
Template Application
WebSphere Business Integration
WebSphere Business Integration Integratec
WebSphere Business Integration Modeler
Examples
Integrated Test Environment Project
< Back
Next >
Finish
Cancel

Figure 7-82 Creating new Integration Test Environment Project

b. At the New Integrated Test Environment Project screen, type a name for the test project in the Project name field. We used `OrderManagementITE`.

   Project names can only contain alphanumeric characters and underscores, and must be specified in English.



New Integrated Test Environment Project

New Integrated Test Environment Project

Select the name of the new Integrated Test Environment project

Project name: OrderManagementITE

Project contents

☑ Use default

Directory: C:\IBM\WSADIE\Workspace\OrderManagementITE

Figure 7-83 Naming new Integration Test Environment Project

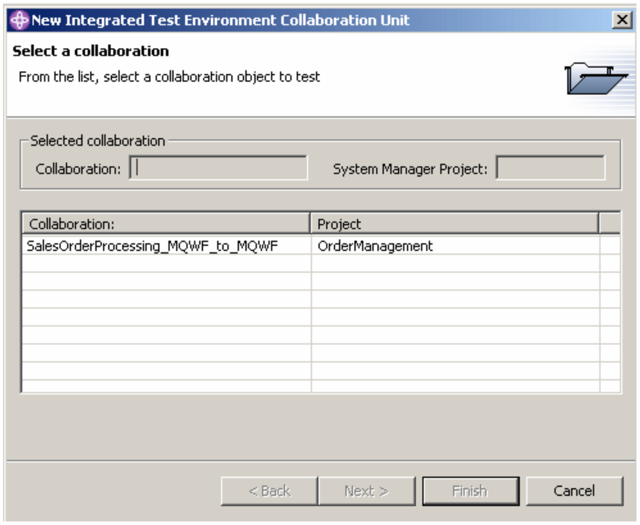
7. Create a test unit for the interface you want to test.

   You can either create a test unit from within Integrated Test Environment, or from within System Manager.

   – Do the following to create a test unit from within Integrated Test Environment:

      i. Select **File** → **New** → **Integrated Test Environment Test Unit** from the menu bar.

ii. At the Select collaboration screen (Figure 7-84), select the collaboration object you want to test from the list of all the collaboration objects in all the integration component libraries defined in the system.



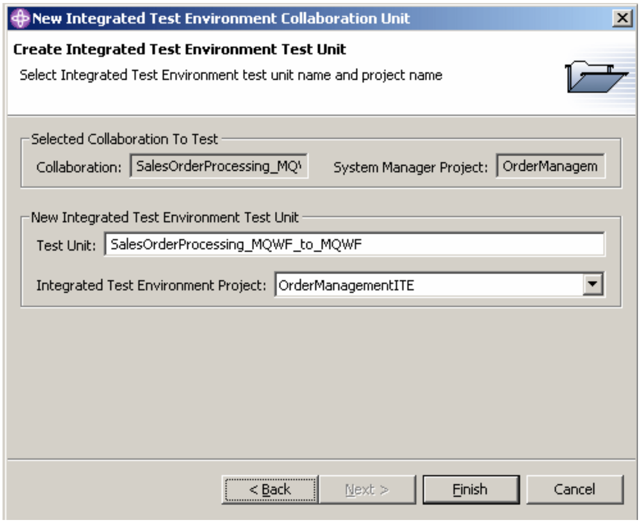
Collaboration:	Project
SalesOrderProcessing_MQWF_to_MQWF	OrderManagement

Figure 7-84 Adding unit from Integration Test Environment

– Do the following to create a test unit from within the System Manager:

Change to the System Manager perspective and right-click the collaboration object that represents the interface. Select **Debug in Integrated Test Environment** from the context menu.

With either method used to create the Test Unit, the Create Integrated Test Environment Test Unit screen will open (Figure 7-85). At this screen, type a name for the test unit in the Test Unit field, and select the test project it should be created in from the **Integrated Test Environment Project** drop-down menu.


New Integrated Test Environment Collaboration Unit

Create Integrated Test Environment Test Unit

Select Integrated Test Environment test unit name and project name

Selected Collaboration To Test

Collaboration: SalesOrderProcessing_MQ\ System Manager Project: OrderManagem

New Integrated Test Environment Test Unit

Test Unit: SalesOrderProcessing_MQWF_to_MQWF

Integrated Test Environment Project: OrderManagementITE

Figure 7-85 Complete Test Unit creation

8. It is recommended, but not required, that you deploy the components outside of the Integrated Test Environment. If you plan to deploy the components in the interface you are testing using Integrated Test Environment, do the following:

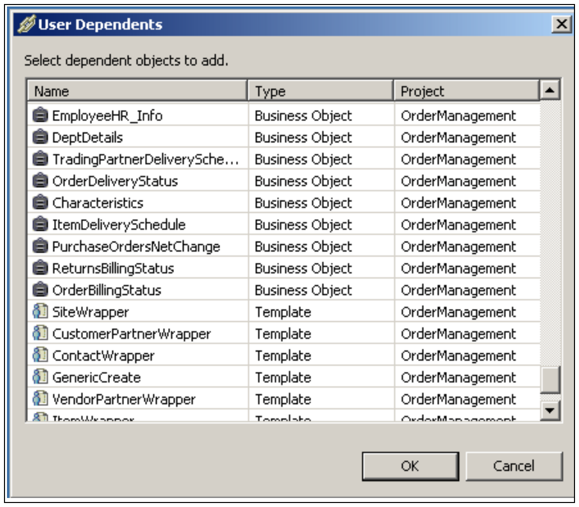
   a. Verify that all the artifacts that you deployed are displayed in the Dependents section.

      In the Outline view, expand the Artifacts node and then the Dependents node.

   b. If you want to add additional dependents to the list, you can:

      i. In the Outline view, expand the Artifacts node, right-click the **User Artifacts** node and select **Add User Artifacts** from the context menu.

ii. At the User Dependents window (Figure 7-86) select the components you want to add. You can use standard multiple-selection techniques, such as holding down **Shift** to select contiguous rows and holding down **Ctrl** to select noncontiguous rows.



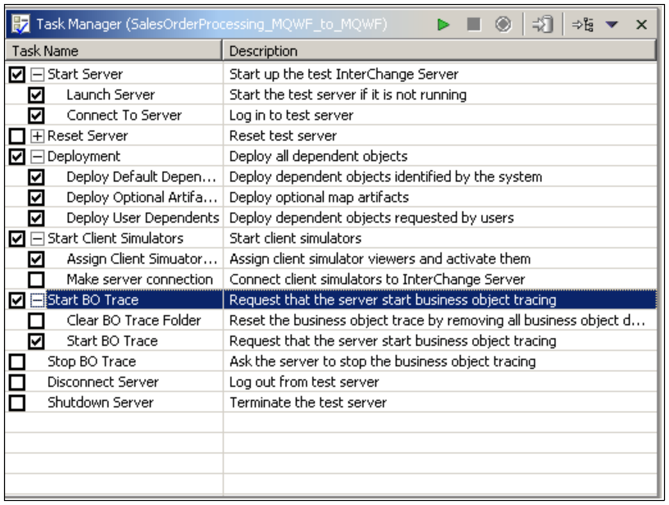
Select dependent objects to add.

Name	Type	Project
EmployeeHR_Info	Business Object	OrderManagement
DeptDetails	Business Object	OrderManagement
TradingPartnerDeliverySche...	Business Object	OrderManagement
OrderDeliveryStatus	Business Object	OrderManagement
Characteristics	Business Object	OrderManagement
ItemDeliverySchedule	Business Object	OrderManagement
PurchaseOrdersNetChange	Business Object	OrderManagement
ReturnsBillingStatus	Business Object	OrderManagement
OrderBillingStatus	Business Object	OrderManagement
SiteWrapper	Template	OrderManagement
CustomerPartnerWrapper	Template	OrderManagement
ContactWrapper	Template	OrderManagement
GenericCreate	Template	OrderManagement
VendorPartnerWrapper	Template	OrderManagement
ItemWrapper	Template	OrderManagement

Figure 7-86 Add user dependents

9. Make sure the IBM Java Object Request Broker is started on the machine where the InterChange Server is installed. This is accomplished by launching the Persistent Name Server.
10. Use the Task Manager view to start the server (if it is not already started), bind the Integrated Test Environment agent to it, and connect the Integrated Test Environment to it.

This is accomplished by checking or unchecking the desired options in the Task Manager view and then clicking the green Play button on the Task Manager view menu bar (Figure 7-87).

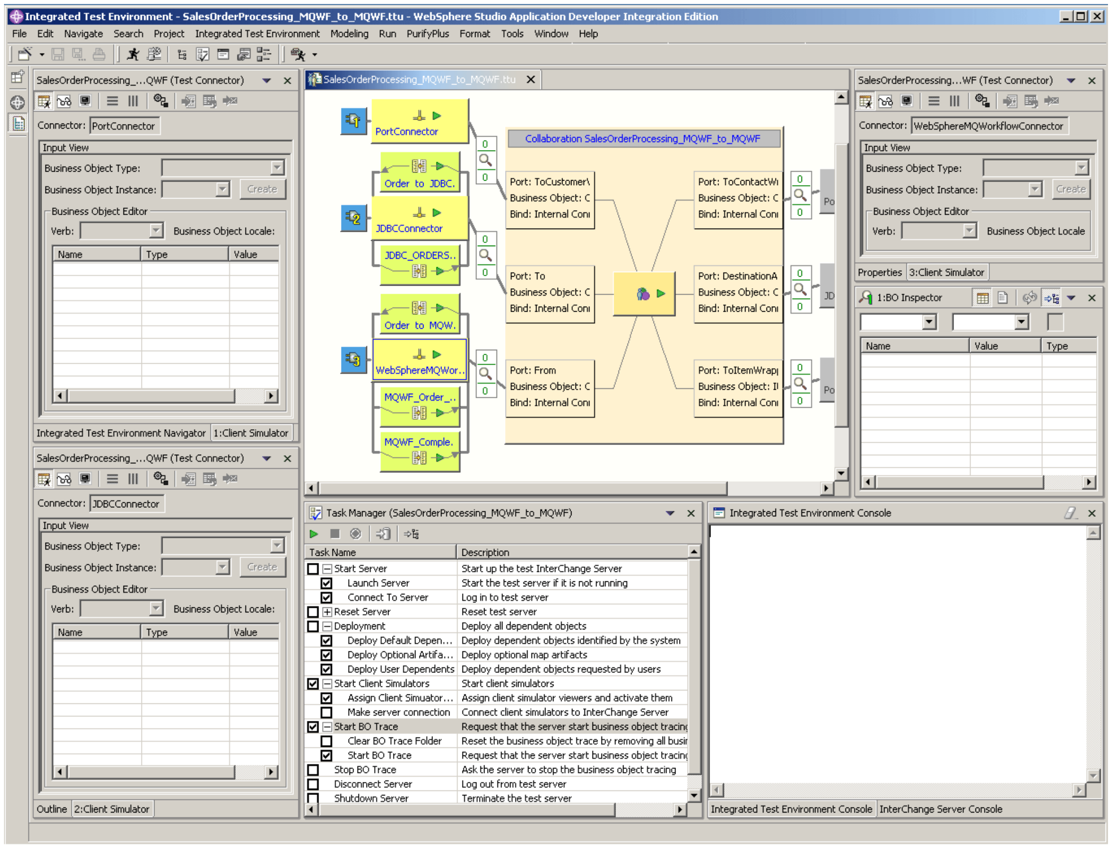


Task Manager (SalesOrderProcessing_MQWF_to_MQWF)

Task Name	Description
☑ ⊟ Start Server	Start up the test InterChange Server
☑ Launch Server	Start the test server if it is not running
☑ Connect To Server	Log in to test server
☐ ⊞ Reset Server	Reset test server
☑ ⊟ Deployment	Deploy all dependent objects
☑ Deploy Default Depen...	Deploy dependent objects identified by the system
☑ Deploy Optional Artifa...	Deploy optional map artifacts
☑ Deploy User Dependents	Deploy dependent objects requested by users
☑ ⊟ Start Client Simulators	Start client simulators
☑ Assign Client Simuator...	Assign client simulator viewers and activate them
☐ Make server connection	Connect client simulators to InterChange Server
☑ ⊟ Start BO Trace	Request that the server start business object tracing
☐ Clear BO Trace Folder	Reset the business object trace by removing all business object d...
☑ Start BO Trace	Request that the server start business object tracing
☐ Stop BO Trace	Ask the server to stop the business object tracing
☐ Disconnect Server	Log out from test server
☐ Shutdown Server	Terminate the test server

Figure 7-87 Integrated Test Environment Task Manager

When the Integrated Test Environment is bound to the running InterChange Server, the interface looks similar to Figure 7-88.


Integrated Test Environment - SalesOrderProcessing_MQWF_to_MQWF.ttu - WebSphere Studio Application Developer Integration Edition
File Edit Navigate Search Project Integrated Test Environment Modeling Run PurifyPlus Format Tools Window Help
SalesOrderProcessing_...QWF (Test Connector)
Connector: PortConnector
Input View
Business Object Type:
Business Object Instance:
Create
Business Object Editor
Verb:
Business Object Locale:
Name
Type
Value
Integrated Test Environment Navigator
1:Client Simulator
SalesOrderProcessing_...QWF (Test Connector)
Connector: JDBCConnector
Outline
2:Client Simulator
SalesOrderProcessing_MQWF_to_MQWF.ttu
PortConnector
Order to JDBC.
JDBCConnector
JDBC_ORDERS..
Order to MQW.
WebSphereMQWor..
MQWF_Order_..
MQWF_Comple.
Collaboration SalesOrderProcessing_MQWF_to_MQWF
Port: ToCustomer\
Business Object: C
Bind: Internal Con
Port: To
Port: From
Port: ToContactWi
Port: DestinationA
Port: ToItemWrap
Business Object: I
SalesOrderProcessing...WF (Test Connector)
Connector: WebSphereMQWorkflowConnector
Properties
3:Client Simulator
1:BO Inspector
Integrated Test Environment Console
InterChange Server Console

Task Manager (SalesOrderProcessing_MQWF_to_MQWF)

Task Name	Description
☐ ⊟ Start Server	Start up the test InterChange Server
☑ Launch Server	Start the test server if it is not running
☑ Connect To Server	Log in to test server
☐ ⊞ Reset Server	Reset test server
☐ ⊟ Deployment	Deploy all dependent objects
☑ Deploy Default Depen…	Deploy dependent objects identified by the system
☑ Deploy Optional Artifa…	Deploy optional map artifacts
☑ Deploy User Dependents	Deploy dependent objects requested by users
☑ ⊟ Start Client Simulators	Start client simulators
☑ Assign Client Simuator…	Assign client simulator viewers and activate them
☐ Make server connection	Connect client simulators to InterChange Server
☑ ⊟ Start BO Trace	Request that the server start business object tracing
☐ Clear BO Trace Folder	Reset the business object trace by removing all busi
☑ Start BO Trace	Request that the server start business object tracing
☐ Stop BO Trace	Ask the server to stop the business object tracing
☐ Disconnect Server	Log out from test server
☐ Shutdown Server	Terminate the test server

Figure 7-88 Integrated Test Environment bound to icsdev1 for OrderManagement integration project

For detailed information about options within the Task Manager view, review the Using Integrated Test Environment section of the System Implementation guide located at this Web site:

`http://publib.boulder.ibm.com/infocenter/wbihelp/index.jsp`

11. Enable the Server Context Overlay by right-clicking the background of the Test Unit view and select Server Context Overlay from the context menu. By default, the Server Context Overlay is already enabled.

    The Server Context Overlay displays information about the components in the interface within the Test Unit view and enables context menu options for deploying components and manipulating their states.

12. Show Client Simulator views for the clients in the interface.

It is recommended that you organize the Client Simulator views in a way that makes sense to you. For instance, you might find it easiest to have the view for the source connector in position 1 in the perspective (shared with the Integrated Test Environment Navigator view), and to have the view for the destination connector in position 4 in the perspective (shared with the Properties view).

In the Eclipse-based tooling, it is easy to change the views by simply clicking a view's title bar and dragging the view to a different location on the screen. Feel free to experiment with different configurations of the views. Knowing that you can easily reset the views by clicking the Windows menu option and selecting the individual view or resetting the perspective.

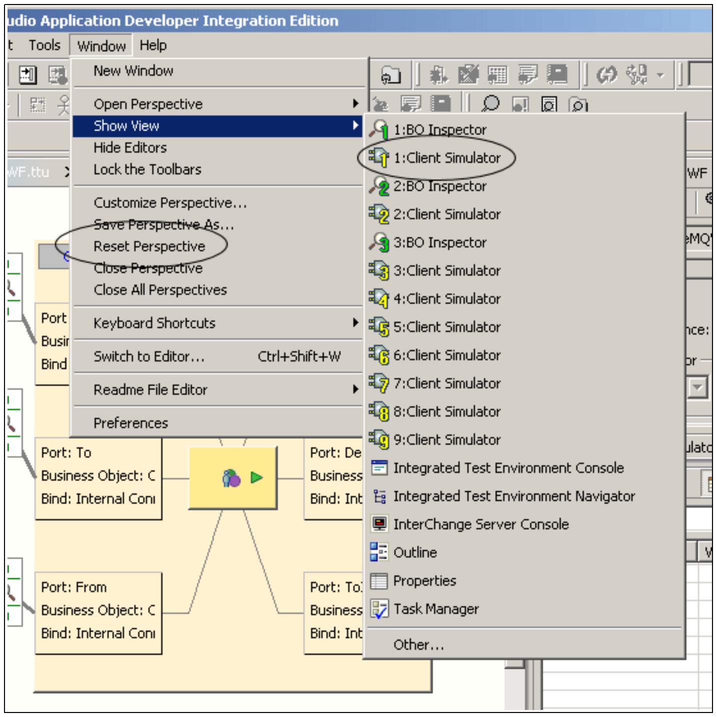
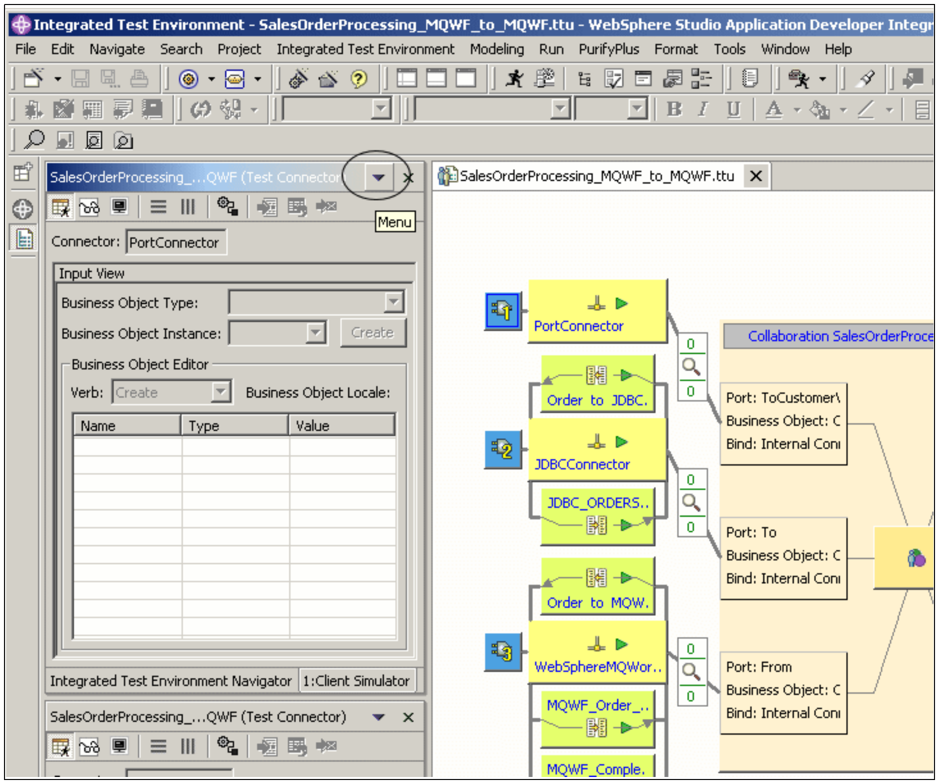


Figure 7-89 Working with views and resetting perspectives

Tip: If you reset the perspective, you can restart the Client Simulator from the Task Manager to bring all the Client Simulators views to the forefront.

13.Connect the Client Simulator views to the server. This can be accomplished by using the adapter configuration stored in the InterChange Server repository or by the externally saved configuration file. Each client simulator to be used in testing needs to be started.

To start the client simulators, click the menu button on the title bar of the targeted Client Simulator view (Figure 7-90), then click **Server** and your connection option, either **Connect** or **Connect with *.cfg**.



Integrated Test Environment - SalesOrderProcessing_MQWF_to_MQWF.ttu - WebSphere Studio Application Developer Integr
File Edit Navigate Search Project Integrated Test Environment Modeling Run PurifyPlus Format Tools Window Help
SalesOrderProcessing_...QWF (Test Connector)
Menu
Connector: PortConnector
Input View
Business Object Type:
Business Object Instance:
Create
Business Object Editor
Verb: Create
Business Object Locale:
Name
Type
Value
Integrated Test Environment Navigator
1:Client Simulator
SalesOrderProcessing_...QWF (Test Connector)
SalesOrderProcessing_MQWF_to_MQWF.ttu
PortConnector
Order to JDBC.
JDBCConnector
JDBC_ORDERS..
Order to MQW.
WebSphereMQWor..
MQWF_Order_..
MQWF_Comple.
Collaboration SalesOrderProce
Port: ToCustomer\
Business Object: C
Bind: Internal Con
Port: To
Business Object: C
Bind: Internal Con
Port: From
Business Object: C
Bind: Internal Con

Figure 7-90 Client Simulator connection menu

j

Tip: The DeliveryTransport property of the adapter you are emulating must be set to the value **IDL** to connect to the server using the repository definition. Otherwise, you must use the Connect with *.cfg option.

When the Status View for that client simulator displays `Ready`, it has successfully connected.

After you have confirmed that the clients connected to the server successfully, configure the Client Simulator view for the source connector to use the Input Pane by clicking the **Input Pane** button located at the top of that particular client simulator view. For the destination connector, click the to **Result Pane** button.

14. If you want to use business object tracing, start it at this point so that the data is captured when you begin to send business objects in the next steps. To start business object tracing use the Task Manager view.

15. Use the Outline view to confirm that the interface is ready for testing. Click the **Outline** tab in the bottom left corner of the Integrated Test Environment window and verify that all the required components are highlighted in green.

16. Do the following to create and send a business object request from the source connector:

    a. Create a business object instance within the WebSphereMQWorkflowConnector client window by:

        i. In the Input Pane, select the name of the business object you want to create from the Business Object Type drop-down menu.

        ii. Click **Create** next to the Business Object Instance field.

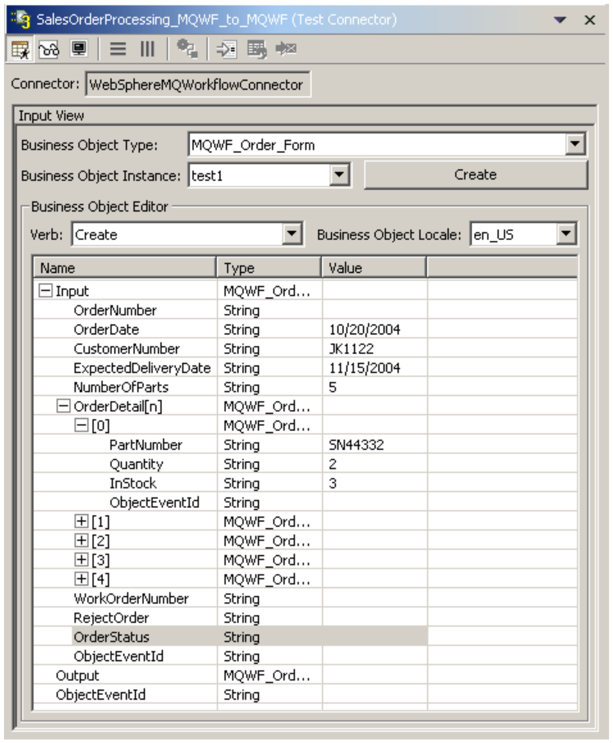
        iii. When presented with the New Instance dialog, type a name for the instance in the Enter Name field, any name can be used.

        iv. Select the desired verb from the Verb drop-down menu.

        v. Select the desired locale from the Business Object Locale drop-down menu.

        vi. Provide values for the simple attributes and child business objects within the top-level object.

        i. Click **OK**. See Figure 7-91 on page 362.



Connector: WebSphereMQWorkflowConnector

Input View

Business Object Type: MQWF_Order_Form

Business Object Instance: test1

Business Object Editor

Verb: Create

Business Object Locale: en_US

Name	Type	Value
⊟ Input	MQWF_Ord...	
OrderNumber	String	
OrderDate	String	10/20/2004
CustomerNumber	String	JK1122
ExpectedDeliveryDate	String	11/15/2004
NumberOfParts	String	5
⊟ OrderDetail[n]	MQWF_Ord...	
⊟ [0]	MQWF_Ord...	
PartNumber	String	SN44332
Quantity	String	2
InStock	String	3
ObjectEventId	String	
⊞ [1]	MQWF_Ord...	
⊞ [2]	MQWF_Ord...	
⊞ [3]	MQWF_Ord...	
⊞ [4]	MQWF_Ord...	
WorkOrderNumber	String	
RejectOrder	String	
OrderStatus	String	
ObjectEventId	String	
Output	MQWF_Ord...	
ObjectEventId	String	

Figure 7-91 Creating the Business Object instance in the client simulator

b. Save the business object instance to a file to be used in subsequent tests by clicking the menu triangle button on the right side of the client simulator title bar, then by clicking **Edit** and **Save All Business Objects**.

c. Send the business object instance as a synchronous request by first changing to synchronous mode.

   To change to synchronous mode, click the menu button, then **Server**, **Mode**, and select synchronous object requests synchronously as appropriate.

   To send the business object instance click the send business object button and select the appropriate collaboration to which to post the business object (Figure 7-92 on page 363).

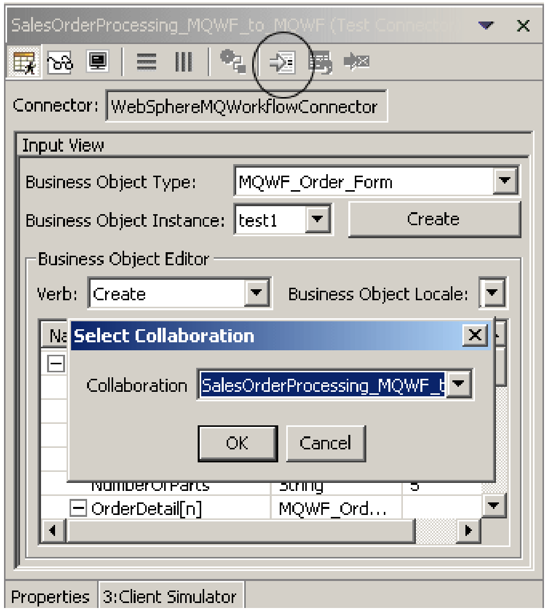
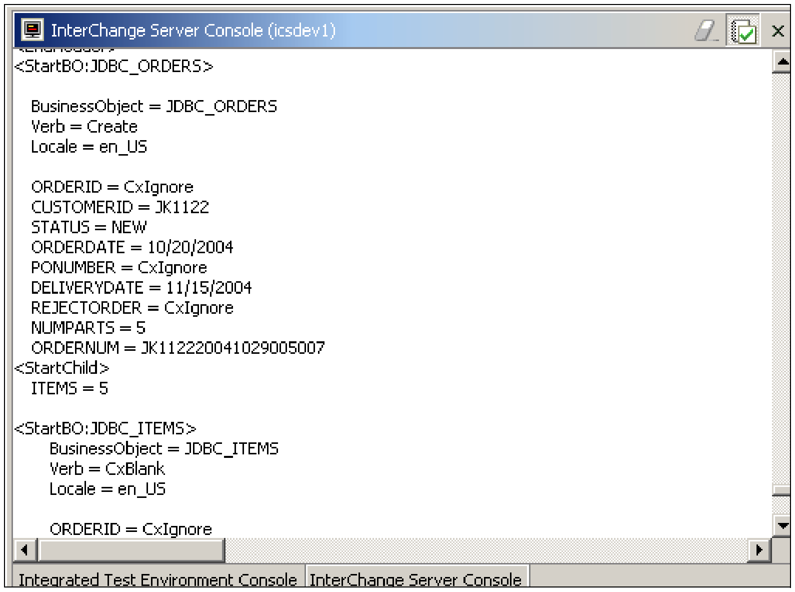

SalesOrderProcessing_MQWF_to_MQWF (Test Connector)
Connector: WebSphereMQWorkflowConnector
Input View
Business Object Type: MQWF_Order_Form
Business Object Instance: test1
Create
Business Object Editor
Verb: Create
Business Object Locale:
Select Collaboration
Collaboration SalesOrderProcessing_MQWF_t
OK
Cancel
NumberOfParts String 5
OrderDetail[n] MQWF_Ord...
Properties
3:Client Simulator

Figure 7-92 Submit business object

17. Use the InterChange Server Console view (Figure 7-93) to observe the processing of the business object.


InterChange Server Console (icsdev1)
Integrated Test Environment Console
InterChange Server Console

```
<StartBO:JDBC_ORDERS>

   BusinessObject = JDBC_ORDERS
   Verb = Create
   Locale = en_US

   ORDERID = CxIgnore
   CUSTOMERID = JK1122
   STATUS = NEW
   ORDERDATE = 10/20/2004
   PONUMBER = CxIgnore
   DELIVERYDATE = 11/15/2004
   REJECTORDER = CxIgnore
   NUMPARTS = 5
   ORDERNUM = JK112220041029005007
<StartChild>
   ITEMS = 5

<StartBO:JDBC_ITEMS>
      BusinessObject = JDBC_ITEMS
      Verb = CxBlank
      Locale = en_US

      ORDERID = CxIgnore
```

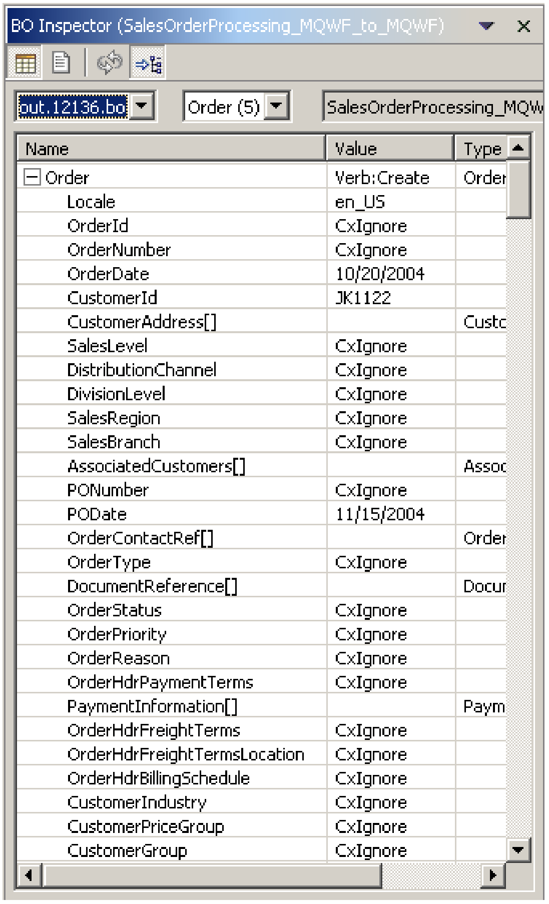
Figure 7-93 Integrated Test Environment InterChange Server Console view

18. Examine the business objects as different components finish processing it (Figure 7-94 on page 365).

    If you use business object tracing, Integrated Test Environment gathers information about a business object while the system processes the business object. Integrated Test Environment captures an image of the business object after it has been processed by a map and after it has been processed by a collaboration.

    For example, if you are testing an interface in which a connector sends a business object request to a collaboration object which sends it to a destination connector, which processes it and then returns a response, then Integrated Test Environment captures the following business object data:

    – The generic business object produced by the map that is called by the source connector when sending the business object request to InterChange Server
    – The generic business object supplied as input to the map that is called by the destination connector
    – The generic business object produced by the map that is called by the destination connector when returning the business object response to InterChange Server
    – Exception messages related to failed flows



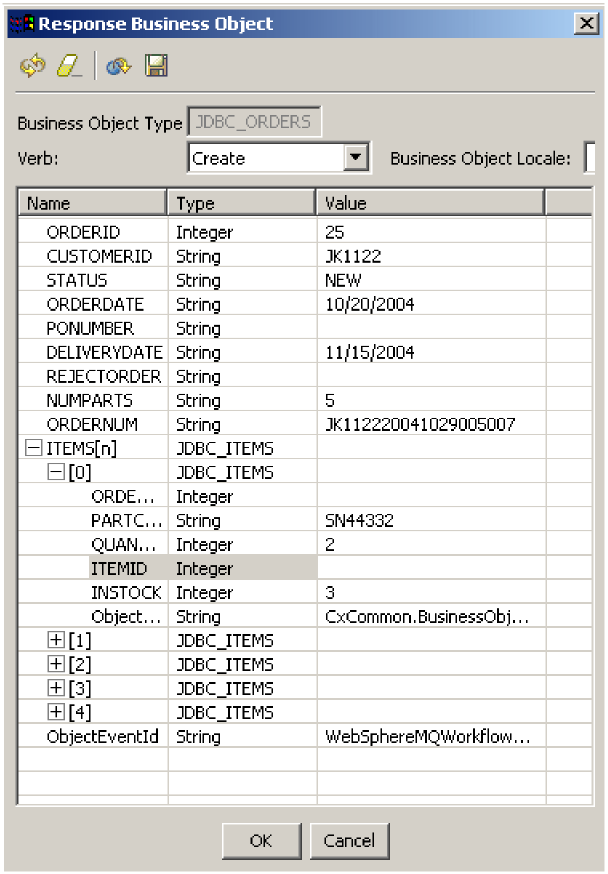
BO Inspector (SalesOrderProcessing_MQWF_to_MQWF)

out.12136.bo | Order (5) | SalesOrderProcessing_MQW

Name	Value	Type
Order	Verb:Create	Order
Locale	en_US	
OrderId	CxIgnore	
OrderNumber	CxIgnore	
OrderDate	10/20/2004	
CustomerId	JK1122	
CustomerAddress[]		Custo
SalesLevel	CxIgnore	
DistributionChannel	CxIgnore	
DivisionLevel	CxIgnore	
SalesRegion	CxIgnore	
SalesBranch	CxIgnore	
AssociatedCustomers[]		Assoc
PONumber	CxIgnore	
PODate	11/15/2004	
OrderContactRef[]		Order
OrderType	CxIgnore	
DocumentReference[]		Docur
OrderStatus	CxIgnore	
OrderPriority	CxIgnore	
OrderReason	CxIgnore	
OrderHdrPaymentTerms	CxIgnore	
PaymentInformation[]		Paym
OrderHdrFreightTerms	CxIgnore	
OrderHdrFreightTermsLocation	CxIgnore	
OrderHdrBillingSchedule	CxIgnore	
CustomerIndustry	CxIgnore	
CustomerPriceGroup	CxIgnore	
CustomerGroup	CxIgnore	

Figure 7-94 Integrated Test Environment BO Inspector View

19. Edit the response business object in the Result Pane of the destination Client Simulator view with the appropriate values (Figure 7-95).



Business Object Type: JDBC_ORDERS
Verb: Create
Business Object Locale:

Name	Type	Value
ORDERID	Integer	25
CUSTOMERID	String	JK1122
STATUS	String	NEW
ORDERDATE	String	10/20/2004
PONUMBER	String	
DELIVERYDATE	String	11/15/2004
REJECTORDER	String	
NUMPARTS	String	5
ORDERNUM	String	JK112220041029005007
⊟ ITEMS[n]	JDBC_ITEMS	
⊟ [0]	JDBC_ITEMS	
ORDE...	Integer	
PARTC...	String	SN44332
QUAN...	Integer	2
ITEMID	Integer	
INSTOCK	Integer	3
Object...	String	CxCommon.BusinessObj...
⊞ [1]	JDBC_ITEMS	
⊞ [2]	JDBC_ITEMS	
⊞ [3]	JDBC_ITEMS	
⊞ [4]	JDBC_ITEMS	
ObjectEventId	String	WebSphereMQWorkflow...

Figure 7-95 Edit response business object

20. Send the business object response as a reply by clicking the business object in the result pane and the Reply Successful button shown in Figure 7-96. Because we were set for synchronous mode, the reply will then be forwarded to the result pane of the client simulator view for the WebSphereMQWorkflowConnector also shown in Figure 7-96.

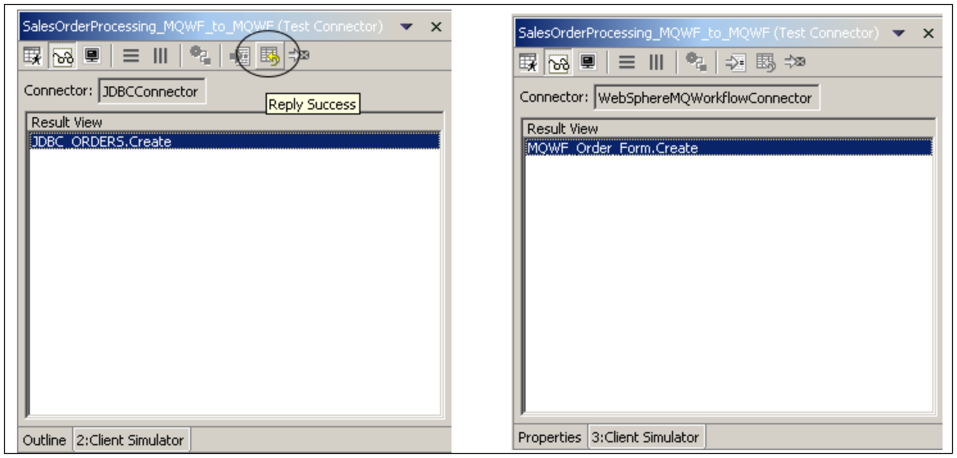


Figure 7-96 Integrated Test Environment reply success

21. Repeat steps 16 through 20 to test this interface again, or repeat steps 6 through 20 to test another interface

7.13.2 Unit testing with the Visual Test Connector

The Visual Test Connector is a utility for simulating a running adapter's connectivity with a target or source application. By using the Visual Test Connector, you can unit test various sections of the integration solution. For testing our scenario, we started all the corresponding InterChange Server components except for the WebSphere MQ Workflow adapter, which we simulated with the Visual Test Connector.

1. To start the test connector, click **Start** → **Programs** → **IBM WebSphere InterChangeServer** → **IBM WebSphere Business Integration Toolset** → **Development** → **Test Connector**.
2. After the Visual Test Connector is started, a new window opens (Figure 7-97 on page 368). In this window, there are three panes:
   – The Supported Business Object pane, in the upper left corner

     This pane is used to create business object instances and send them to the InterChange Server.

– The BO Request List pane, in the upper right corner

  This pane is used to display the business objects that are received by the connector.

– The Output pane, at the bottom

  This is used to display messages about the interaction between the Visual Test Connector and the InterChange Server.

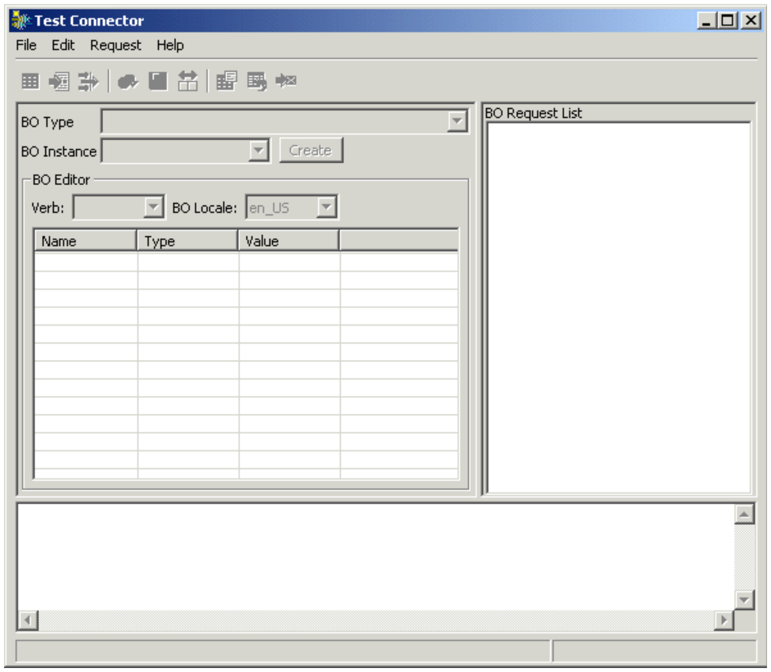


Figure 7-97 Test connector

3. Because the Visual Test Connector can simulate any connector that is installed in the system, it is possible to have different profiles for each adapter that is configured in the system. To create a new profile or to open an existing profile, click **File** → **Create/Select Profile**.

4. Because this is the first time that we have used the Visual Test Connector, no profiles are defined (Figure 7-98) so we must create a new profile. Select **File** → **New Profile** or click the corresponding icon.

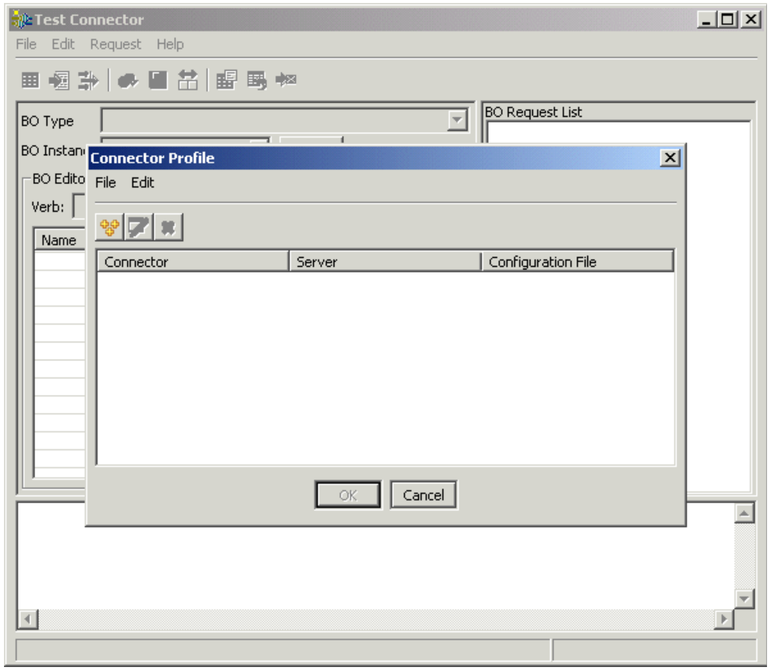


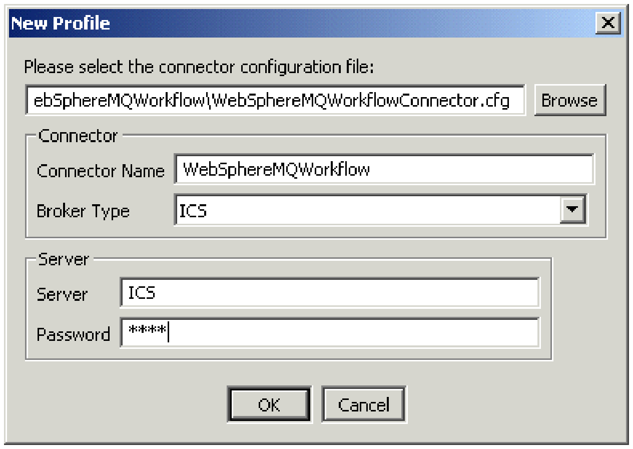
Figure 7-98 Create connector profile

5. In the New Profile window (Figure 7-99 on page 370), specify the following information:
   - The Connector configuration file contains all information that is related to the connector that we want to simulate.
   - The Connector Name *must* be the same name as was specified in the Standard Properties tab. It is also the name that is passed to the connector as a parameter on the command line.
   - The Broker Type is ICS when working with the InterChange Server.

     The following information is needed only if we use the ICS broker type option:

     - Server: InterChange Server name
     - Password: InterChange Server Administrator password
6. In our case, we use the Visual Test Connector to simulate the WebSphere MQ Workflow connector and use the ICS as the Broker Type. Browse the system for the WebSphere MQ Workflow connector configuration file, enter

WebSphereMQWorkflowConnector as the connector name, and enter as the server name ICS and null as the password. Click **OK**.


New Profile

Please select the connector configuration file:

ebSphereMQWorkflow\WebSphereMQWorkflowConnector.cfg

Connector

Connector Name: WebSphereMQWorkflow

Broker Type: ICS

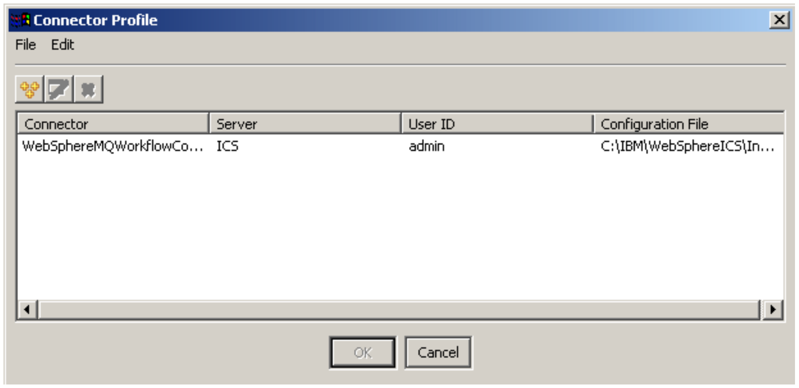
Server

Server: ICS

Password: ****

Figure 7-99 Connector profile properties

7. The profile has been created (Figure 7-100). Select it and click **OK**.


Connector Profile

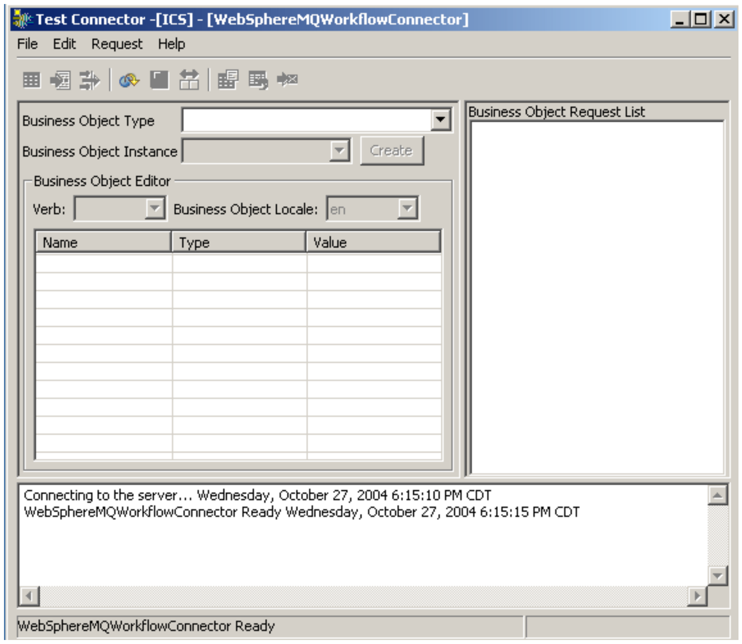
Connector	Server	User ID	Configuration File
WebSphereMQWorkflowCo...	ICS	admin	C:\IBM\WebSphereICS\In...

Figure 7-100 Select the connector profile

8. The next step is to activate the Visual Test Connector. To do this select **File** → **Connect**.

Attention: Before connecting the Test connector (which simulates the WebSphere MQ Workflow connector) to the InterChange Server, make sure that the real connector is not running. If it is, stop it first by typing `q` in the command window.

Figure 7-101 shows in the Output pane the messages indicating the Visual Test Connector. Also, the BOType box has been activated to enable selection of one of the business objects selected by this connector.



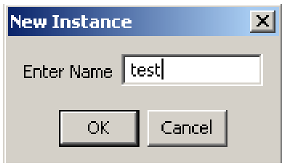
```
Connecting to the server... Wednesday, October 27, 2004 6:15:10 PM CDT
WebSphereMQWorkflowConnector Ready Wednesday, October 27, 2004 6:15:15 PM CDT
```

WebSphereMQWorkflowConnector Ready

Figure 7-101 Test connector is active

9. We want to test the collaboration object we had created. The input application-specific business object to our collaboration object is MQWF_Order_Form; select this as the BOType. Click **Create** to create the application-specific business object.

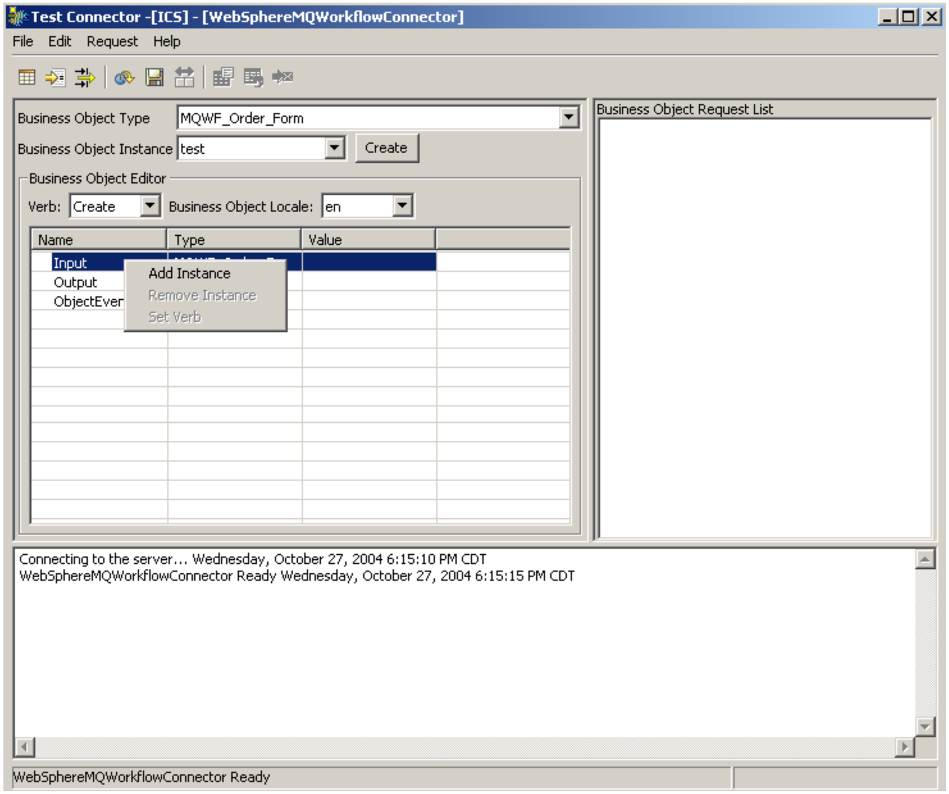
10. Enter the business object name **Test_Message** and click **OK** (Figure 7-102).



Enter Name: test

Figure 7-102 Enter the Business Object name

11.Now the business object top-level structure is displayed (Figure 7-103). To create the child element in our structure, right-click the **Input** element and select **Add instance**.



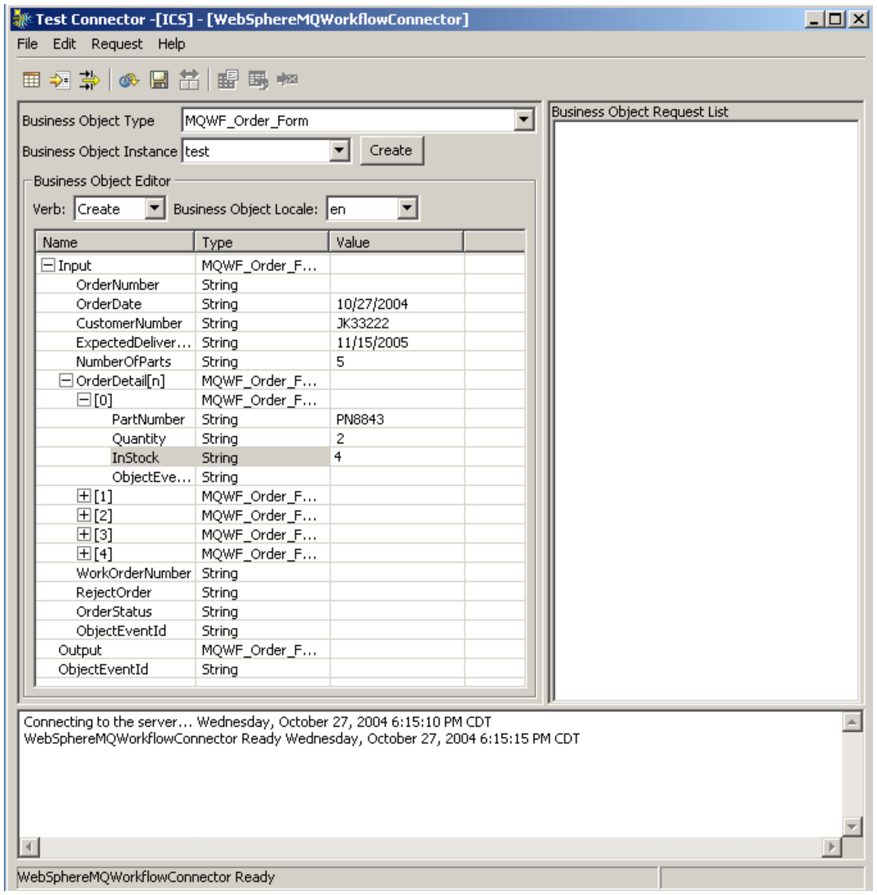
Business Object Type: MQWF_Order_Form
Business Object Instance: test
Verb: Create
Business Object Locale: en

```
Connecting to the server... Wednesday, October 27, 2004 6:15:10 PM CDT
WebSphereMQWorkflowConnector Ready Wednesday, October 27, 2004 6:15:15 PM CDT
```

WebSphereMQWorkflowConnector Ready

Figure 7-103 Add an instance to the child business object

12. Enter data in the business object (Figure 7-104) and create additional instances for the OrderDetail child business object. Specify a different verb if you wish.


Test Connector -[ICS] - [WebSphereMQWorkflowConnector]
File Edit Request Help
Business Object Request List
Create

Business Object Type: MQWF_Order_Form
Business Object Instance: test
Business Object Editor
Verb: Create
Business Object Locale: en

Name	Type	Value
Input	MQWF_Order_F...	
OrderNumber	String	
OrderDate	String	10/27/2004
CustomerNumber	String	JK33222
ExpectedDeliver...	String	11/15/2005
NumberOfParts	String	5
OrderDetail[n]	MQWF_Order_F...	
[0]	MQWF_Order_F...	
PartNumber	String	PN8843
Quantity	String	2
InStock	String	4
ObjectEve...	String	
[1]	MQWF_Order_F...	
[2]	MQWF_Order_F...	
[3]	MQWF_Order_F...	
[4]	MQWF_Order_F...	
WorkOrderNumber	String	
RejectOrder	String	
OrderStatus	String	
ObjectEventId	String	
Output	MQWF_Order_F...	
ObjectEventId	String	

```
Connecting to the server... Wednesday, October 27, 2004 6:15:10 PM CDT
WebSphereMQWorkflowConnector Ready Wednesday, October 27, 2004 6:15:15 PM CDT
```

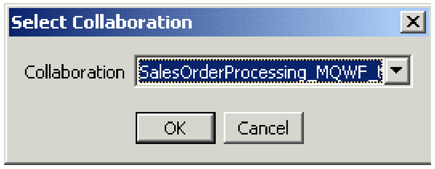
WebSphereMQWorkflowConnector Ready

Figure 7-104 Enter business object values

13. We want to test our collaboration in synchronous mode, so change mode by selecting **Request** → **Mode** → **Synchronous**.

14. To send the business object to the InterChange Server, select **Request** → **Send**.

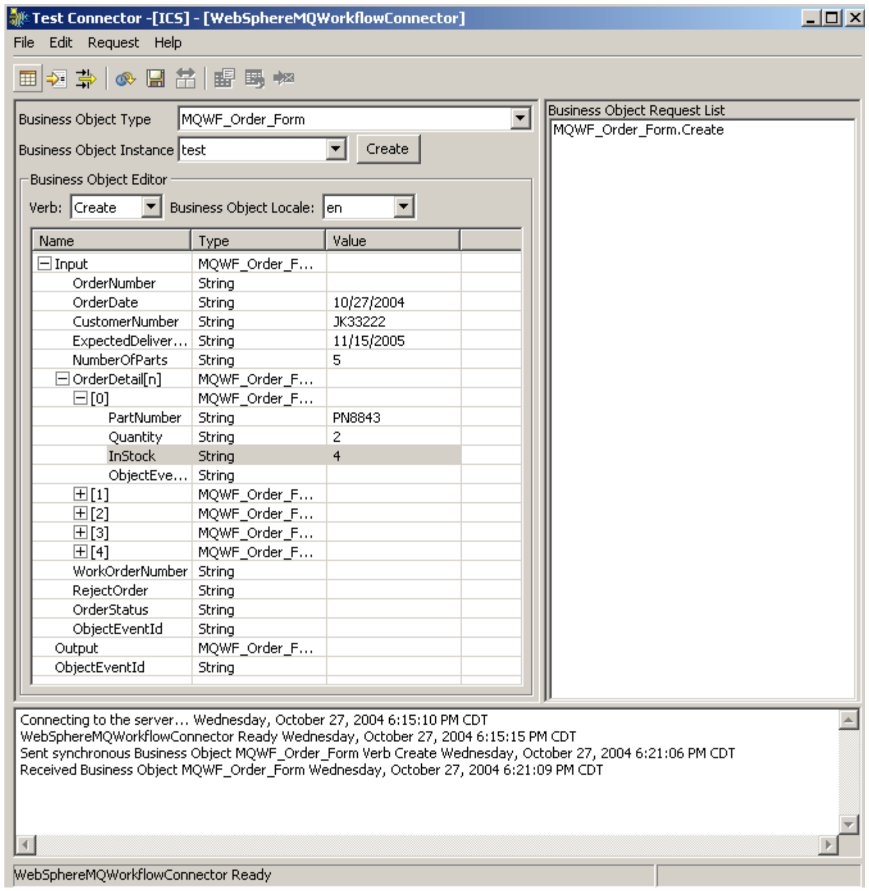
15. Because we are sending the message in a synchronous mode, select the collaboration to start, **SalesOrderProcessing_MQWF_to_MQWF**, and click **OK.** (Figure 7-105).



Collaboration: SalesOrderProcessing_MQWF_t

Figure 7-105 Select the collaboration

Figure 7-106 shows that the business object has been sent and the response has been received.



Test Connector -[ICS] - [WebSphereMQWorkflowConnector]
File
Edit
Request
Help
Create
Business Object Editor
Business Object Request List

Business Object Type: MQWF_Order_Form

Business Object Instance: test

Verb: Create

Business Object Locale: en

Name	Type	Value
Input	MQWF_Order_F...	
OrderNumber	String	
OrderDate	String	10/27/2004
CustomerNumber	String	JK33222
ExpectedDeliver...	String	11/15/2005
NumberOfParts	String	5
OrderDetail[n]	MQWF_Order_F...	
[0]	MQWF_Order_F...	
PartNumber	String	PN8843
Quantity	String	2
InStock	String	4
ObjectEve...	String	
[1]	MQWF_Order_F...	
[2]	MQWF_Order_F...	
[3]	MQWF_Order_F...	
[4]	MQWF_Order_F...	
WorkOrderNumber	String	
RejectOrder	String	
OrderStatus	String	
ObjectEventId	String	
Output	MQWF_Order_F...	
ObjectEventId	String	

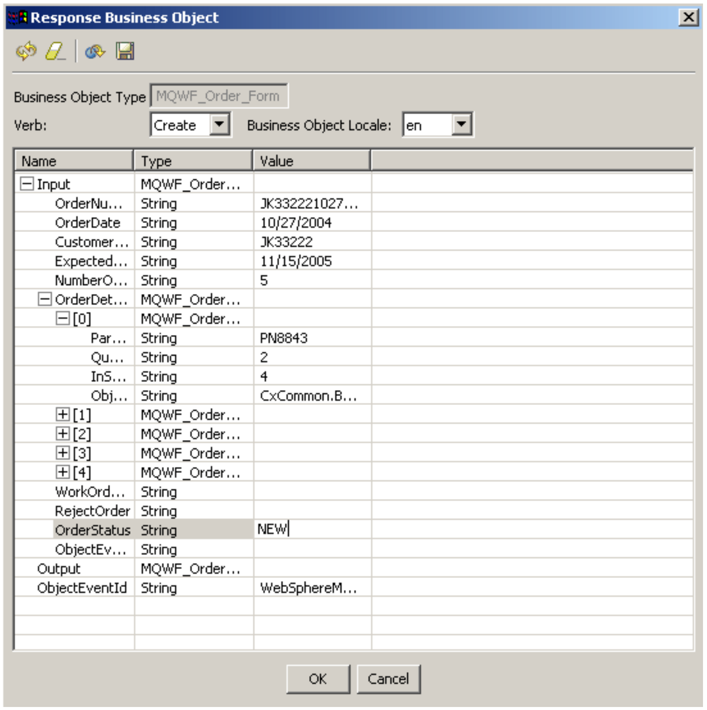
MQWF_Order_Form.Create

```
Connecting to the server... Wednesday, October 27, 2004 6:15:10 PM CDT
WebSphereMQWorkflowConnector Ready Wednesday, October 27, 2004 6:15:15 PM CDT
Sent synchronous Business Object MQWF_Order_Form Verb Create Wednesday, October 27, 2004 6:21:06 PM CDT
Received Business Object MQWF_Order_Form Wednesday, October 27, 2004 6:21:09 PM CDT
```

WebSphereMQWorkflowConnector Ready

Figure 7-106 Successfully sent the message

16. To see the response business object, double-click the business object that was created in the BORequest pane, as shown in Figure 7-107.



Business Object Type: MQWF_Order_Form
Verb: Create
Business Object Locale: en

Name	Type	Value
⊟ Input	MQWF_Order...	
OrderNu...	String	JK332221027...
OrderDate	String	10/27/2004
Customer...	String	JK33222
Expected...	String	11/15/2005
NumberO...	String	5
⊟ OrderDet...	MQWF_Order...	
⊟ [0]	MQWF_Order...	
Par...	String	PN8843
Qu...	String	2
InS...	String	4
Obj...	String	CxCommon.B...
⊞ [1]	MQWF_Order...	
⊞ [2]	MQWF_Order...	
⊞ [3]	MQWF_Order...	
⊞ [4]	MQWF_Order...	
WorkOrd...	String	
RejectOrder	String	
OrderStatus	String	NEW
ObjectEv...	String	
Output	MQWF_Order...	
ObjectEventId	String	WebSphereM...

Figure 7-107 Response Business Object

7.13.3 End-to-end testing using the Web Client

We can now stop the Virtual Test connector and restart the real WebSphere MQ Workflow connector, allowing real testing between the WebSphere MQ Workflow server and the InterChange Server.

This time, when using the WebSphere MQ Workflow Web Client to create a new process instance, the workflow XML message is picked up by the WebSphere MQ Workflow adapter where it is parsed into a business object and delivered to the collaboration. When the JDBC adapter gets the new Order information, the records are inserted into the database and the new order number is passed back to the collaboration, the WebSphere MQ Workflow adapter, and, eventually, to the WebSphere MQ Workflow server. This completes the Create Sales Order activity (Figure 7-108 on page 376), and the MQ Workflow runtime engine executes the Order Parts activity.

The process monitor shows us now that the activity Order Parts is running, which means that the WebSphere MQ Workflow server has sent an XML message to the WebSphere Business Integration Message Broker, which provides the implementation of that activity. The implementation of that activity is the subject of the next chapter.

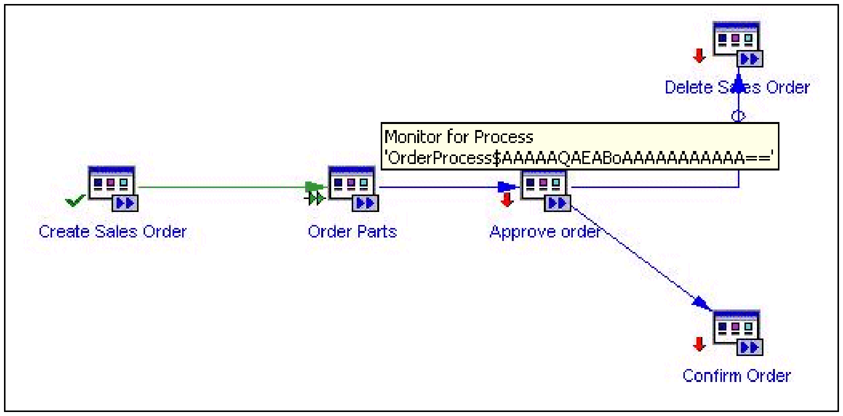


Figure 7-108 Completed Sales Order Activity

When the Message Broker has passed the return data back to the running workflow, the user completes the manual step (Approve Order) of approving or rejecting the order. If the order is approved, the Confirm Order activity is invoked, which will send a message to the InterChange Server to update the status of the order from NEW to Approved within the database.

If the user rejected the sales order the Delete Sales Order activity will be invoked that will send a message to the InterChange server with a status that informs the InterChange Server to perform a soft delete of the previously entered order.

8

Replenishing parts in WebSphere BI Message Broker

This chapter describes the development and deployment of message flows that implement the part replenishment section of the business process. The actual logic of the message flows is relatively simple, because we focus more on the integration between the runtime components of WebSphere Business Integration Server and implementation techniques.

8.1 Overview

Now that the order is created in the order management system, you can order the individual parts.

The purpose of the message flow is to accept the order message from the workflow and then to split out every part of the order into a separate message. These separate messages flow to another flow the simulates an external application sending order response messages.

To do this, the message flow uses the aggregation nodes in WebSphere Business Integration Message Broker. The message flow waits for all of the answers to return from the simulation flow and combines all answers into one message, formatted according to WebSphere MQ Workflow rules.

This reply message flows back to WebSphere MQ Workflow Runtime where the activity completes and the process flow continues to the next activity.

The aggregation is done by two message flows: fan-out and fan-in. *Fan-out* picks up the message from workflow and splits it into a separate message for every part in the order. *Fan-in* retrieves the response messages and, as soon as all responses have arrived, builds the combined response.

8.2 Implementation steps

Before we start the development of these message flows, we first make sure that workflow XML messages arrive at the broker and that the broker can send responses back to the workflow server. Several MQ objects must be created to achieve this state.

We can then develop the fan-in and fan-out flow using Aggregate node. We also develop the response flow, which calculates a random price for committed price and invoiced price, date on expected date, shipment date, reception date and registration date, case number information and other information.

After we have deployed the flows to the broker, we can resume the business process and test the message flows.

8.3 WebSphere MQ configuration

For the broker to be able to receive WebSphere MQ Workflow XML messages, it must have MQ communication with the FMCQM queue manager associated with the WebSphere MQ Workflow server. Because the FMCQM queue manager is, by default, part of a cluster, it is sufficient that the broker queue manager joins this cluster. This can be accomplished by defining a cluster receiver channel and a cluster sender channel. The role of the cluster receiver channel is to publish to the cluster the way in which other queue managers in the cluster should communicate with it. The role of the cluster sender channel is to enable communication with a full repository queue manager in the cluster. The FMCQM queue manager of the WebSphere MQ Workflow server is by default a full repository queue manager.

1. The following commands are to be executed in a **runmqsc** session for the BKQM queue manager.

   This command creates a cluster receiver channel:

   ```
   def chl(TO.BKQM.TCP) chltype(clusrcvr) conname(wbimb) trptype(tcp)
   cluster(FMCGRP)
   ```

   This command creates a cluster sender channel, connecting the BKQM queue manager to the FMCQM queue manager:

   ```
   def chl(TO.FMCQM.TCP) chltype(clussdr) conname('wbiwf(5010)') trptype(tcp)
   cluster(FMCGRP)
   ```

2. The next step is to define the queue toward which the WebSphere MQ Workflow server will send the XML message. The name of this queue was set in the UPES definition in WebSphere MQ Workflow. The following command defines this queue and makes it part of the cluster. This means that the WebSphere MQ Workflow queue manager FMCQM will know about it as soon as it is defined.

   ```
   def ql(ORDER.INPUT) cluster(FMCGRP)
   ```

3. The XML response message for the WebSphere MQ Workflow server must be sent to a predefined queue (FMC.FMCGRP.EXE.XML) on the FMCQM queue manager. Because this queue is by default part of the cluster, no special definitions have to be made on the BKQM queue manager.
4. The XML request message is broken into several submessages, one for each part that must be ordered. Each message for a single part is sent to an ORDER.REQUEST queue. To define this queue, use a command such as:

   ```
   def ql(ORDER.REQUEST)
   ```

   Note that the ORDER.REQUEST queue is not part of the cluster. It is not visible to other queue managers in the cluster.

The ORDER.REQUEST queue could be the input queue for some business-to-business product that forwards the order message to the supplier. In our simplified scenario, we simulate the supplier interaction via another message flow. This message flow's input queue is the ORDER.REQUEST queue. This message flow builds a response message for each order request for a single part. This response message is sent to the ORDER.REQUEST queue, which you define with this command:

```
def ql(ORDER.RESPONSE)
```

The ORDER.REQUEST queue acts as the input queue for the fan-in message flow where all response messages are aggregated into a single response for the WebSphere MQ Workflow server. The fan-in and the fan-out message flows work together by exchanging a control message via another queue, called ORDER.CONTROL. Define this queue with the following command:

```
def ql(ORDER.CONTROL)
```

One last queue is required to pass the original workflow request message to the fan-in flow. Usually, this can be accomplished by passing along the required context information for workflow in the so-called LocalEnvironment. However, when the fan-in and fan-out flows are detached, the LocalEnvironment is not preserved and other techniques are required. For the time being, accept that another queue is required. Define this queue using the following command:

```
def ql(ORDER.SAVE)
```

This completes the required setup for the message broker queue manager.

8.4 Implementation of the fan-out and fan-in flow

Example 8-1 shows a sample WebSphere MQ Workflow request message as it will be received by the fan-out flow in the message broker.

Example 8-1 Sample workflow input XML message to message broker

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<WfMessage>
 <WfMessageHeader>
  <ResponseRequired>Yes</ResponseRequired>
 </WfMessageHeader>
 <ActivityImplInvoke>

<ActImplCorrelID>RUEAAAABACrAGwAAAAAAAAAAAAAABQAAAAEALEAAAAAAAAAAAAAAAAFQQAAAAEAKsAdAAAAAAAAAA
BF</ActImplCorrelID>
  <Starter>ADMIN</Starter>
  <ProgramID>
   <ProcTemplID>AAAAAQAsQAAAAAAAAAAAAA==</ProcTemplID>
   <ProgramName>Dummy Unattended</ProgramName>
  </ProgramID>
  <ImplementationData>
   <ImplementationPlatform>WindowsNT</ImplementationPlatform>
   <ExeOptions>
    <PathAndFileName>DummyUnattended.cmd</PathAndFileName>
    <InheritEnvironment>true</InheritEnvironment>
    <StartInForeGround>true</StartInForeGround>
    <WindowStyle>Visible</WindowStyle>
   </ExeOptions>
  </ImplementationData>
  <ProgramInputData>
   <_ACTIVITY>Order Parts</_ACTIVITY>
   <_PROCESS>OrderProcess$AAAAAQAqwBsAAAAAAAAAAA==</_PROCESS>
   <_PROCESS_MODEL>OrderProcess</_PROCESS_MODEL>
   <Parts_Replenishment_Form>
    <WorkOrderNumber>552345</WorkOrderNumber>
    <PartsForm>
     <NumberOfParts>5</NumberOfParts>
     <PartList>
      <PartOrder>
       <PartNumber>CPU</PartNumber>
       <Quantity>5</Quantity>
       <InStock>7</InStock>
      </PartOrder>
     </PartList>
     <PartList>
      <PartOrder>
       <PartNumber>Monitor</PartNumber>
       <Quantity>3</Quantity>
```

```
        <InStock>5</InStock>
       </PartOrder>
      </PartList>
      <PartList>
       <PartOrder>
        <PartNumber>Keyboard</PartNumber>
        <Quantity>3</Quantity>
        <InStock>9</InStock>
       </PartOrder>
      </PartList>
      <PartList>
       <PartOrder>
        <PartNumber>Power adapter</PartNumber>
        <Quantity>4</Quantity>
        <InStock>9</InStock>
       </PartOrder>
      </PartList>
      <PartList>
       <PartOrder>
        <PartNumber>USB drive</PartNumber>
        <Quantity>3</Quantity>
        <InStock>6</InStock>
       </PartOrder>
      </PartList>
     </PartsForm>
    </Parts_Replenishment_Form>
   </ProgramInputData>
   <ProgramOutputDataDefaults>
    <_ACTIVITY>Order Parts</_ACTIVITY>
    <_PROCESS>OrderProcess$AAAAAQAqwBsAAAAAAAAAAA==</_PROCESS>
    <_PROCESS_MODEL>OrderProcess</_PROCESS_MODEL>
    <Parts_Replenishment_Form>       </Parts_Replenishment_Form>
   </ProgramOutputDataDefaults>
  </ActivityImplInvoke>
</WfMessage>
```

The fan-out message flow picks up the message from the ORDER.INPUT queue and generates one or more order request messages on the ORDER.REQUEST queue, depending on the NumberofParts information set at Workflow server. Example 8-2 shows a sample order request message that this flow has to build.

Example 8-2 Sample order request message

```
<Part_Replenishment>
 <WorkOrderNumber>WO3483</WorkOrderNumber>
 <PartNumber>PartNumber1</PartNumber>
 <Quantity>2</Quantity>
 <InStock>5</InStock>
```

```
</Part_Replenishment>
```

The sample output message from message broker to Workflow server is shown in Example 8-3.

Example 8-3 Sample output message to Workflow

```
<WfMessage>
 <WfMessageHeader>
  <ResponseRequired>No</ResponseRequired>
 </WfMessageHeader>
 <ActivityImplInvokeResponse>

<ActImplCorrelID>RUEAAAABACrAGwAAAAAAAAAAAAAABQAAAAEALEAAAAAAAAAAAAAAAAAFQQAAAAEAKsAdAAAAAAAAAA
BF</ActImplCorrelID>
  <ProgramRC>0</ProgramRC>
  <ProgramOutputData>
   <Parts_Replenishment_Form>
    <PartsForm>
     <NumberOfParts>5</NumberOfParts>
     <PartList>
      <PartOrder>
       <PartNumber>CPU</PartNumber>
       <Quantity>5</Quantity>
       <InStock>7</InStock>
      </PartOrder>
      <Supplier>IBM ITSO</Supplier>
      <CommittedPrice>1.64E+0</CommittedPrice>
      <OrderReference>2004-10-26 10:30:34.585</OrderReference>
      <ExpectedDate>2004-11-02</ExpectedDate>
      <ShipmentDate>2004-11-09</ShipmentDate>
      <CaseNumber>16</CaseNumber>
      <ReceptionDate>2004-10-28</ReceptionDate>
      <RegistrationDate>2004-10-29</RegistrationDate>
      <InvoicedPrice>8.2E-1</InvoicedPrice>
     </PartList>
     <PartList>
      <PartOrder>
       <PartNumber>Monitor</PartNumber>
       <Quantity>3</Quantity>
       <InStock>5</InStock>
      </PartOrder>
      <Supplier>IBM ITSO</Supplier>
      <CommittedPrice>1.46E+0</CommittedPrice>
      <OrderReference>2004-10-26 10:30:34.605</OrderReference>
      <ExpectedDate>2004-11-02</ExpectedDate>
      <ShipmentDate>2004-11-09</ShipmentDate>
      <CaseNumber>14</CaseNumber>
      <ReceptionDate>2004-10-28</ReceptionDate>
      <RegistrationDate>2004-10-29</RegistrationDate>
      <InvoicedPrice>7.3E-1</InvoicedPrice>
     </PartList>
     <PartList>
      <PartOrder>
       <PartNumber>Keyboard</PartNumber>
       <Quantity>3</Quantity>
       <InStock>9</InStock>
      </PartOrder>
      <Supplier>IBM ITSO</Supplier>
      <CommittedPrice>1.46E+0</CommittedPrice>
      <OrderReference>2004-10-26 10:30:34.635</OrderReference>
      <ExpectedDate>2004-11-02</ExpectedDate>
      <ShipmentDate>2004-11-09</ShipmentDate>
      <CaseNumber>14</CaseNumber>
      <ReceptionDate>2004-10-28</ReceptionDate>
      <RegistrationDate>2004-10-29</RegistrationDate>
      <InvoicedPrice>7.3E-1</InvoicedPrice>
     </PartList>
     <PartList>
      <PartOrder>
       <PartNumber>Power adapter</PartNumber>
       <Quantity>4</Quantity>
       <InStock>9</InStock>
      </PartOrder>
      <Supplier>IBM ITSO</Supplier>
      <CommittedPrice>1.55E+0</CommittedPrice>
      <OrderReference>2004-10-26 10:30:34.645</OrderReference>
      <ExpectedDate>2004-11-02</ExpectedDate>
      <ShipmentDate>2004-11-09</ShipmentDate>
      <CaseNumber>15</CaseNumber>
      <ReceptionDate>2004-10-28</ReceptionDate>
      <RegistrationDate>2004-10-29</RegistrationDate>
      <InvoicedPrice>7.7E-1</InvoicedPrice>
     </PartList>
     <PartList>
      <PartOrder>
       <PartNumber>USB drive</PartNumber>
       <Quantity>3</Quantity>
       <InStock>6</InStock>
      </PartOrder>
      <Supplier>IBM ITSO</Supplier>
      <CommittedPrice>1.46E+0</CommittedPrice>
      <OrderReference>2004-10-26 10:30:34.665</OrderReference>
      <ExpectedDate>2004-11-02</ExpectedDate>
      <ShipmentDate>2004-11-09</ShipmentDate>
      <CaseNumber>14</CaseNumber>
      <ReceptionDate>2004-10-28</ReceptionDate>
```

```
      <RegistrationDate>2004-10-29</RegistrationDate>
      <InvoicedPrice>7.3E-1</InvoicedPrice>
     </PartList>
    </PartsForm>
    <ReturnCode>0</ReturnCode>
   </Parts_Replenishment_Form>
  </ProgramOutputData>
 </ActivityImplInvokeResponse>
</WfMessage>
```

The following steps guide you through building such a message flow in the Message Broker Toolkit of WebSphere Business Integration Message Broker V5. (The implementation of the toolkit is discussed in Chapter 4, "Implementing client components" on page 119.)

1. As with all solution components that are developed in an Eclipse-based development tool, solution components are grouped in a project. To create a project to hold message flows, in the Broker Development perspective of the Broker Toolkit, select **File** → **New** → **Message Flow Project**. Name this project `Parts_Replenishment_Project`, as shown in Figure 8-1.

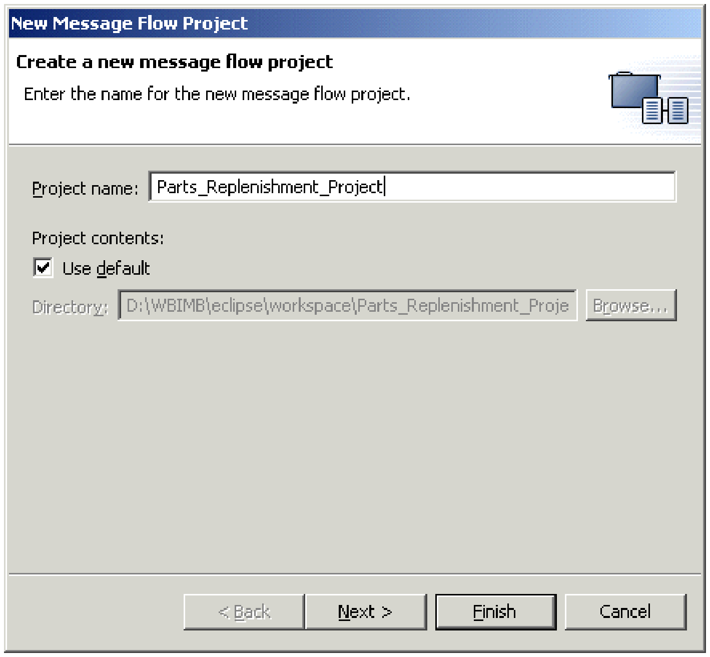
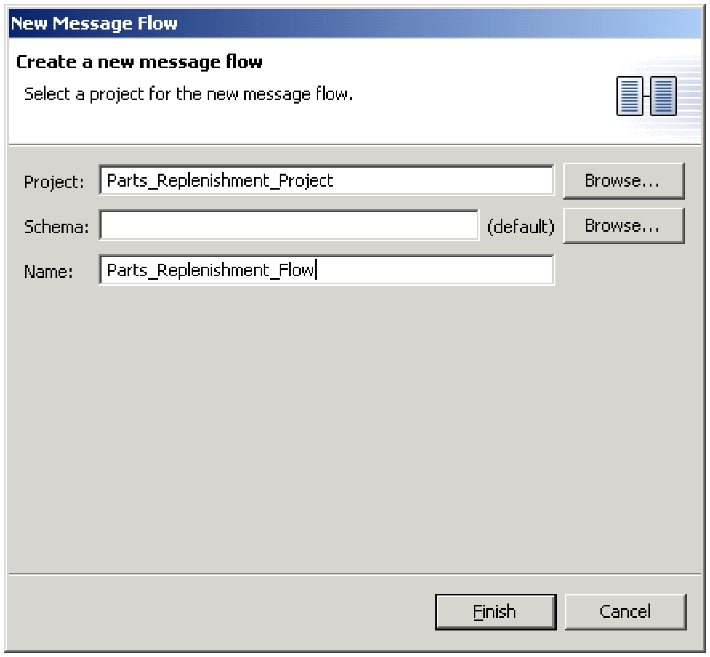

New Message Flow Project
Create a new message flow project
Enter the name for the new message flow project.
Project name: Parts_Replenishment_Project
Project contents:
Use default
Directory: D:\WBIMB\eclipse\workspace\Parts_Replenishment_Proje
Browse...
< Back
Next >
Finish
Cancel

Figure 8-1 Create a message flow project

2. Next, we create a message flow. Select **File** → **New** → **Message Flow**. If the cursor was not positioned on the message flow project, you might have to provide its name (Figure 8-2). You can do this by clicking the **Browse** button or by typing the name of the project. Name the new flow **Parts_Replenishment_Flow**.



Create a new message flow

Select a project for the new message flow.

Project: Parts_Replenishment_Project

Schema: (default)

Name: Parts_Replenishment_Flow

Figure 8-2 Create new message flow

8.4.1 Building the graphical flow

The message flow editor is now open. A palette of existing nodes enables you to add to the message flow editor to build the required logic. Figure 8-3 on page 387 shows the completed message flow, consisting of the fan-out flow (top flow in the figure) and the fan-in flow (bottom flow).

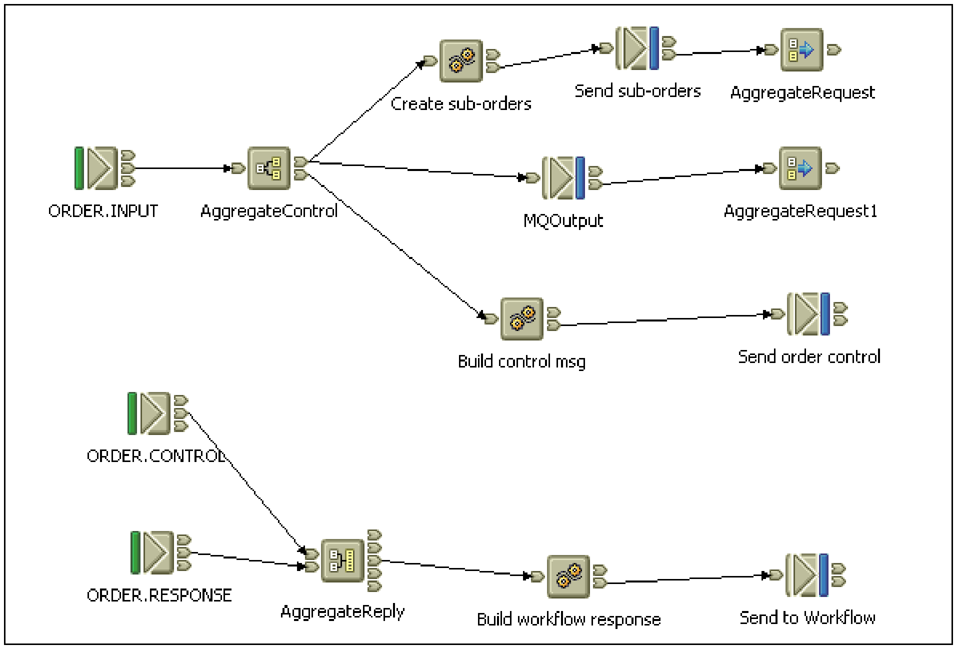


Figure 8-3 Completed fan-in and fan-out flow

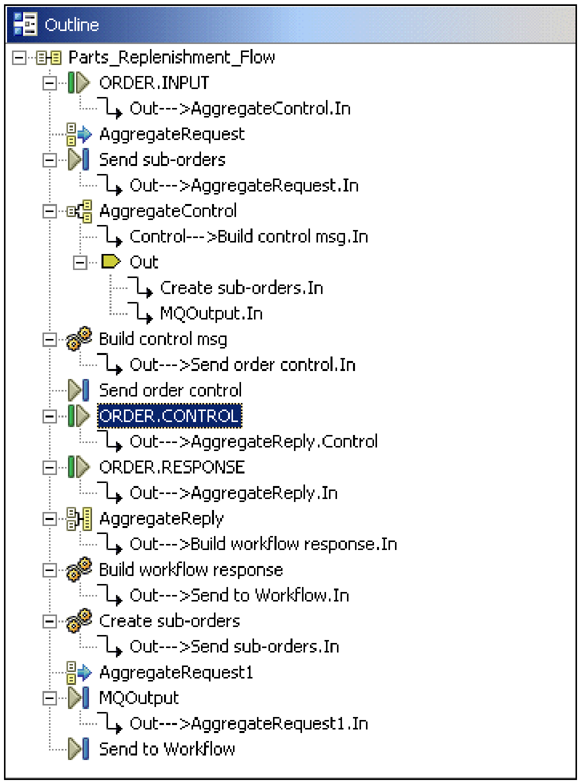
The fan-out flow consists of the following nodes, from left to right:

- An MQInput node, renamed as ORDER.INPUT
- An AggregateControl node
- A Compute node, renamed as Create sub-orders
- An MQOutput node, renamed as Send sub-orders
- An AggregateRequest node
- A Compute node, renamed as Build control msg
- An MQOutput node, renamed as Send order control

The fan-in flow consists of the following nodes, from left to right:

- An MQInput node, renamed as ORDER.CONTROL
- An MQInput node, renamed as ORDER.RESPONSE
- An AggregateReply node
- A Compute node, renamed as Build workflow response
- An MQReply node, renamed as Send to Workflow

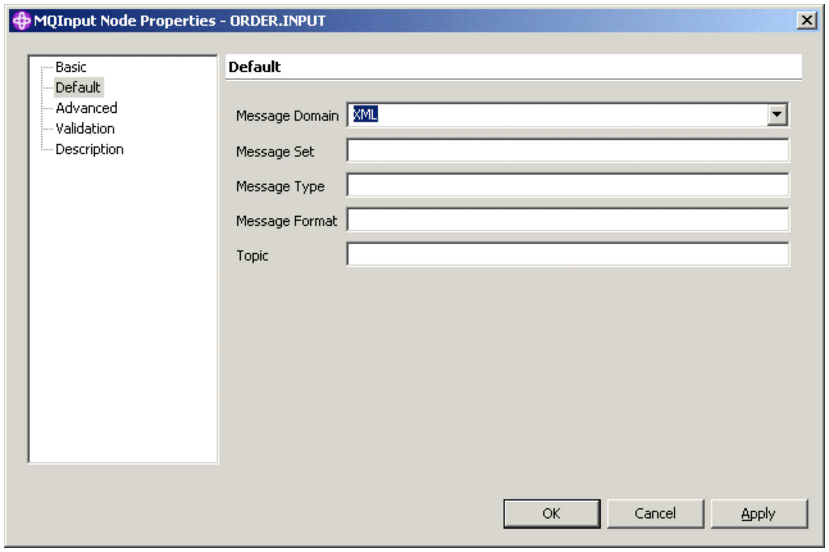
Figure 8-3 shows all of the nodes connected to each other. The Outline view in Figure 8-4 on page 388 gives another view of how they connect to each other. The selected node in the figure marks the start of the fan-in flow. The nodes above it are part of the fan-out flow.



```
Outline
Parts_Replenishment_Flow
  ORDER.INPUT
    Out--->AggregateControl.In
  AggregateRequest
  Send sub-orders
    Out--->AggregateRequest.In
  AggregateControl
    Control--->Build control msg.In
    Out
      Create sub-orders.In
      MQOutput.In
  Build control msg
    Out--->Send order control.In
  Send order control
  ORDER.CONTROL
    Out--->AggregateReply.Control
  ORDER.RESPONSE
    Out--->AggregateReply.In
  AggregateReply
    Out--->Build workflow response.In
  Build workflow response
    Out--->Send to Workflow.In
  Create sub-orders
    Out--->Send sub-orders.In
  AggregateRequest1
  MQOutput
    Out--->AggregateRequest1.In
  Send to Workflow
```

Figure 8-4 Outline view for the message flows

Now we look at the properties of each node. Starting with the MQInput node ORDER.INPUT of the fan-in flow, the Basic section of this node's properties refer to the MQ queue ORDER.INPUT. In the Default section, shown in Figure 8-5, we select **XML** as the message domain.


MQInput Node Properties - ORDER.INPUT

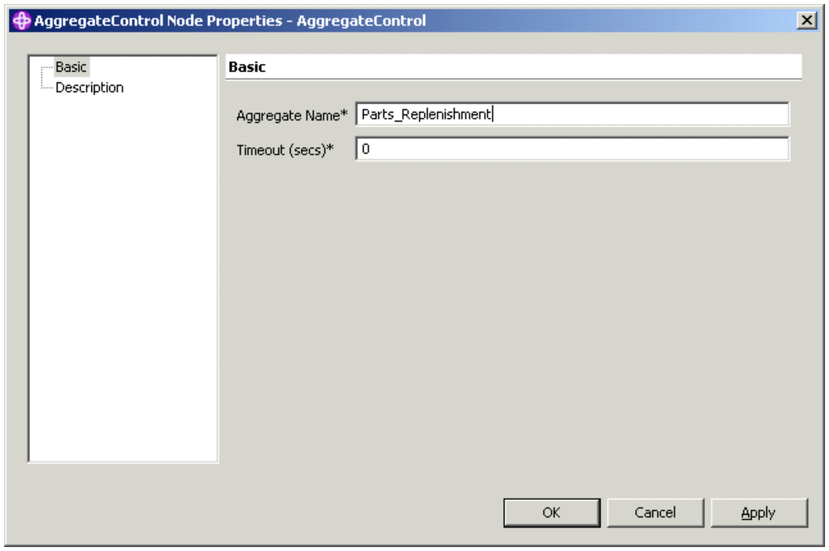
Default

Message Domain: XML
Message Set:
Message Type:
Message Format:
Topic:

Figure 8-5 Default properties of the MQInput node ORDER.INPUT

The node following the MQInput node, an *AggegrateControl* node, initializes the aggregation control logic within the flow. It has two properties: a time-out value and the name of the aggregation.

If a broker has multiple flows that use the aggregation nodes, then each of these flows has an AggregationControl node. Each of these nodes should have a different name. For our scenario, we used `Parts_Replenishment`.


AggregateControl Node Properties - AggregateControl

Basic

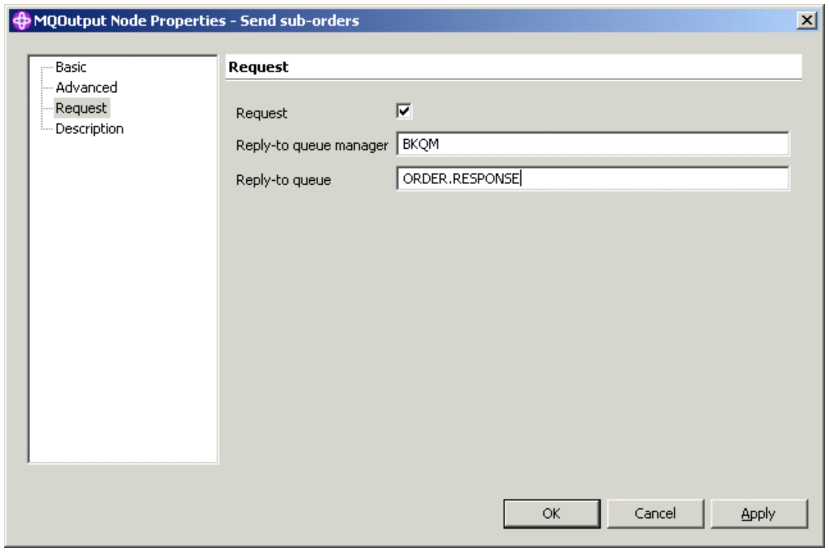
Aggregate Name* Parts_Replenishment

Timeout (secs)* 0

Figure 8-6 Properties of the AggegrateControl node

We skip the Compute nodes Create suborders and Build control msg for now.

The next node is the MQOutput node named Send sub-orders. In the Basic section of this node's properties, we set the name of the ORDER.REQUEST queue and the BKQM queue manager. In the Request section, shown in Figure 8-7, we select the option that the broker should build a **Request** message, and we provide a reply-to queue manager and reply-to queue. The reply-to queue, of course, is the input queue for the fan-in flow.



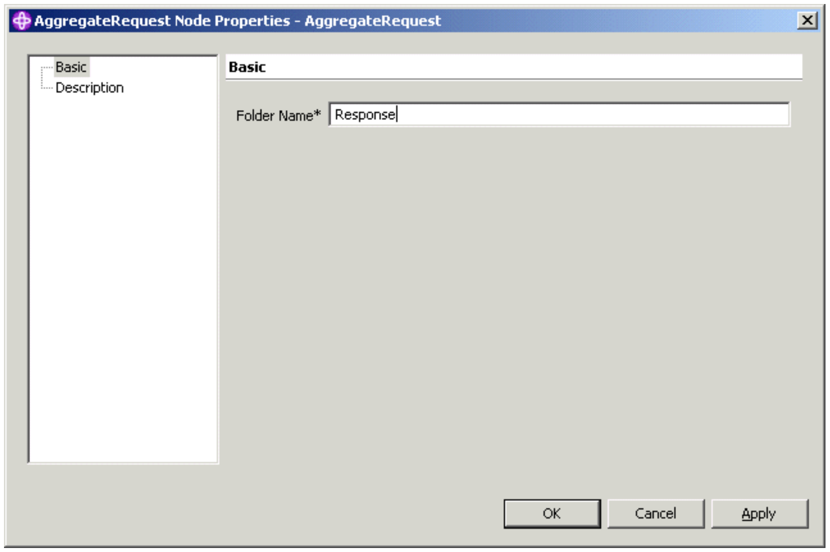
Request ✓

Reply-to queue manager: BKQM

Reply-to queue: ORDER.RESPONSE

Figure 8-7 Request properties of the MQOutput node Send sub-orders

The last node in that part of the fan-out flow is the *AggregateRequest* node. This node should always be at the end of a sequence of nodes that results in a request message that is part of an aggregation. There is only one important property for this node: the Folder Name. This is the name within the message tree that contains the response message.



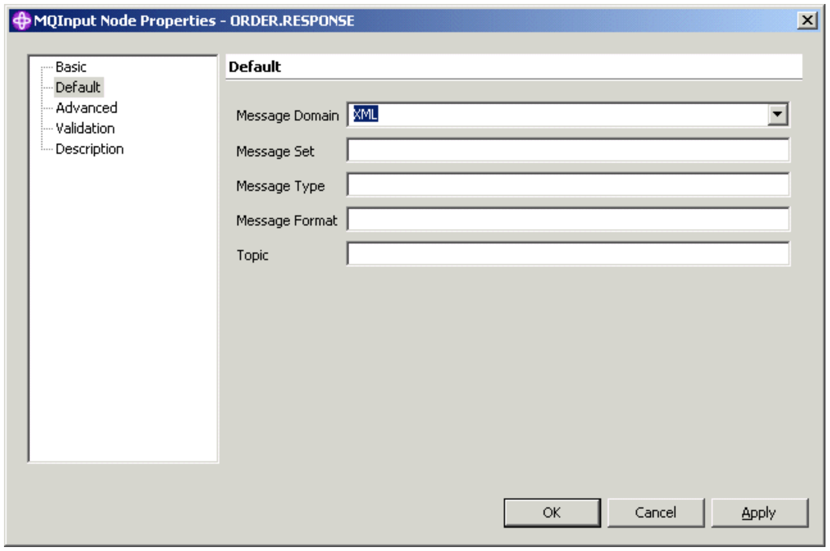
Folder Name* Response

Figure 8-8 Properties of the AggregateRequest node

The properties of the MQOutput node named Send control msg has the ORDER.CONTROL queue as its target queue. No other properties are being used by this node.

The two nodes named MQOutput and AggregateRequest1 are used to make sure that the fan-in flow has access to the workflow context. In the workflow request message, several elements are provided that must be part of the response message to workflow. However, the subflow that builds this workflow response message does not have access to this information. It only has access to the information that is being received from the suppliers. There are several ways to get around that problem. One way is to store that information in a table and then retrieve it when the responses from the suppliers have arrived. Another elegant way is to send out a dummy supplier request message and make that dummy message part of the aggregation logic. This is being done by the MQOutput and AggregateRequest1 nodes. The MQOutput node is configured similar to the Send suborders node. The only difference is that it is uses the ORDER.SAVE queue. It also has the same settings for the request processing. The AggregateRequest1 node uses a different folder name: Request. Thus, when the aggregation is complete, we have a Request folder with the original workflow message in it and a Response folder with an array of responses from the suppliers.

We now look at the fan-in flow. The MQInput node named *ORDER.RESPONSE* has the ORDER.RESPONSE queue in the Basic section of its properties. In the Default section, we select **XML** as the message domain.



MQInput Node Properties - ORDER.RESPONSE

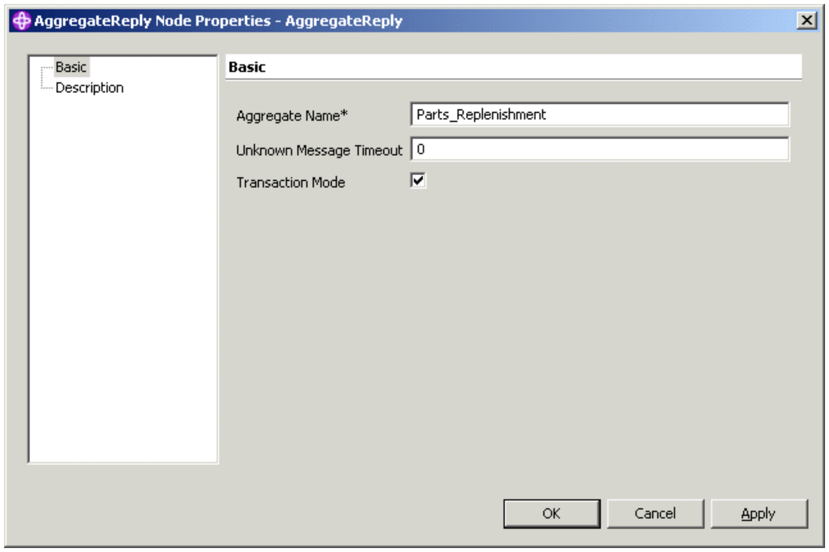
Default

Message Domain: XML
Message Set:
Message Type:
Message Format:
Topic:

Figure 8-9 Default properties of the MQInput node ORDER.RESPONSE

The MQInput node ORDER.CONTROL is similar, with the ORDER.CONTROL queue in the Basic section of its properties. It also has **XML** as message domain.

The next node is the *AggregateReply* node, which has as its task the actual aggregation of messages. This node has to know what responses to expect. All responses should belong to the same Aggregate Name: `Parts_Replenishment`.



Aggregate Name*: Parts_Replenishment
Unknown Message Timeout: 0
Transaction Mode: ✓

Figure 8-10 Properties of the Aggregate Reply node

We again skip the Compute node Build workflow message. The last node is the MQOutput node *Send reply to workflow*, which sends the reply message to the FMC.FMCGRP.EXE.XML queue of the WebSphere MQ Workflow queue manager FMCQM.

8.4.2 Developing ESQL in the Compute nodes

In the previous section we skipped the discussion about the Compute nodes to focus on the visual programming logic. In this section, we discuss the ESQL that sits behind each Compute node.

Compute node Create suborders

The Compute node *Create suborders* breaks the input message into one message for every part in the original order. (See the ESQL in Example 8-4.)

In the `Main()` function of this ESQL module, we first declare a reference to a section of the message tree. (This is an easy way to save on typing and avoid typing errors.) Then, the loop counter is initialized by retrieving the value of the `NumberOfParts` element. Note that you cannot use anything like CARDINALITY for the array PartList, because the WebSphere MQ Workflow server will also send empty elements.

Within the `WHILE` loop, we build an output message for every pass through the loop. Thus, we first copy all the message headers from the full request message, then copy those elements from the message that are required for the supplier of the parts. When the message is built, we use the PROPAGATE function, which sends the current OutputRoot tree to the out terminal, which is connected to the MQOutput node Send suborders. Then, the loop counter is incremented and the loop is repeated if the loop condition still evaluates to true.

When we get out of the loop, we end the Main function by returning FALSE. Usually, you end the ESQL for a Compute node by returning TRUE. But here it is important to return FALSE so that the Compute node does not try to propagate another message. At the end of the loop, the OutputRoot is an empty message tree. Returning TRUE results in forwarding an empty message tree to the MQOutput node and thus in an error.

Example 8-4 ESQL module for Compute node Create suborders

```
CREATE COMPUTE MODULE Parts_Replenishment_Flow_Compute
   CREATE FUNCTION Main() RETURNS BOOLEAN
   BEGIN
      DECLARE refInput REFERENCE TO
         InputRoot.XML.WfMessage.ActivityImplInvoke.ProgramInputData.Parts_Replenishment_Form;
      DECLARE i, partsNumber INTEGER;
      SET i = 1;
      SET partsNumber = refInput.PartsForm.NumberOfParts;

      WHILE i <= partsNumber DO
         CALL CopyMessageHeaders();
         SET OutputRoot.XML.Part_Replenishment.WorkOrderNumber = refInput.WorkOrderNumber;
         SET OutputRoot.XML.Part_Replenishment.PartNumber =
refInput.PartsForm.PartList[i].PartOrder.PartNumber;
         SET OutputRoot.XML.Part_Replenishment.Quantity =
refInput.PartsForm.PartList[i].PartOrder.Quantity;
         SET OutputRoot.XML.Part_Replenishment.InStock =
refInput.PartsForm.PartList[i].PartOrder.InStock;
          SET OutputRoot.XML.Part_Replenishment.InvoicedPrice =
RAND(CAST(refInput.PartsForm.PartList[i].PartOrder.Quantity AS INTEGER)) * 1000;
         SET OutputRoot.XML.Part_Replenishment.CommittedPrice =
RAND(CAST(refInput.PartsForm.PartList[i].PartOrder.Quantityy AS INTEGER)) * 1000;


         PROPAGATE;
         SET i = i + 1;
      END WHILE;
      RETURN FALSE;
   END;

   CREATE PROCEDURE CopyMessageHeaders() BEGIN
      DECLARE I INTEGER 1;
      DECLARE J INTEGER CARDINALITY(InputRoot.*[]);
      WHILE I < J DO
         SET OutputRoot.*[I] = InputRoot.*[I];
         SET I = I + 1;
      END WHILE;
   END;

END MODULE;
```

Compute node Build control msg

Example 8-5 shows the Build Control message Compute node's ESQL. It does nothing more than pass the Complete message to the out terminal of the Compute node. This message is needed by the AggregateReply node.

Example 8-5 ESQL module for the Compute node Build control message

```
CREATE COMPUTE MODULE Parts_Replenishment_Flow_Compute1
   CREATE FUNCTION Main() RETURNS BOOLEAN
   BEGIN
      CALL CopyEntireMessage();
      RETURN TRUE;
   END;

   CREATE PROCEDURE CopyEntireMessage() BEGIN
      SET OutputRoot = InputRoot;
   END;
END MODULE;
```

Compute node Build workflow response

The fan-in message flow retrieves the control message, which is used by the AggregateReply node. It also retrieves one or more response messages from the suppliers and the original request message. The AggregateReply node constructs a special message tree that has all message headers, all message properties, and all message data grouped in a single structure. This message tree is not usable, for example, in an MQOutput node. A Compute node is required to copy the appropriate parts of the message tree into a new message tree. Example 8-6 on page 398 shows this logic.

The Main function starts with declaring two references to the two distinct parts of the AggregateReplyBody: the Request folder and the Response folder. Remember that the Request folder holds the original workflow request message.

In building the OutputRoot, its first child element has to be Properties. While we do not really use any of the properties, we still initialize it. By referring to one single property (MessageSet), the broker adds the standard structure Properties to the message tree.

The same is true for the second child of the OutputRoot message tree, the message descriptor. By referring to the Version element of the MQMD, the broker knows to add the standard structure MQMD to the message tree. We do a little more with the MQMD: We set the MsgType element to `MQMT_REPLY`, indicating that this message is a reply message. We set the Format to `MQFMT_STRING` and we set Persistence to `MQPER_PERSISTENT`.

Below that shows the first use of the information from the original workflow request message. We copy the UserIdentifier element from the request message to the response message that we are building now to ensure that the user ID is known to the workflow.

The next lines of code sets up the actual XML message for WebSphere MQ Workflow. We indicate that we do not expect a response, because our message will be a response on its own. If you send a request message to WebSphere MQ Workflow, for example to start a process instance, you would normally set this flag to Yes. We then have another use of the original request message. WebSphere MQ Workflow uses its own correlation identifiers, which are part of the request message and must be part of the response message so that the WebSphere MQ Workflow server knows what to do with the incoming XML message. Finally, we set the return code for the execution of this activity.

The next block of code is used to populate the user data structure. This data structure has several simple elements and an array. The first lines of code populate the simple elements. All of those simple elements are populated by copying data from the original request message. The NumberOfParts element is calculated by counting the number of responses in the Response array, which is populated by the AggregateReply node.

To save on typing, we declare a reference, this time to the PartList array in the OutputRoot. The loop condition logic uses yet another way to do a controlled traverse of an array structure. Within the body of the loop, we copy all of the required elements for the i-st entry of Response to the i-st entry of the OutputRoot structure. We then move the pointer refResponse to its sibling, which basically means the same thing as going to the next element in the array. The condition logic of the loop verifies whether the last move operation was successful. An unsuccessful move means that we were moving beyond the boundaries of the array.

Note: By the way that the condition logic is set up, we assume that there is at least one response. The condition `LASTMOVE(refResponse)` always returns true the first time. This technique might not always be appropriate. Another way relies on the actual size of the array, which we calculated earlier by calling the CARDINALITY function.

Finally, when the loop is completed, we set the ReturnCode element of the data structure and complete the ESQL module.

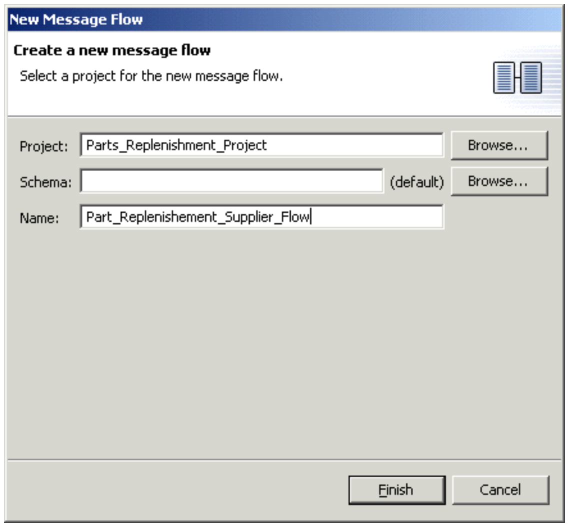
Example 8-6 ESQL module for Compute node Build workflow response

```
CREATE COMPUTE MODULE Parts_Replenishment_Flow_Compute2
   CREATE FUNCTION Main() RETURNS BOOLEAN
   BEGIN

      DECLARE refResponse REFERENCE TO InputRoot.ComIbmAggregateReplyBody.Response;
      DECLARE refRequest REFERENCE TO InputRoot.ComIbmAggregateReplyBody.Request;
      SET OutputRoot.Properties.MessageSet = ' ';
        SET OutputRoot.MQMD.Version = MQMD_CURRENT_VERSION;
        SET OutputRoot.MQMD.MsgType = MQMT_REPLY;
        SET OutputRoot.MQMD.Format = MQFMT_STRING;
      SET OutputRoot.MQMD.Persistence = MQPER_PERSISTENT;
        SET OutputRoot.MQMD.UserIdentifier = refRequest.MQMD.UserIdentifier;

      SET OutputRoot.XML.WfMessage.WfMessageHeader.ResponseRequired  = 'No';
      SET OutputRoot.XML.WfMessage.ActivityImplInvokeResponse.ActImplCorrelID =
         refRequest.XML.WfMessage.ActivityImplInvoke.ActImplCorrelID;
      SET OutputRoot.XML.WfMessage.ActivityImplInvokeResponse.ProgramRC = '0';

      SET
OutputRoot.XML.WfMessage.ActivityImplInvokeResponse.ProgramOutputData.Parts_Replenishment_Form.
WorkOrderNumber =
            refRequest.XML.Part_Replenishment_Form.WorkOrderNumber;
      SET
OutputRoot.XML.WfMessage.ActivityImplInvokeResponse.ProgramOutputData.Parts_Replenishment_Form.
FlowOriginator =
            refRequest.XML.Part_Replenishment_Form.FlowOriginator;
      SET
OutputRoot.XML.WfMessage.ActivityImplInvokeResponse.ProgramOutputData.Parts_Replenishment_Form.
Authorized =
            refRequest.XML.Part_Replenishment_Form.Authorized;

      SET
OutputRoot.XML.WfMessage.ActivityImplInvokeResponse.ProgramOutputData.Parts_Replenishment_Form.
PartsForm.NumberOfParts =
            CARDINALITY(InputRoot.ComIbmAggregateReplyBody.Response[]);
      DECLARE refParts REFERENCE TO

OutputRoot.XML.WfMessage.ActivityImplInvokeResponse.ProgramOutputData.Parts_Replenishment_Form.
PartsForm;
      DECLARE i INTEGER;
      SET i = 1;

      WHILE LASTMOVE(refResponse) DO
         SET refParts.PartList[i].PartOrder.PartNumber =
refResponse.XML.Part_Replenishment.PartNumber;
         SET refParts.PartList[i].PartOrder.Quantity =
refResponse.XML.Part_Replenishment.Quantity;
         SET refParts.PartList[i].PartOrder.InStock =
refResponse.XML.Part_Replenishment.InStock;
         SET refParts.PartList[i].Supplier = refResponse.XML.Part_Replenishment.Supplier;
         SET refParts.PartList[i].CommittedPrice =
refResponse.XML.Part_Replenishment.CommittedPrice;
         SET refParts.PartList[i].OrderReference =
refResponse.XML.Part_Replenishment.OrderReference;
         SET refParts.PartList[i].ExpectedDate =
refResponse.XML.Part_Replenishment.ExpectedDate;
         SET refParts.PartList[i].ShipmentDate =
refResponse.XML.Part_Replenishment.ShipmentDate;
         SET refParts.PartList[i].CaseNumber = refResponse.XML.Part_Replenishment.CaseNumber;
         SET refParts.PartList[i].ReceptionDate =
refResponse.XML.Part_Replenishment.ReceptionDate;
         SET refParts.PartList[i].RegistrationDate =
refResponse.XML.Part_Replenishment.RegistrationDate;
         SET refParts.PartList[i].InvoicedPrice =
refResponse.XML.Part_Replenishment.InvoicedPrice;

         MOVE refResponse NEXTSIBLING NAME FIELDNAME(refResponse);
         SET i = i + 1;
      END WHILE;

      SET
OutputRoot.XML.WfMessage.ActivityImplInvokeResponse.ProgramOutputData.Parts_Replenishment_Form.
ReturnCode = '0';
      RETURN TRUE;
   END;

END MODULE;
```

8.5 Supporting message flow

In our setup, there is no implemented business-to-business solution that interacts with suppliers. The response of the suppliers is simulated with another message flow consisting of two subflows: the first one builds actual response messages and calculates values for elements such as InvoicedPrice, and the other subflow is responsible for passing back the original workflow request message so that it seems to be part of the aggregation.

1. While in the Broker Application Development perspective, select **File** → **New** → **Message Flow**. Provide a name for the flow (we use `Part_Replenishment_Supplier_Flow`) and make sure it is defined in the same message flow project, as shown in Figure 8-11.


New Message Flow

Create a new message flow

Select a project for the new message flow.

Project: Parts_Replenishment_Project

Schema: (default)

Name: Part_Replenishement_Supplier_Flow

Figure 8-11 Create new message flow

2. Add two MQInput nodes and two MQReply nodes to the message editor, and add a Compute node and connect them as shown in Figure 8-12.

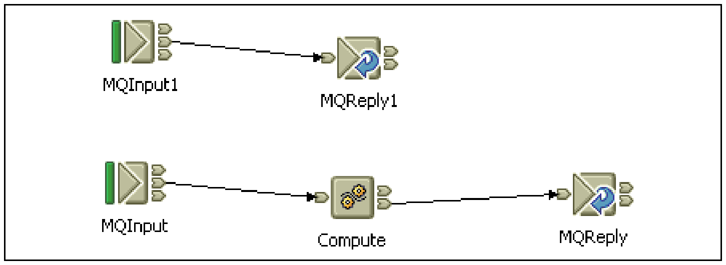


Figure 8-12 Connected flow

The top MQInput node refers to the ORDER.SAVE queue and expects an XML message. The bottom MQInput node refers to the ORDER.REQUEST queue and also expects XML messages. There are no specific parameters to configure for the MQReply nodes.

Example 8-7 lists the ESQL module that sits behind the Compute node in Figure 8-12. It first copies the full request message in the OutputRoot message tree. then adds the ExpectedDate and ShipmentDate elements. These elements are calculated based on the current date and by adding either 7 or 14 days to it.

The CaseNumber is calculated using a randomizer, which is initialized by casting the input element Quantity as an Integer. The resulting random value is multiplied by 10,000 and then truncated. It is then cast as an Integer.

Similar logic is used for the InvoicedPrice element. Here the actual value is truncated to two decimal digits.

Note: The logic in the ESQL module is not really intended to be standard business logic. It is merely intended to populate the response message with some data that is different for each incoming request.

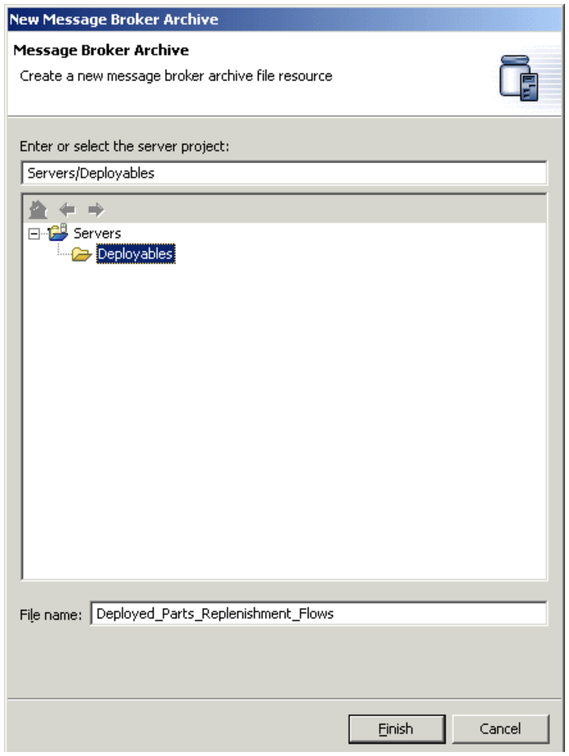
Example 8-7 ESQL module to build a supplier response message

```
CREATE COMPUTE MODULE Part_Replenishement_Supplier_Flow_Compute
   CREATE FUNCTION Main() RETURNS BOOLEAN
   BEGIN
      CALL CopyEntireMessage();
      SET OutputRoot.XML.Part_Replenishment.ExpectedDate = CURRENT_DATE + INTERVAL '7' DAY;
      SET OutputRoot.XML.Part_Replenishment.ShipmentDate = CURRENT_DATE + INTERVAL '14' DAY;
      SET OutputRoot.XML.Part_Replenishment.CaseNumber =
CAST(TRUNCATE(RAND(CAST(InputRoot.XML.Part_Replenishment.Quantity AS INTEGER)) * 10000, 0) AS
INTEGER);
      SET OutputRoot.XML.Part_Replenishment.InvoicedPrice =
TRUNCATE(RAND(CAST(InputRoot.XML.Part_Replenishment.Quantity AS INTEGER)) * 500, 2);
      SET OutputRoot.XML.Part_Replenishment.CommittedPrice =
TRUNCATE(RAND(CAST(InputRoot.XML.Part_Replenishment.Quantity AS INTEGER)) * 1000, 2);
      SET OutputRoot.XML.Part_Replenishment.ReceptionDate = CURRENT_DATE + INTERVAL '2' DAY;
      SET OutputRoot.XML.Part_Replenishment.RegistrationDate = CURRENT_DATE + INTERVAL '3'
DAY;
      SET OutputRoot.XML.Part_Replenishment.Supplier = 'IBM ITSO';
       SET OutputRoot.XML.Part_Replenishment.OrderReference =  CURRENT_TIMESTAMP;
      RETURN TRUE;
   END;

   CREATE PROCEDURE CopyMessageHeaders() BEGIN
      DECLARE I INTEGER 1;
      DECLARE J INTEGER CARDINALITY(InputRoot.*[]);
      WHILE I < J DO
         SET OutputRoot.*[I] = InputRoot.*[I];
         SET I = I + 1;
      END WHILE;
   END;

   CREATE PROCEDURE CopyEntireMessage() BEGIN
      SET OutputRoot = InputRoot;
   END;
END MODULE;
```

8.6 Deployment and testing of the message flow

The development of the message flows is now complete and we can deploy the solution to the broker.

1. The deployment process is performed in the Broker Toolkit Broker Administration. Within this perspective, select **File** → **New** → **Message Broker Archive** to start the deployment process.
2. Provide a name for the archive, such as `Deployed_Parts_Replenishment_Flows`, and select a target folder (Figure 8-13).


New Message Broker Archive

Message Broker Archive

Create a new message broker archive file resource

Enter or select the server project:

Servers/Deployables

Servers

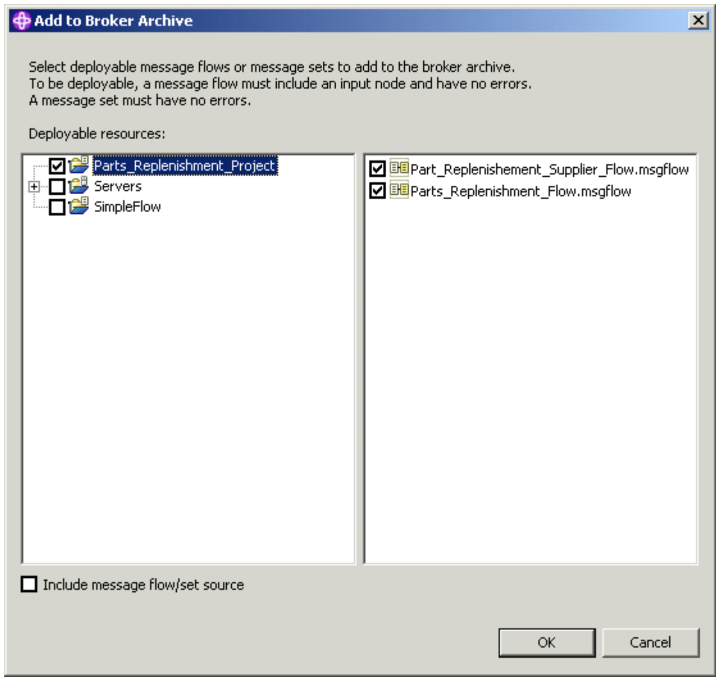
Deployables

File name: Deployed_Parts_Replenishment_Flows

Figure 8-13 Create new message broker archive

The broker archive editor is now open.

3. Click the green icon (see Figure 8-15) to add resources to this archive. The Add to Broker Archive window is now shown, enabling you to select message flows and message sets from workspace projects. Select **Parts_Replenishment_Project** and click **OK**.



Select deployable message flows or message sets to add to the broker archive.
To be deployable, a message flow must include an input node and have no errors.
A message set must have no errors.

Deployable resources:

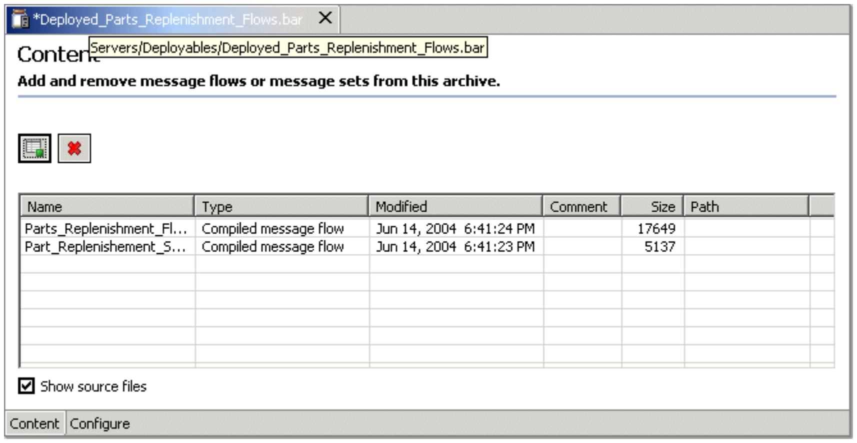
[✓] Parts_Replenishment_Project
[ ] Servers
[ ] SimpleFlow

[✓] Part_Replenishement_Supplier_Flow.msgflow
[✓] Parts_Replenishment_Flow.msgflow

[ ] Include message flow/set source

Figure 8-14 Add resources to the broker archive

The resources are now compiled. Save the broker archive. See Figure 8-15 for a a view of the populated broker archive.



*Deployed_Parts_Replenishment_Flows.bar

Content — Servers/Deployables/Deployed_Parts_Replenishment_Flows.bar

Add and remove message flows or message sets from this archive.

Name	Type	Modified	Comment	Size	Path
Parts_Replenishment_Fl...	Compiled message flow	Jun 14, 2004 6:41:24 PM		17649	
Part_Replenishement_S...	Compiled message flow	Jun 14, 2004 6:41:23 PM		5137	

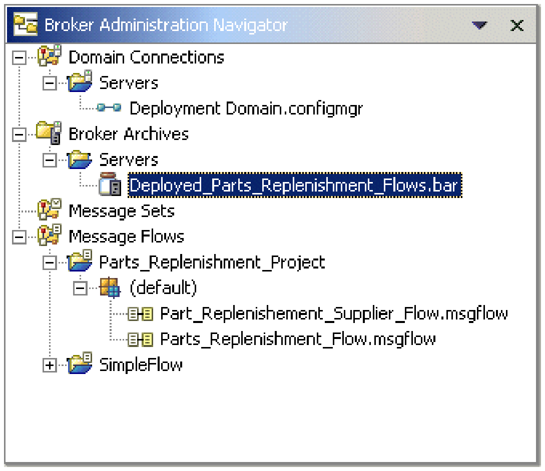
☑ Show source files

Content | Configure

Figure 8-15 Populated broker archive

4. Switch to the Broker Administration Navigator view (Figure 8-16), where you will find the broker archive. Before deploying this archive to the broker, make sure that you are connected to the configuration manager of your broker domain. You can verify the connection status by inspecting the Domains view shown in Figure 8-18 on page 407. If you are not connected, right-click the configuration manager entry (**CMQM@wbimb:1415**) and select **Connect**. Of course, the configuration manager and the broker should be running at this time.

   When you are connected, right-click the archive and select **Deploy File**.



```
Broker Administration Navigator
Domain Connections
  Servers
    Deployment Domain.configmgr
Broker Archives
  Servers
    Deployed_Parts_Replenishment_Flows.bar
Message Sets
Message Flows
  Parts_Replenishment_Project
    (default)
      Part_Replenishement_Supplier_Flow.msgflow
      Parts_Replenishment_Flow.msgflow
  SimpleFlow
```

Figure 8-16 Broker navigator

5. A new window opens for selecting an execution group within a broker to which this archive must be deployed. Select the Default execution group and click **OK**.

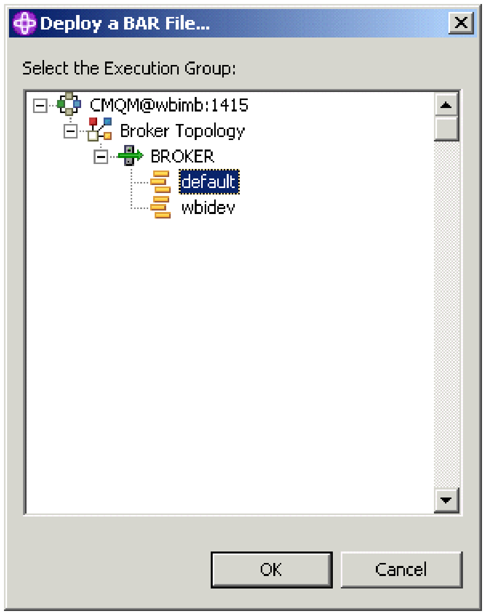


Figure 8-17 Deploy to an execution group of a broker

The deployment process is a two-step asynchronous process. First, the Toolkit sends the deployment request to the configuration manager, which performs several validation steps. If these validation steps do not discover any problems, the configuration manager forwards a modified version of the archive to the actual broker. At that time, the configuration manager replies back to the Toolkit and you receive a prompt about successful initiation of the deployment.

When the broker receives the deployment message from the configuration manager, it also performs validation and stores the archive in its database. When the process completes, either successfully or unsuccessfully, the broker publishes the status of the deployment. The configuration manager subscribes to deployment status messages, which you can see by accessing the Event Log in the Domains view of the Broker Toolkit (Figure 8-18). Note that the Event Log is not the same tool as the Event Viewer, which is also used by the broker and the configuration manager.

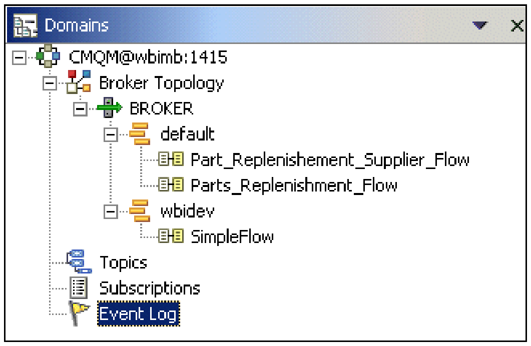


Figure 8-18 The view Domains

8.7 Testing from the Web Client

When we create an instance of the Order Process in the WebSphere MQ Workflow Web Client, the process executes the first two activities automatically. The Create Sales Order activity is implemented by the SalesOrderManagement collaboration in the InterChange Server and returns to the WebSphere MQ Workflow server a new order number. It also stores the order and order line items in the database.

The next activity is implemented by the message flow that is discussed in this chapter. When the response messages are retrieved from the supplier simulator message flow and the workflow response message is picked up in the WebSphere MQ Workflow server, the activity completes and we move on to the Approve Order activity.

Performing the Approve Order activity is a user activity that we discuss in Chapter 6, “Implementing a process model in WebSphere MQ Workflow” on page 201.

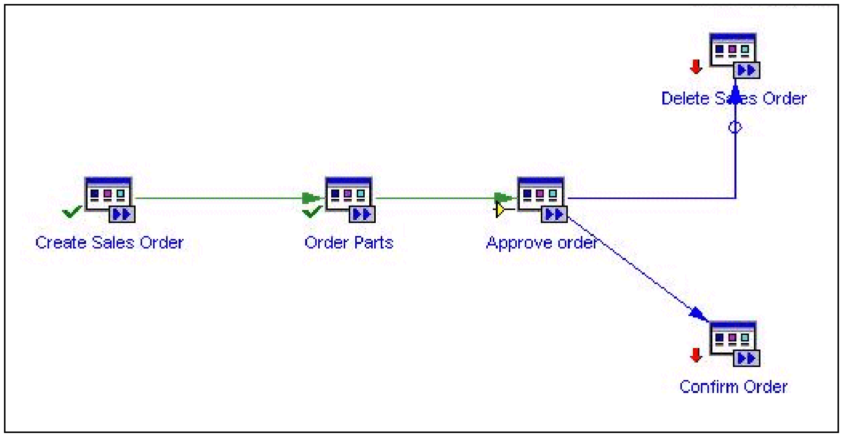


Figure 8-19 Process monitor after completing the Order Parts activity

Part 3

Managing a business integration solution

9 Handling deployment and change

The previous chapters in this book are mostly about developing and testing integration solutions. In this chapter we discuss production run time issues: how we move integration solutions from a development or test environment to a production environment and how we handle changes in a production environment.

A production WebSphere Business Integration Server must be able to cope with changing requirements. Business logic, data models, and enterprise systems can change during the lifetime of a WebSphere Business Integration Server.

Business processes can run from seconds to several years. It is typically not possible to wait for termination or to actively terminate all instances of a given business process. Therefore, different versions of a given business process have to be able to run at the same time in parallel.

9.1 Preparing for production deployment

To build a production runtime environment, you can refer to the first part of this book, in which several installation types were discussed. Implementation of runtime servers and of runtime and management clients were are discussed in Chapter 3, “Implementing the runtime components” on page 45 and Chapter 4, “Implementing client components” on page 119. In this section, we discuss the process of moving integration solutions from one environment to another.

9.1.1 Preparing an WebSphere MQ Workflow solution for production

There are certain procedures you must take before promoting a WebSphere MQ Workflow solution for production. As with any application, you would want to be able to manage it, especially in an environment where the system is setup for multiple developers. Having said this, WebSphere MQ Workflow (v.3.5) does not include a versioning control feature. As a result, there are no check in or checkout options employed by MQ Workflow Buildtime. Unless you are licensed to use WBI Workbench Server (which makes use of a centralized database), a common practice is to utilize a library management system that you might have already. In addition, you must establish a set of standard operation procedures to be integrated into your change control process.

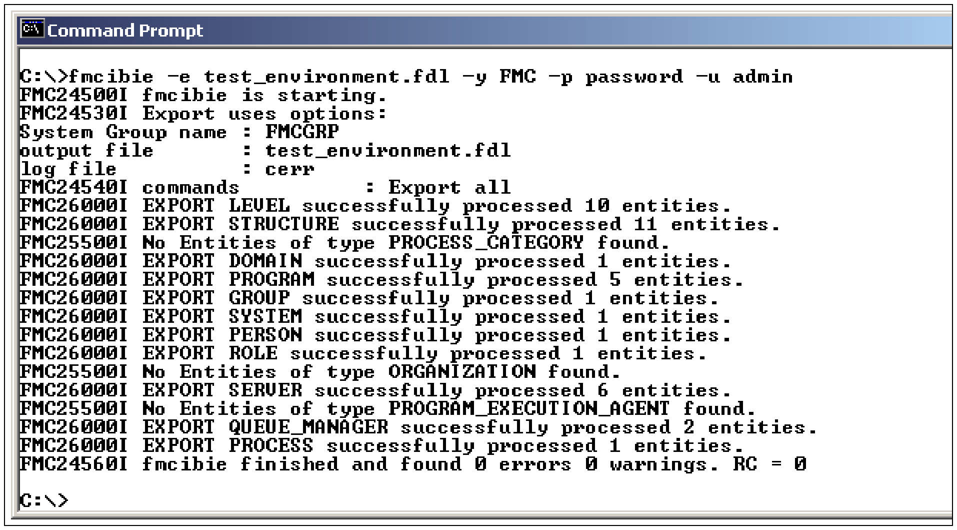
Other aspects to consider before migrating a FDL from development to production include object distinction and authorization. Certain definitions might not reference the same development resources (MQ objects) as in production. By no means, do you want to grant your user community the same privileges in production that are recognized in development. Therefore, it recommended that you segregate all topology settings, people, and individual processes into separate FDLs.

Assuming that your production environment is up and running with all the necessary permissions and WebSphere MQ resources, we begin the migration.

The procedures are as follows:

1. After logging-on the test server, go to a command prompt and **enter:**

   ```
   fmcibie -e test_environment.fdl -y FMC -p password -u admin
   ```

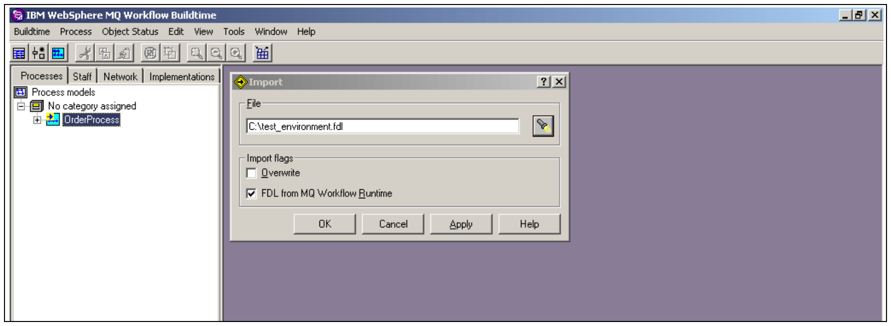


```
C:\>fmcibie -e test_environment.fdl -y FMC -p password -u admin
FMC24500I fmcibie is starting.
FMC24530I Export uses options:
System Group name : FMCGRP
output file       : test_environment.fdl
log file          : cerr
FMC24540I commands          : Export all
FMC26000I EXPORT LEVEL successfully processed 10 entities.
FMC26000I EXPORT STRUCTURE successfully processed 11 entities.
FMC25500I No Entities of type PROCESS_CATEGORY found.
FMC26000I EXPORT DOMAIN successfully processed 1 entities.
FMC26000I EXPORT PROGRAM successfully processed 5 entities.
FMC26000I EXPORT GROUP successfully processed 1 entities.
FMC26000I EXPORT SYSTEM successfully processed 1 entities.
FMC26000I EXPORT PERSON successfully processed 1 entities.
FMC26000I EXPORT ROLE successfully processed 1 entities.
FMC25500I No Entities of type ORGANIZATION found.
FMC26000I EXPORT SERVER successfully processed 6 entities.
FMC25500I No Entities of type PROGRAM_EXECUTION_AGENT found.
FMC26000I EXPORT QUEUE_MANAGER successfully processed 2 entities.
FMC26000I EXPORT PROCESS successfully processed 1 entities.
FMC24560I fmcibie finished and found 0 errors 0 warnings. RC = 0

C:\>
```

Figure 9-1 Fmcibie Export

As you can see from Figure 9-1, the **`fmcibie`** (**`-e`** = export) utility extracts all information about the runtime server, including domain information and user-defined program execution servers. The export in Figure 9-1 can also be achieved by utilizing a centralized buildtime environment.

2. Log-in to the Buildtime, Go to **Buildtime** → **import.**

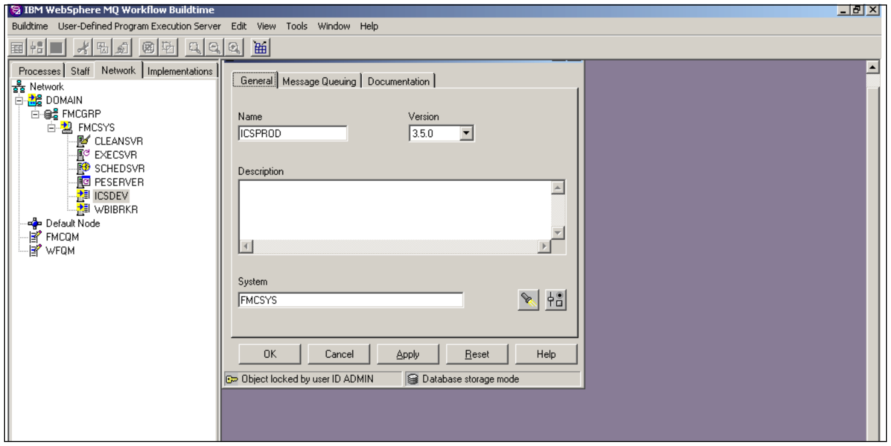

IBM WebSphere MQ Workflow Buildtime
Buildtime Process Object Status Edit View Tools Window Help
Processes Staff Network Implementations
Process models
No category assigned
OrderProcess
Import
OK Cancel Apply Help

File: C:\test_environment.fdl

Import flags:
- [ ] Overwrite
- [✓] FDL from MQ Workflow Runtime

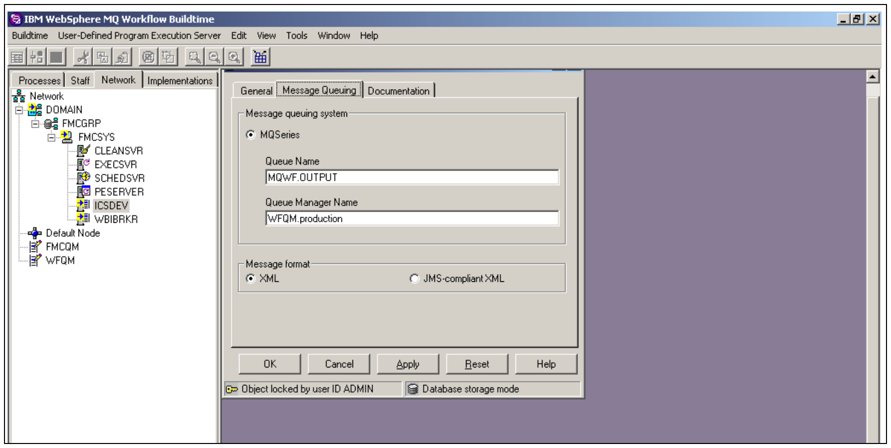
Figure 9-2 Importing runtime FDL in Buildtime

When importing runtime FDL into Buildtime, make sure that you mark this in the import dialog, as shown in Figure 9-2. After the test_environment is imported into your buildtime database, make the necessary changes to the FDL to make it production specific. For example, the General and Message Queuing tab for both UPES ICSDEV and UPES WBIBRKR must be changed as shown in figures Figure 9-3 and Figure 9-4 on page 414. If you left the queue manager name blank in test, there will be no need to edit the Message queuing tab.


IBM WebSphere MQ Workflow Buildtime
Buildtime User-Defined Program Execution Server Edit View Tools Window Help
Processes Staff Network Implementations
Network
DOMAIN
FMCGRP
FMCSYS
CLEANSVR
EXECSVR
SCHEDSVR
PESERVER
ICSDEV
WBIBRKR
Default Node
FMCQM
WFQM
General Message Queuing Documentation
OK Cancel Apply Reset Help

Name: ICSPROD
Version: 3.5.0
Description:
System: FMCSYS
Object locked by user ID ADMIN
Database storage mode

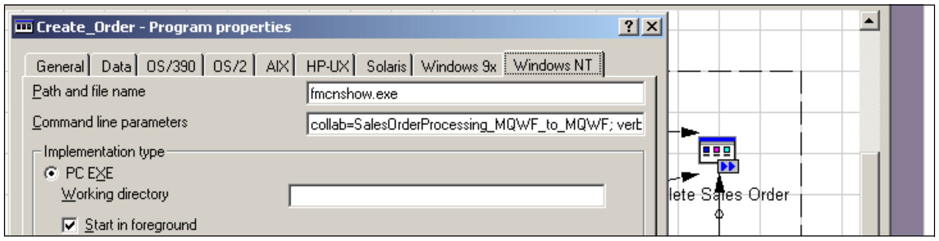
Figure 9-3 UPES General tab


IBM WebSphere MQ Workflow Buildtime
Buildtime User-Defined Program Execution Server Edit View Tools Window Help
Processes Staff Network Implementations
Network
DOMAIN
FMCGRP
FMCSYS
CLEANSVR
EXECSVR
SCHEDSVR
PESERVER
ICSDEV
WBIBRKR
Default Node
FMCQM
WFQM
General Message Queuing Documentation
OK Cancel Apply Reset Help

Message queuing system: [✓ MQSeries]
Queue Name: MQWF.OUTPUT
Queue Manager Name: WFQM.production
Message format: [✓ XML] JMS-compliant XML
Object locked by user ID ADMIN
Database storage mode

Figure 9-4 UPES Message Queuing tab

3. Review and edit the program objects. Validate whether the names assigned to the collaboration in the production InterChange Server environment are the same as in the development Interchange environment. If the names are different, certain objects such as command line parameters shown in FigFigure 9-5 on page 415 should be edited.


Create_Order - Program properties

Path and file name: fmcnshow.exe

Command line parameters: collab=SalesOrderProcessing_MQWF_to_MQWF; verb

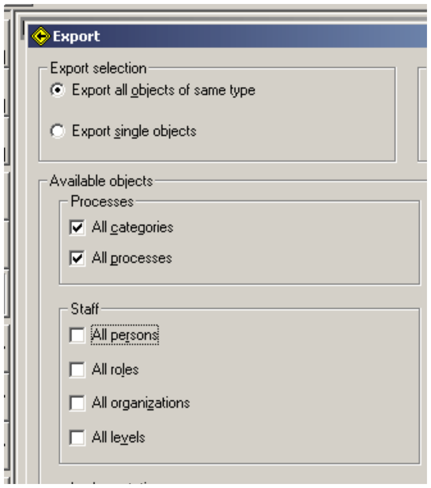
Implementation type: PC EXE (selected)

Working directory:

[✓] Start in foreground

Figure 9-5 Command line parameters

4. When all necessary modifications are made in Buildtime, export the newly prepared production solution. If your production environment relies on user ID management and not on LDAP, for example, be careful not overwrite runtime server passwords with development server passwords. To avoid this scenario, *deselect* the Staff options in bottom of Figure 9-6.


Export

Export selection

(●) Export all objects of same type

( ) Export single objects

Available objects

Processes

[✓] All categories

[✓] All processes

Staff

[ ] All persons

[ ] All roles

[ ] All organizations

[ ] All levels

Figure 9-6 Staff

Another way of preparing the test solution for production is to open the FDL in a text editor, make the necessary changes or additions, and import the file into the production runtime environment. However, the easier method is to utilize the Buildtime tool.

The overall goal in preparing a production solution is to selectively export and import the resources that you need. As part of this promotion process, the underlying WebSphere MQ framework must be defined. The WebSphere MQ Workflow server, the InterChange Server, and the message broker all use the services of a queue manager. Intercommunication among the three runtime servers is done using WebSphere MQ. Therefore, to move a WebSphere MQ

Workflow solution to production, make sure that channels and queues are defined.

The queues that Workflow requires are in the UPES definitions. Workflow does not make any assumptions about the type of queue. They could be alias queues, remote queues, or cluster queues. Make sure that the queue defined to the Workflow server for sending request messages is defined to MQ so that the message ends up on the queue named in the MQInput node of the message flow, or on a remote queue that points to an input queue for the WebSphere MQ Workflow connector. The input queue for the connector is the InputQueue configuration property.

The WebSphere MQ Workflow server expects to find responses in its FMC.FMCGRP.EXE.XML queue. This is a clustered queue on the workflow queue manager. Thus, making the InterChange Server and message broker queue manager part of the workflow cluster is the quickest way to ensure that the InterChange Server and the broker can use this queue. Similarly, making the message flow input queue and the connector input queue part of the cluster is the quickest way to achieve intercommunication. For our discussion, the broker and application server queue managers are clustered with the workflow server. The InterChange server is connected to the Workflow server through sender/receiver channels. Commands for setting up cluster queues and for queue managers to join a cluster are given in Chapter 7, “Sales order management in InterChange Server” on page 249, and in Chapter 8, “Replenishing parts in WebSphere BI Message Broker” on page 377.

After all changes are made to the process model and to external resources, you can export the updated FDL from Buildtime and import it to the production runtime server using the fmcibie utility. Assuming that deploying a solution only involves adding resources and process models, there are no operational implications. When the import is completed, the process model is ready for end users. It appears in their list of process templates if they are authorized to use the new process template.

9.1.2 Preparing a message broker solution for production

The main solution artifact for message broker solutions is the message broker archive. To move a solution to production, export the message broker archive from the development system:

1. While in the Broker Administration perspective, select **File** → **Export**. Select **File system** (Figure 9-7) as the target for the export and click **Next**.

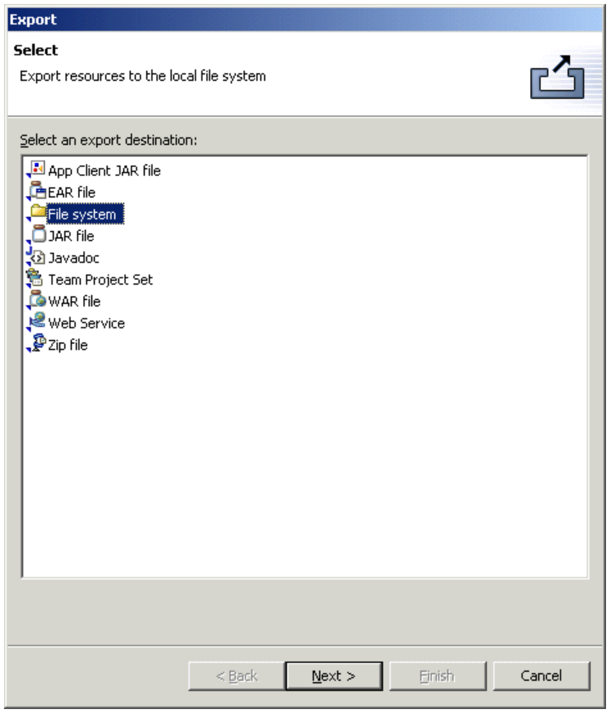
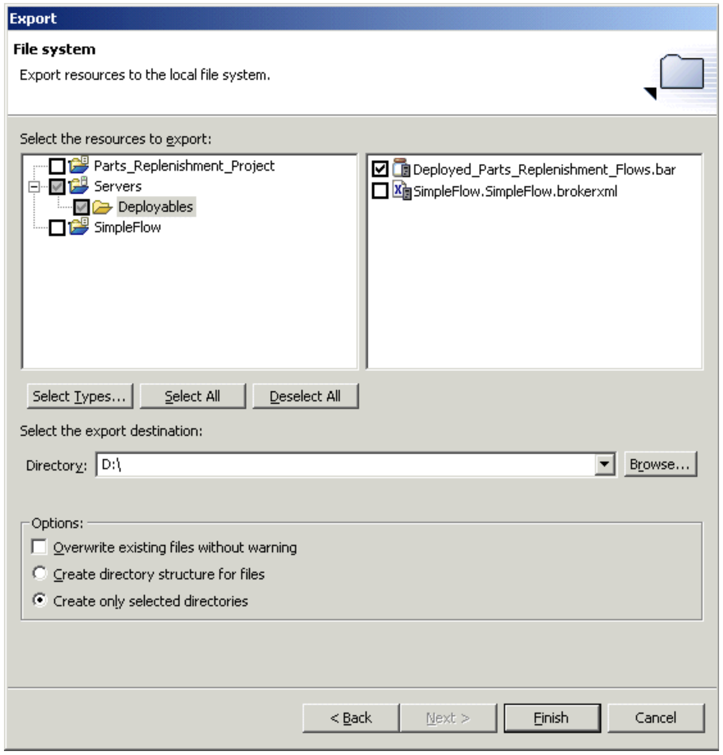


Figure 9-7 Select target for export operation

2. In the next window (Figure 9-8), select the resources that you want to export, the broker archive **Deployed_Parts_Replenishment_Flows**. Select a destination directory and click **Finish**.



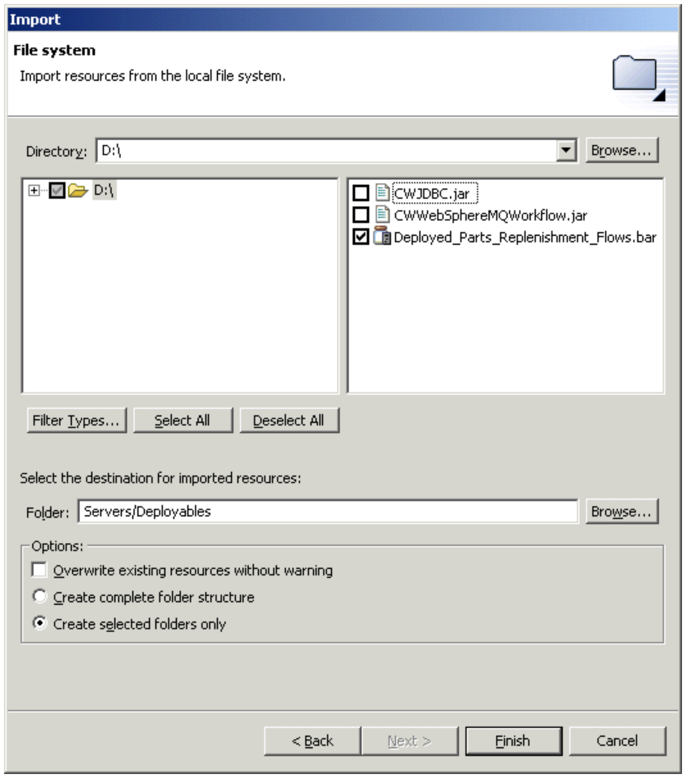
[✓] Deployed_Parts_Replenishment_Flows.bar
[ ] SimpleFlow.SimpleFlow.brokerxml

Select the export destination:
Directory: D:\

Options:
[ ] Overwrite existing files without warning
( ) Create directory structure for files
(●) Create only selected directories

Figure 9-8 Select resources to export

3. This broker archive can now be imported into the Broker Toolkit that is linked to the production configuration manager and broker, such as the management client that we implement in Chapter 4, “Implementing client components” on page 119. Within this instance of the Broker Toolkit, select **File** → **Import** (Figure 9-9). Select **File System** as the source for the import and click **Next**. Identify the directory for the broker archive, select it, and provide a suitable destination folder in the workspace of this instance of the Broker Toolkit.


Import

File system

Import resources from the local file system.

Directory: D:\

D:\

☐ CWJDBC.jar
☐ CWWebSphereMQWorkflow.jar
☑ Deployed_Parts_Replenishment_Flows.bar

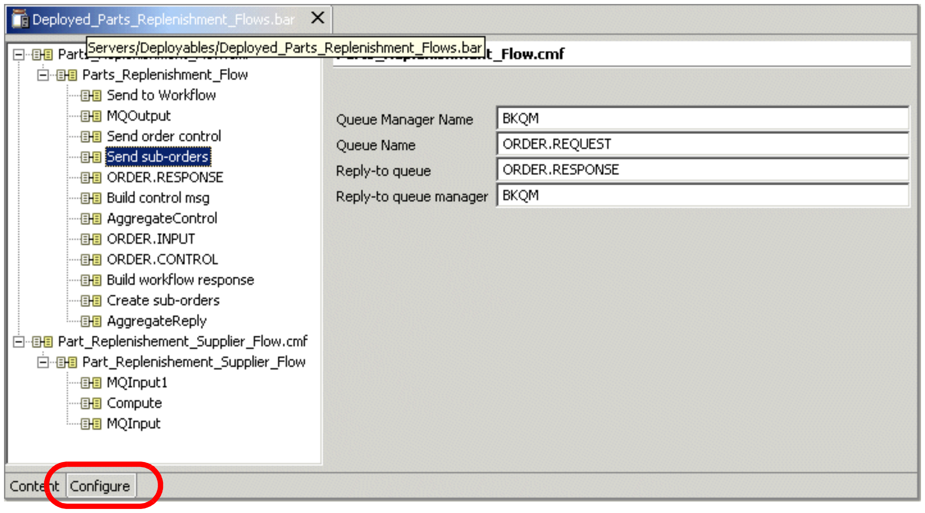
Select the destination for imported resources:

Folder: Servers/Deployables

Options:
☐ Overwrite existing resources without warning
○ Create complete folder structure
◉ Create selected folders only

Figure 9-9 Select broker archive to import in production environment

4. Similar to a process model, a message broker solution refers to several external resources. This could be queue names, file names (for the Trace node), database names, and so on. The message broker archive editor enables you to verify and change all of these names without requiring changes to the message flows themselves. Double-click the imported broker archive and select the **Configure** tab in the broker archive editor (Figure 9-10 on page 420).


Deployed_Parts_Replenishment_Flows.bar

Servers/Deployables/Deployed_Parts_Replenishment_Flows.bar

- Parts_Replenishment_Flow.cmf
  - Parts_Replenishment_Flow
    - Send to Workflow
    - MQOutput
    - Send order control
    - Send sub-orders
    - ORDER.RESPONSE
    - Build control msg
    - AggregateControl
    - ORDER.INPUT
    - ORDER.CONTROL
    - Build workflow response
    - Create sub-orders
    - AggregateReply
- Part_Replenishement_Supplier_Flow.cmf
  - Part_Replenishement_Supplier_Flow
    - MQInput1
    - Compute
    - MQInput

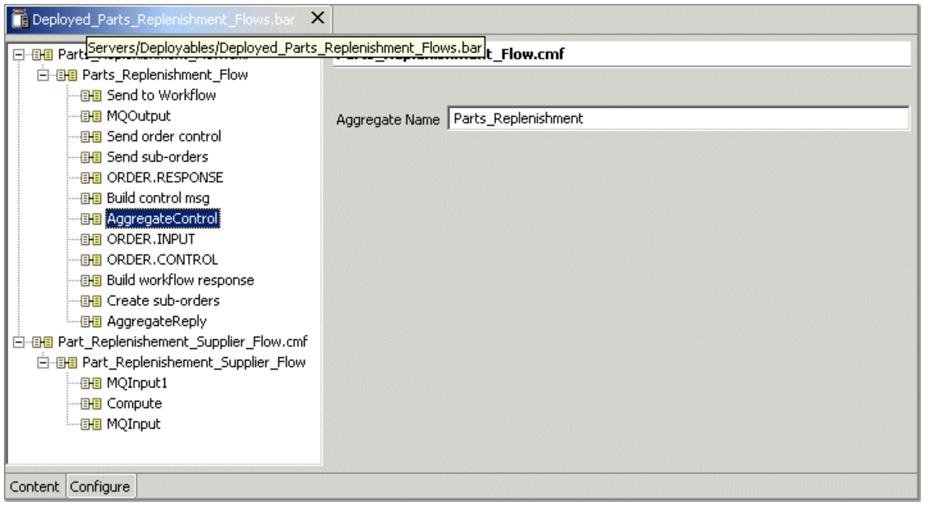
Queue Manager Name	BKQM
Queue Name	ORDER.REQUEST
Reply-to queue	ORDER.RESPONSE
Reply-to queue manager	BKQM

Figure 9-10 Updating external references for an MQOutput node

This presents a list of all nodes that rely on some kind of an external resource. Figure 9-10 shows the external resources for the Send suborders MQOutput node. The external resources are a destination queue name and queue manager name, and the reply-to queue name and reply-to queue manager name. Remember from Chapter 8, “Replenishing parts in WebSphere BI Message Broker” on page 377 that this node is used to send request messages and that the broker expects the corresponding responses to arrive at the queue referenced in the MQInput node of another flow. You can use this editor to update these names to whatever names are applicable in the production environment.

Similarly, when you select an MQInput node, such as ORDER.INPUT, you can change the name of the actual input queue for this message flow. Note that you cannot change the name of the queue manager because this is fixed to the name of the queue manager that is used by the broker itself. A broker only receives messages from its own queue manager, but it can send messages to any queue manager for which channels are defined.

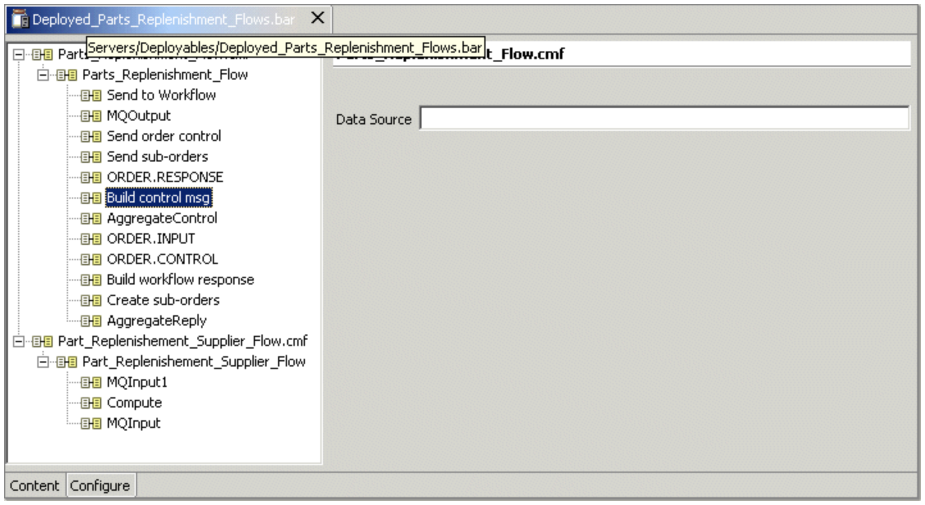
Selecting one of the Aggregate nodes, such as AggregateControl, enables you to change the name of the aggregation, as shown in Figure 9-11 on page 421. Remember that this name has to be unique within a broker. Given that two integration solutions might have been developed by different teams, there is no guarantee that they have used different names. As the broker administrator, you have the option to change it to prevent conflicts between two or more integration solutions that use the aggregation nodes.



Aggregate Name: Parts_Replenishment

Figure 9-11 Updating the aggregation name

Figure 9-12 shows the external resources for a Compute node. In our scenario, we did not use any databases in the Compute nodes. However, if you had used external resources, the broker archive editor could be used to update the name of the database because you cannot assume that the production database has the same name as the development database.



Data Source:

Figure 9-12 Updating data source references

The broker archive editor is an easy way to customize the archive before deployment to the production broker, and the editor can be used to find out which resources must be defined and available in the production environment. To obtain a list of all queues that are used by a message flow, you do not have to review every possible MQInput, MQOutput, or Compute node. The editor lists them for you. But you still have to define them manually.

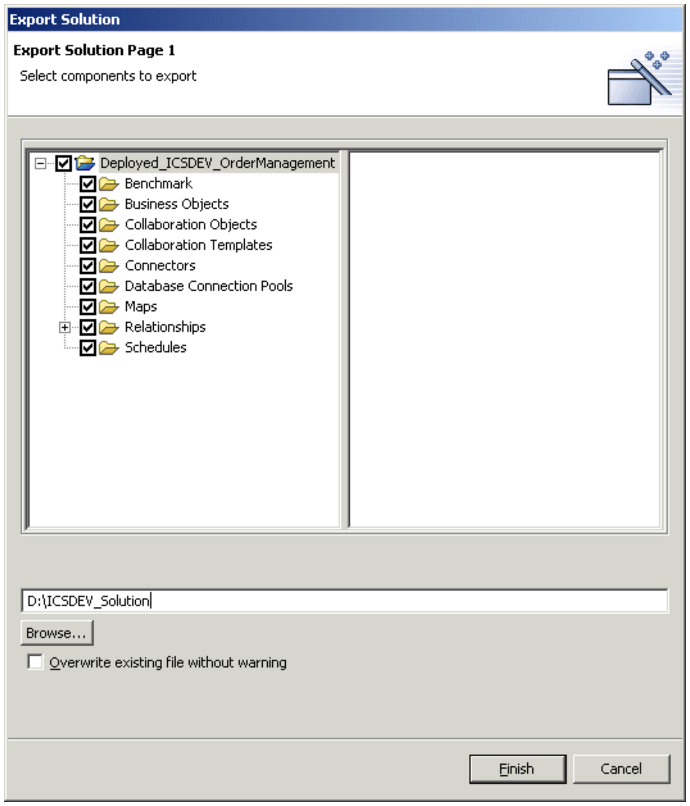
After all changes have been made to the broker archive, it can be deployed to the production brokers. Assuming that the solution does not depend on other message flows, there is no operational impact. When the deployment is completed, the execution group or groups to which the message flow is assigned open the input queues of the message flows and start retrieving and processing messages.

9.1.3 Preparing InterChange Server solutions for production

The InterChange Server has a similar archive concept, which is called a *repository JAR file*. This file can be exported from the development environment and imported into the System Manager that is linked to the production InterChange Server. However, this JAR file is not the only solution artefact. Start-up scripts for connectors and configuration files for connectors must to be changed as well. Also, adding or updating connectors has an operational impact. The new of updated connectors cannot be used until the InterChange Server has been restarted. This, of course, has operational implications for other integration solutions that are deployed to the same InterChange Server.

Exporting and importing a repository JAR file is one way to handle deployment from one environment to another. An alternative way is to export and import a solution. A solution is the term that is used to describe the collection of artefacts in an integration component library and the collection of shortcuts in a user project, which is what is deployed to an InterChange Server.

To export a solution, right-click the **InterChange Server Projects** folder in System Manager and select **Export Solution**. Select one or more user projects to be exported and provide a folder name (Figure 9-13).


Export Solution

Export Solution Page 1

Select components to export

- [✓] Deployed_ICSDEV_OrderManagement
  - [✓] Benchmark
  - [✓] Business Objects
  - [✓] Collaboration Objects
  - [✓] Collaboration Templates
  - [✓] Connectors
  - [✓] Database Connection Pools
  - [✓] Maps
  - [✓] Relationships
  - [✓] Schedules

D:\ICSDEV_Solution

Figure 9-13   Export a solution from System Manager

The result of exporting a solution is a directory structure that contains a System folder and a User folder (Figure 9-14). The System folder contains the actual components in source format. The User folder contains the shortcuts.

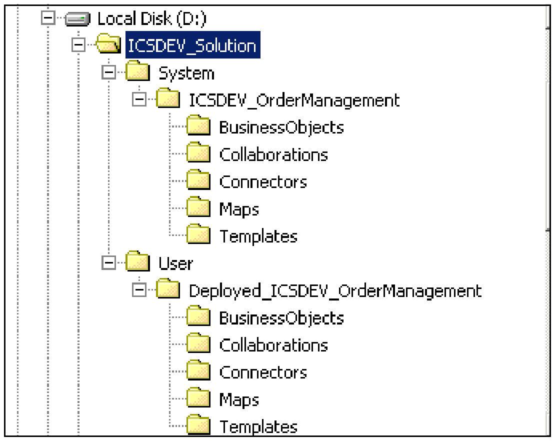
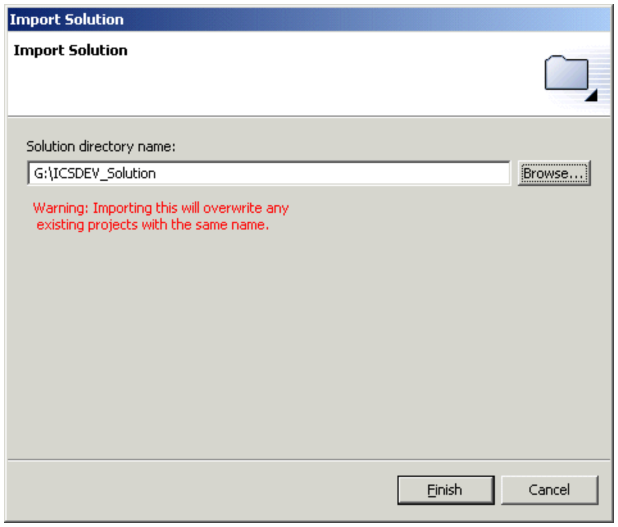

```
Local Disk (D:)
  ICSDEV_Solution
    System
      ICSDEV_OrderManagement
        BusinessObjects
        Collaborations
        Connectors
        Maps
        Templates
    User
      Deployed_ICSDEV_OrderManagement
        BusinessObjects
        Collaborations
        Connectors
        Maps
        Templates
```

Figure 9-14 Structure of an exported solution

When you make such a solution export available to the production environment, you can perform a solution import. Right-click the **InterChange Server** projects folder and select **Import Solution** (Figure 9-15). Provide the name of the directory to which the solution was exported and click **Finish**.



Solution directory name: G:\ICSDEV_Solution

Warning: Importing this will overwrite any existing projects with the same name.

Figure 9-15 Import solution

As you can see in Figure 9-16, the result is that the ICL and the user project are restored within the production environment. The solution (the user project) can now be deployed immediately to the production InterChange Server.

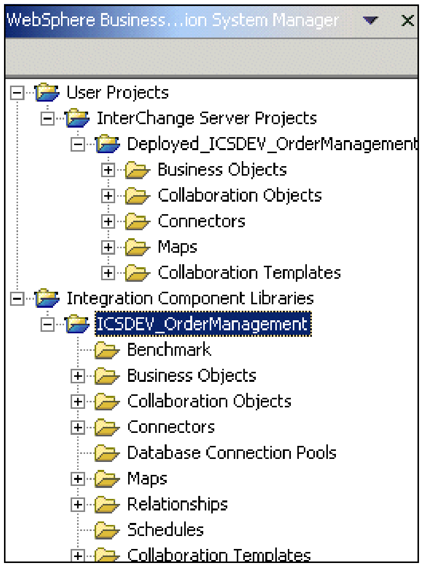


Figure 9-16 Imported solution

Before we can deploy this user project, we must ensure that any references to external resources are consistent with the resources that are available in the production environment. In most cases, this is limited to reviewing the connector configuration, which is expected because connectors act as the interface between the InterChange Server and the production environment of your applications. Some collaborations, which either are provided with the product or are available as additional products, also have references to external resources, such as databases. But that is not the general case.

For all connectors, review the standard properties in the Connector Configurator. The JMS.MessageBrokerName property refers to the name of the queue manager that is associated with the InterChange Server. Usually, the name of the development queue manager is not the same as the name of the production queue manager. The standard properties also contain several queues for each connector and you must define these queues on the production queue manager. In describing the implementation of the connectors in the development environment, we saved a script for each connector. Use these scripts again for implementing the connectors in the production environment. The names that we used are:

- WMQWFconnector_mq.tst

- JDBCconnector_mq.tst
- Portconnector_mq.tst

When you review the connector-specific properties of the JDBC connector, you likely will have to make changes for the following properties:

- ApplicationPassword and ApplicationUsername
- DatabaseURL
- SchemaName
- Archive and event tables

Therefore, make sure that the application database is available for the JDBC connector. For DB2, this means that you must catalog the remote database and system in the local directory. Depending on your setup, you might have to use a different user ID to access that database. Also, define the archive and event tables in the production application database.

When you review the connector-specific properties of the WebSphere MQ Workflow connector, you likely will make changes for the following properties:

- ApplicationPassword and ApplicationUserID (a WebSphere MQ Workflow user ID, such as ADMIN)
- WorkflowSystemName and WorkflowSystemGroup
- MQSeriesQueueManager, MQSeriesHostName, MQSeriesPort, and MQSeriesChannel, which are the properties that the connector uses to retrieve and send workflow messages

The connector-specific properties also list five queues that the connector uses to manage its interaction with the WebSphere MQ Workflow server. Likely the names do not need any changes, but you should define these queues to the queue manager mention in MQSeriesQueueManager.

If the MQSeriesQueueManager property is not the name of the queue manager associated with the WebSphere MQ Workflow server (but it is, for example, the name of the queue manager that is associated with the InterChange Server), set up MQ communication between both queue managers.

When all of the changes have been made to the connector configuration, save them to the project and to a file. The need to use an external configuration file for each connector is a consequence of using JMS as the transport type. For IDL, this is not required.

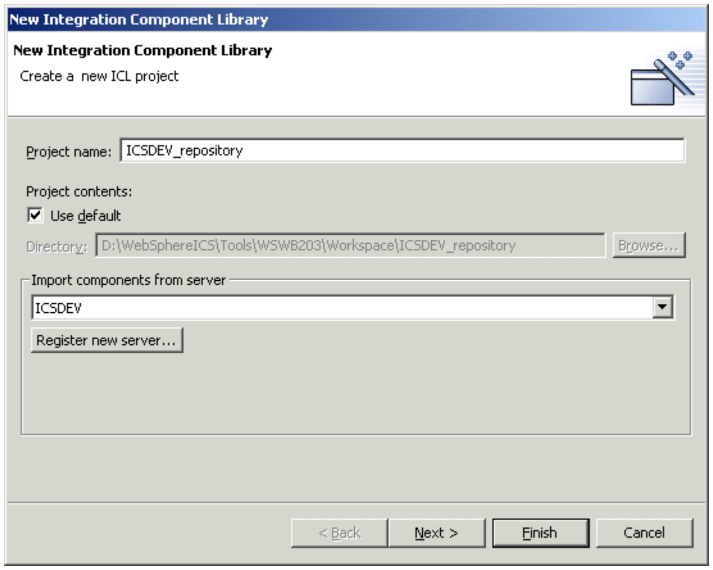
The next review task is editing the command files that start the connector. These are start_JDBC.bat and start_WebSphereMQWorkflow.bat. For the JDBC connector, add references to DB2 files and directories. For the WebSphere MQ Workflow connector, this is a reference to the WebSphere MQ Workflow JAR files that implement the WebSphere MQ Workflow API. This process was explained when we built the development environment in Chapter 7, “Sales order management in InterChange Server” on page 249.

Finally, the shortcuts that you use to launch the two connectors must contain a reference to the external configuration files. The standard installation of the connectors creates shortcuts that do not have this additional parameter. Refer to Chapter 7, “Sales order management in InterChange Server” on page 249 for the specific details.

After all of these changes are made, you can plan for the deployment of the solution. Make sure that the deployment of the new solution does not affect other solutions that are in production. A good example of an adverse impact is the following: Consider that the new solution contains a customized version of a generic business object. However, other solutions that use this same generic business object in an unchanged format or with different changes might be deployed. While changing generic business objects that are part of the product is not recommended, nothing prevents you from doing it. Its impact might only be noticed before deploying the new solution or, worse, after it has been deployed. One way to detect possible conflicts is to start the deployment of the new solution and then carefully review the output of the second step of the deployment wizard. At that time, the production InterChange Server lists all objects that will be overwritten if you continue with the deployment. Make sure that overwriting any object (business object, map, and so on) that appears in this list will not cause any impact.

If you find any objects that would be overwritten if the new solution were deployed, research what components rely on the current definition of that object. One way to research this is to create an ICL that contains all objects that are deployed in the InterChange Server:

1. Create an ICL called `ICSDEV_repository` (Figure 9-17) for example, and select **ICSDEV** as the server from which you want to copy the repository contents. Click **Next**.


New Integration Component Library
New Integration Component Library
Create a new ICL project

Project name: ICSDEV_repository

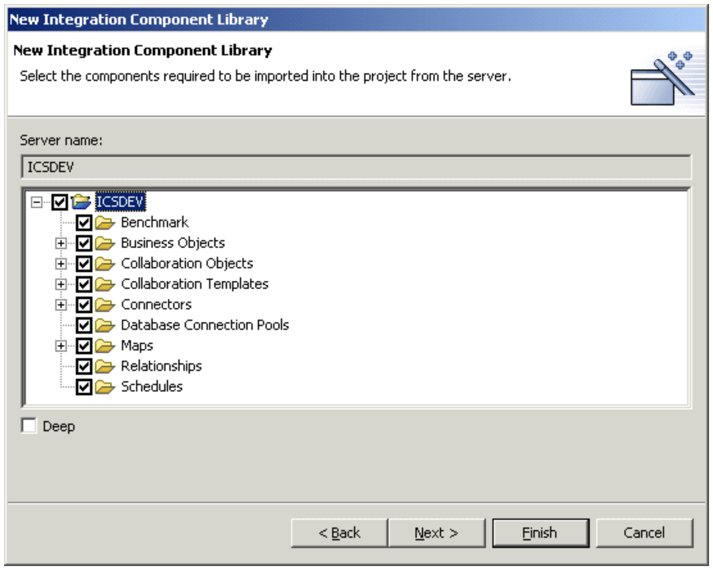
Project contents:
☑ Use default

Directory: D:\WebSphereICS\Tools\WSWB203\Workspace\ICSDEV_repository

Import components from server: ICSDEV

Figure 9-17 Create an ICL mirroring the repository

2. In the next step (Figure 9-18 on page 429), select all object types and click **Finish**.


New Integration Component Library

New Integration Component Library

Select the components required to be imported into the project from the server.

Server name:

ICSDEV

- [x] ICSDEV
  - [x] Benchmark
  - [x] Business Objects
  - [x] Collaboration Objects
  - [x] Collaboration Templates
  - [x] Connectors
  - [x] Database Connection Pools
  - [x] Maps
  - [x] Relationships
  - [x] Schedules

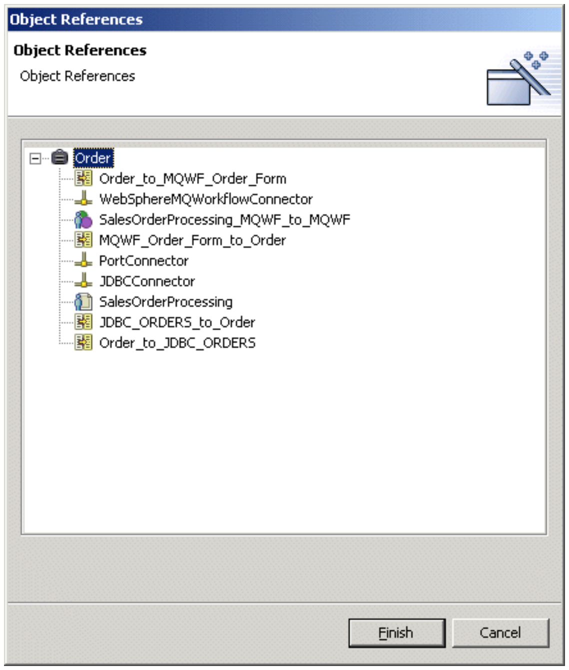
- [ ] Deep

Figure 9-18 Select objects to import

Attention: Be aware that this process might take significant time and resources to complete if your InterChange Server contains many objects. Make sure that your actions do not affect the ongoing processing in the production InterChange Server.

After all objects are imported, you can use the facilities of the System Manager to discern if a component is used by another component.

Assume for a moment that the predeployment test has revealed that the generic business object Order will be overwritten if the deployment is actually performed. Locate **Order** in the ICL ICSDEV_repository. Right-click it and select **Show References**. Figure 9-19 on page 430 shows the object references for the GBO Order in our environment. With this information, you can determine whether overwriting Order will affect any of the listed users of that object.



Order
- Order_to_MQWF_Order_Form
- WebSphereMQWorkflowConnector
- SalesOrderProcessing_MQWF_to_MQWF
- MQWF_Order_Form_to_Order
- PortConnector
- JDBCConnector
- SalesOrderProcessing
- JDBC_ORDERS_to_Order
- Order_to_JDBC_ORDERS

Figure 9-19 Find object references

Assuming that you worked out or cleared all possible conflicts between the deployed solutions and the to-be-deployed solution, you can plan for the activation of the new solution. Remember that adding or updating connectors to an existing InterChange Server results in restarting the InterChange Server before the changes become effective. Even more, components that have an active state, such as maps, collaborations, and connectors, can only be updated if these objects have been stopped before the deployment. This means that performing the deployment and activating the new solution must be planned carefully. A full impact analysis is required to avoid any disruption to existing integration solutions. An impact analysis should be performed with the idea that you have only one opportunity to do it right.

9.2 Managing runtime-specific changes

Today's business environment is changing at an ever-increasing pace. This is partly a consequence of more integration among the different actors in a business process. A few years ago, a bank customer who wanted to know account balances likely had to visit a branch or make a phone call to the customer service department. Nowadays, this same bank customer has plenty of facilities to interact directly with the bank's IT infrastructure to perform more than simply learn an account balance. Customers have become accustomed to these features, which are possible only because there is an integration infrastructure in place. Customer expectations can change rapidly, so the integration infrastructure must be ready for change. But also, the administrators of the integration infrastructure must be prepared for managing changes and they must understand what it means to perform changes in any component of a complex integration solution. In this section, we discuss changes that are limited to one runtime component. We do not consider changes to the interfaces between components, which we discuss in the next section.

9.2.1 Changes to WebSphere MQ Workflow process model

The life span of a WebSphere MQ Workflow process instance can range from seconds to years. By design, when a process instance is created and started, it will follow the process model at time of creation. In addition, all data is saved along the process path. Therefore, a simultaneous change in the process model has no affect on the running instances.

This design principle has several consequences. First, implementing a new version of a process model does not have any impact on existing and running process instances. For example, if you have deployed a credit approval process that typically runs for a few days between the initial request and the final approval, a change to the process model will only effect new credit requests. As a result, making changes during an existing process run could become problematic.

Another consequence is in discovering what process model is in use for a given running process instance. Consider a situation in which a process instance runs for a few months. An example is a complex insurance claim. If several changes are made to the process model during the lifetime of an active instance, it might become difficult to track down the source version of that running process instance. The WebSphere MQ Workflow administrator can track the identifier of a given process instance back to its original process model. This makes it easy for someone on a help desk to correlate a user inquiry with the corresponding version of the process flow.

Given that a process instance is tied to its original process model for as long as it is active, some modeling best-practices have been created to provide a way to implement new functionality that is also picked up by existing process instances. This technique recommends using a very simple top-level process in which all activities are subprocesses. The top-level process cannot be changed for the duration of the process, but changes to the subprocess will be picked up. As such, you create a process model that is flexible for changes as well as resilient.

9.2.2 Changes to a message flow

In this section we consider any change to a message flow that does not affect the input or output message or any other use of external resources. In the message broker, only one version of a message flow can be active at a given time. The assumption is that the execution of a message flow is short in time, typically in the range of a few milliseconds to a few seconds. The execution of a message flow is usually a single transaction, where we use the term transaction as a short-living atomic operation.

Deploying an updated message flow means that an active transaction can be completed and that a new transaction will use the updated message flow. Because we have made the assumption in this section that the changes do not affect the users of the message flow, deploying an updated message flow generally is not a difficult task to complete.

9.2.3 Changes to a collaboration

A change to a solution in the InterChange Server that does not have an impact on the outside world is *by definition* limited to changes to the collaboration and, possibly, the maps. Both types of objects can be updated only if they are stopped before the deployment. This means that any ongoing work is completed and new work remains queued. For both maps and collaborations, only one version of such an object can be deployed in the server at any point in time.

Starting with Version 4.2 of the InterChange Server, a collaboration can be long-running. Before that, the execution time for a collaboration was also in the range of a a few milliseconds to a few seconds; a long-running collaboration can run for a much longer time. Making changes to such a collaboration requires much more planning.

9.3 Managing interface changes

Managing changes that are limited to the internal logic of a runtime component is relatively easy. It becomes a lot more complex when the interface requires changes. In this section, we discuss several changes to the application scenario that we implemented for this book.

9.3.1 Changes to the data structure used to invoke the collaboration

Consider the Create Sales Order activity in our process model. This activity uses the Order_Form data structure, which is basically the interface between the process model and the collaboration.

The Order_Form data structure was used for several activities in the process model. Consequently, the change might be induced by another component and not necessarily by the InterChange Server or further downstream the Sales Order Management system.

In this case, it would be much better to develop a new data structure that is based on the existing Order_Form data structure and use the new data structure for activities that induced the change. Assume that the Order Parts activity requires more data that must be provided at the creation of the process instance. Because the Order_Form data structure acts as the input of the process and as the data structure for the Create Sales Order activity, the Create Sales Order activity is merely the victim of the change. To limit the impact, you could define a new data structure called Order_Form_Process and update the mapping in WebSphere MQ Workflow into the Create Sales Order activity. As such, the changed requirements for one activity is limited.

A more problematic situation involves changes to the Sales Order Management system that require changes to the process model and its data structures. Assume that a new data element is required for the Sales Order Management system. In the ideal case, this can be solved by changing the map that transforms the generic business object into the application-specific business object of the Sales Order Management system. Therefore, the impact of the change is limited. However, a change to the map might not be sufficient and this new data element might be provided only by the process instance that invokes the collaboration. In such a case, the change will propagate throughout the whole system, affecting maps, application-specific business objects, possibly the collaboration template, and the process model. Additionally, you cannot update a running process instance. Clearly, updating the integration solution is not an easy task.

To make this type of change more manageable, it could be easier to implement a new integration solution that runs in parallel with the old integration solution. Instead of updating the business objects, we can then create new business objects based on the old ones. To the outside world, it looks as though there is a process template and a collaboration template whose names have some kind of a version indication. By disabling the old template from creating new process instances, you can basically allow the existing instances to continue to use the old process model, the old interface to the InterChange Server, and the old collaboration in the InterChange Server. Implementing the new solution next to the old one will still have an impact on the WebSphere MQ Workflow connector. The connector's configuration has to be updated to support the new business objects. This again becomes a change that is manageable.

9.3.2 Changes to the data structure used to invoke the message flow

Consider the Order Parts activity in our process model. This activity uses the Parts_Replenishment_Form data structure, which acts as the interface between the process model and the message flow. This data structure is also used by the Approve Order process. Again, the first step in solving the problem is to understand the reason for the change.

Assume that the approval step requires more data than is provided by its input data structure. Adding an additional element to that data structure does not affect the processing in the message flow because the XML parses will simply ignore XML elements that it does not need. But it also means that the extra element will not flow back to the workflow server in the response message. This limitation is easily solved by appropriate mapping in the process model.

If changes are induced by changed requirements in the interaction with suppliers, then the change will affect the message flow. Because we are using self-defining XML, we do not have to make changes to message definitions in the broker. The change is limited to the ESQL in the Compute or to any other node that might rely on the specific structure of the messages.

Deploying a changed message flow is not a difficult task. However, it might again be difficult to synchronize between the process model and the message flow. A better solution might be to deploy the changed message flow under a new name and with different external resources such as queues. By doing so, the existing process instances can continue to use the old message flow, while new instances that use the new data structure can use the new message flow. Of course, this assumes that the interface with the suppliers is flexible enough to handle the old interface and the new interface in parallel.

9.4 Summary

Change management and deploying changed solutions is a very complex subject. The discussion in this chapter makes it clear that every change to a live system must be planned and analyzed. The impact of a change can be much deeper than originally thought. However, by performing some analysis, it is usually possible to limit the impact of the change.

10

Operational aspects of a WebSphere BI server implementation

Operating an integration infrastructure that spans several systems, possibly several platforms, and that consists of several components and layers on any given system can be a difficult task. By understanding the interaction and dependencies of the components, you can avoid frustration and maintain an operational system that satisfies the business requirements.

10.1 Starting and stopping components

The integration infrastructure in this book consists of six main runtime components:

- An IBM DB2 UDB Server
- A WebSphere Business Integration Message Broker
- A WebSphere InterChange Server
- A WebSphere MQ Workflow Server
- Two WebSphere Application Servers
- A Tivoli® Directory Server

In addition to the runtime servers listed here, there is also a locally installed WebSphere MQ server with each component, except for the Tivoli Directory Server. Two additional components, the WebSphere Adapter for JDBC and the WebSphere Adapter for WebSphere MQ Workflow, were required to be used as the communication bridge with the WebSphere Interchange Server. The WebSphere Adapter for JDBC bridges between the WebSphere InterChange Server and the application database. The WebSphere Adapter for WebSphere MQ Workflow bridges between the WebSphere MQ Workflow Server and the WebSphere InterChange Server.

Except for the database server itself, each of these runtime components have dependencies on their base components. The WebSphere Business Integration Message Broker, the WebSphere InterChange Server, and the WebSphere MQ Workflow Server rely on the availability of their repository database and their queue manager. In fact, the database system and the queue managers are so crucial that you should avoid stopping them unless the whole system is being restarted.

The WebSphere MQ Workflow Web Client can only be used if the WebSphere MQ Workflow server is up and running and if MQ communication is active between the two servers. However, you can start the Web server and the WebSphere MQ Workflow server independently from each other. They only start communicating with each other when a user logs on and requests services from the WebSphere MQ Workflow server.

Besides WebSphere MQ and DB2, the WebSphere InterChange Server component of the overall solution consists of four components:

- The Persistent and Transient Name Servers as the Object Resource Broker
- The WebSphere InterChange Server itself
- The two WebSphere BI Adapters for JDBC and WebSphere MQ Workflow
- The optional System Monitor component

The availability of the Persistent Name Server is required before starting (or using, in the case of the System Monitor) the other components. The Persistent Name Server will automatically start the Transient Name Server (the IBM Object Resource Broker). After the name server is started, the next component to start is the WebSphere InterChange Server. The two adapters can then be started, that will connect to the WebSphere InterChange Server.

After stopping or restarting the WebSphere InterChange Server, you do not have to restart the adapters, unless they are the components that you want to update. When the adapters detect that the WebSphere InterChange Server is unavailable they automatically go into a paused state until they are able to reestablish communication with the server.

The System Monitor Web application is installed in an application server called ICSMonitor under the WebSphere Application Server and can be started using the **`startServer ICSMonitor`** command on the WebSphere Application Server machine. No interaction with the Transient Name Server or the WebSphere InterChange Server will take place before a user has logged on to the Web application.

Starting adapter agents that are configured for the MQ or IDL transports requires that the name server and that the WebSphere InterChange Server are running. If the transport is configured for JMS, the name server is not required for adapter communication. To start an adapter, you can use the shortcut that we created in Chapter 7, “Sales order management in InterChange Server” on page 249. This means that you must log on to the system to start the adapter agents. However, alternatives exist:

- Within the System Manager on the management client machine or within the Web application System Monitor, you can stop and start adapter controllers and agent. The feature is called the Object Activation Daemon and it relies on WebSphere MQ and MQ triggering.
- When installing the adapters on Microsoft Windows, the user is given an option to create a Window’s service for administering the start and stop of the adapter.
- On UNIX and Linux, scripts and cron jobs often used to automatically start and stop adapters.

To configure the Object Activation daemon feature, execute these steps:

1. Enable the MQ Trigger Monitor by including it in the IBM MQSeries services application.
2. Create the following WebSphere MQ objects:
   - An initiation queue

   – A process definition, which contains the name and full path to the adapter start-up script
   – An adapter activation queue, which will receive the request message to start the connector
3. Update the adapter configuration.

A script called mqtriggersetup is provided with the product to assist in defining those objects. It creates the queue INITIATION.QUEUE, creates a process called PROCESS.*%ADAPTERNAME%*.TRIGGER and an adapter activation queue AGENTACTIVATIONQUEUE/*%ADAPTERNAME%*CONNECTOR, in which *%ADAPTERNAME%* is substituted with the name of the adapter. Because the name of the queue has a fixed format and the length of a queue name is limited to 48 characters, ensure that adapter names are shorter than 18 characters.

The mqtriggersetup script has four parameters:

- The name of the WebSphere InterChange Server queue manager
- The name of the adapter
- The name and full path to the adapter's startup script
- The name of the WebSphere InterChange Server instance

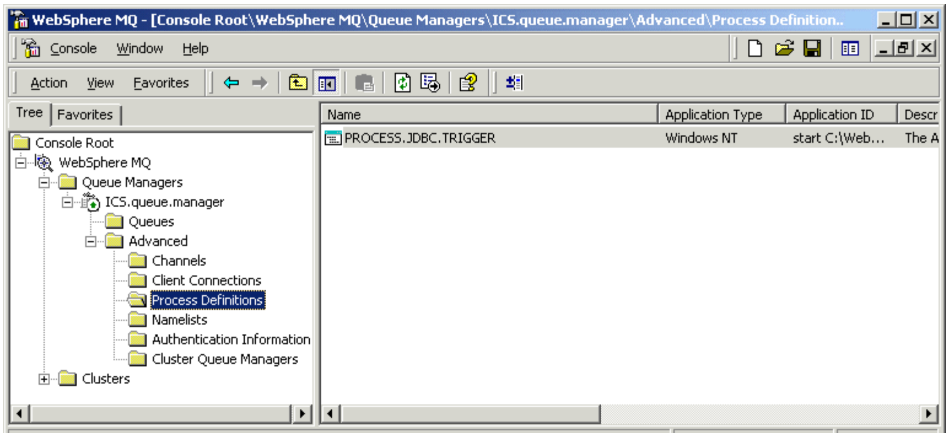
Example 10-1 shows the use of this script for the WebSphere Adapter for JDBC.

Example 10-1 Execute the mqtriggersetup script

```
C:\WebSphereAdapters\bin>mqtriggersetup ICS.queue.manager WebSphereMQWorkflow
C:\WebSphereAdapters\connectors\WebSphereMQWorkflow\start_WebSphereMQWorkflow.b
at ICS
```

The script assumes that the adapter can be started without a configuration file, which is possible when using IDL as the adapter transport. When using JMS as the adapter transport there is a requirement to use a configuration file in the adapter startup. You can either update the script to build a process definition that includes the name of the configuration file, or you can use WebSphere MQ Explorer to adjust the process definition.

1. In WebSphere MQ Explorer, expand the tree structure in the left pane (Figure 10-1 on page 439) to locate the Process Definitions folder for the WebSphere InterChange Server queue manager. The right pane contains the process definition that the script created.



Name	Application Type	Application ID	Descr
PROCESS.JDBC.TRIGGER	Windows NT	start C:\Web...	The A

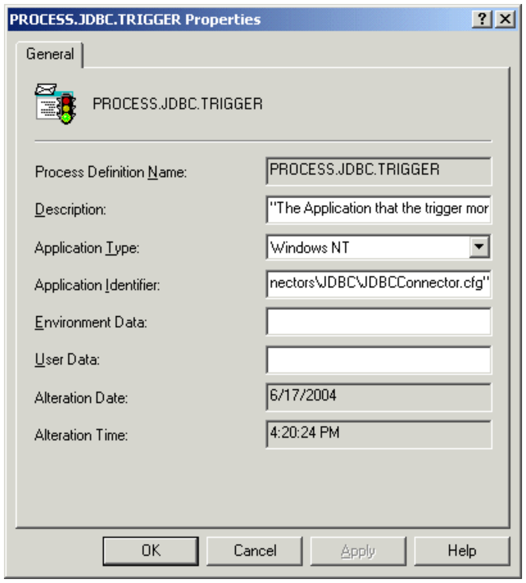
Figure 10-1 List of process definitions

2. Open the properties of the process definition for the WebSphere Adapter for JDBC and update the Application Identifier field. Add the parameter `-c` followed by the name and full path where the configuration file is stored.

   Example 10-2 shows the contents of the Application Identifier field shown in Figure 10-2 on page 440.

Example 10-2 Application Identifier in process definition

```
"start C:\WebSphereAdapters\connectors\JDBC\start_JDBC.bat JDBC DEV
-cC:\WebSphereAdapters\connectors\JDBC\JDBCConnector.cfg"
```


PROCESS.JDBC.TRIGGER Properties

PROCESS.JDBC.TRIGGER

Process Definition Name:	PROCESS.JDBC.TRIGGER
Description:	"The Application that the trigger mor
Application Type:	Windows NT
Application Identifier:	nectors\JDBC\JDBCConnector.cfg"
Environment Data:	
User Data:	
Alteration Date:	6/17/2004
Alteration Time:	4:20:24 PM

Figure 10-2 Update process definition

To enable the MQ Trigger Monitor, start the IBM MQSeries services application:

1. Right-click the queue manager that is used by the InterChange Server and select **New** → **Trigger Monitor**.
2. In the Create Trigger Monitor Service window, provide the name of the initiation queue, `INITIATION.QUEUE`, as shown in Figure 10-3 on page 442, and click **OK**.
3. In WebSphere MQ Services, right-click the new trigger monitor and start it.

   An alternative way to create the trigger monitor is to use this command:

   ```
   amqmdain crttrm ICS.queue.manager INITIATION.QUEUE auto
   ```

   Note, however, that this will not start the trigger monitor.
4. Update the connector configuration. On the **Standard Properties** tab for the WebSphere Adapter for JDBC, set the OADAutoRestartAgent property to `true`. Save this change and deploy it to the WebSphere InterChange Server.

   To verify that the configuration is correct, you can, for example, run the Trigger Monitor as a foreground program instead of as a background service. Use the following command to start the Trigger Monitor in a command window:

   ```
   runmqtrm -q INITIATION.QUEUE -m ICS.queue.manager
   ```

5. Switch to the System Manager and select the JDBC connector in the Component Management view. Right-click the connector and select **Boot JDBC Connector.**

Alternatively, you can use the System Monitor Web application to perform this action.

10.2 Management and problem determination tools

Solving problems efficiently depends on the information and tools that are made available by the systems and the applications. Of course, knowing how to use the information and the tools is also a requirement. From an application and system-design perspective, a balance must be made between providing a lot of information and the usability of that information and actual performance. A constant detailed tracing function might slow down the system significantly. The option to provide additional information on request is a great advantage.

10.2.1 Sources of information

All components of a WebSphere Business Integration Server solution have several tools available to assist in managing the system and solving problems.

WebSphere MQ Workflow

When administering a WebSphere MQ Workflow solution, a distinction must be made between managing process instances and managing the system itself. Managing process instances means the facility to start, stop, resume, suspend, or query an active process instance. These options are available through the WebSphere MQ Workflow Web Client. At the activity level, the Web client feature enables the process administrator to restart or finish an activity. Also, he can transfer an activity from one person to another.

After logging on to the Web Client, select the list of process instances. When the list opens, click the **camera** icon to open the process monitor. Figure 10-3 on page 442, the monitor view of an active process, shows which activities have been completed and which activity is in the active state. Activities that are marked with a red arrow are not yet ready to execute.

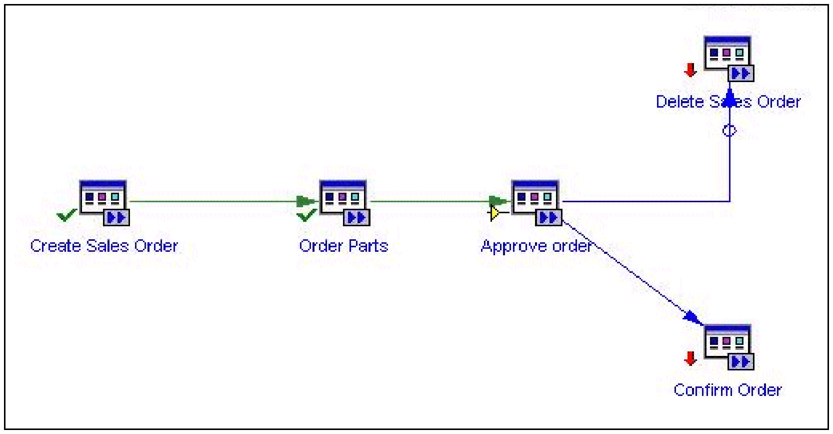
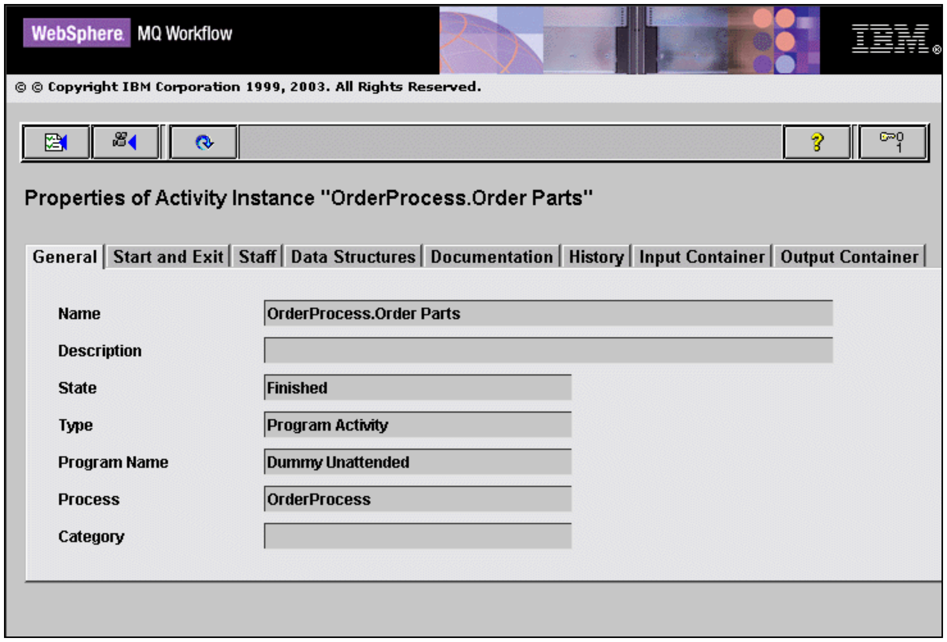


Figure 10-3 Overview of the process monitor

Every activity in Figure 10-3 is a hot link to more details about it. Figure 10-4 shows the details of the completed activity Order Parts. On the **General** tab, you can see that this activity is finished. The **Input** Container and **Output** Container tabs show the data structures that have been passed to this activity and that were returned by this activity when it completed. Other available information includes the stop time and start time of the activity.

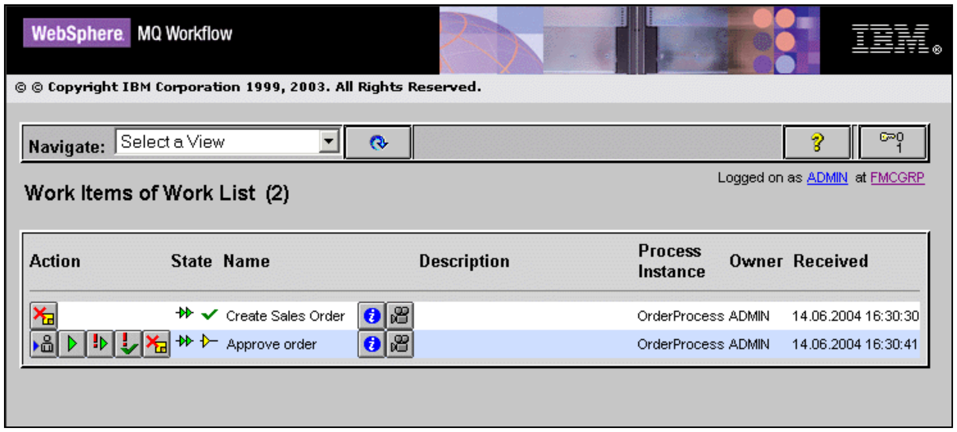

WebSphere MQ Workflow
IBM
© © Copyright IBM Corporation 1999, 2003. All Rights Reserved.
General | Start and Exit | Staff | Data Structures | Documentation | History | Input Container | Output Container

Properties of Activity Instance "OrderProcess.Order Parts"

Name	OrderProcess.Order Parts
Description	
State	Finished
Type	Program Activity
Program Name	Dummy Unattended
Process	OrderProcess
Category	

Figure 10-4 Detailed view for an activity

When selecting the Work Items list, you can see what activities are assigned to you. When logged on as the process administrator, you can see all activities that are on an indivudual's work list. Figure 10-5 shows a list with two activities: a completed activity and, the last one, an activity in a ready state. For that activity, the owner can check it out to complete it, transfer it to another user, or restart it.



© © Copyright IBM Corporation 1999, 2003. All Rights Reserved.

Logged on as ADMIN at FMCGRP

Work Items of Work List (2)

Action	State	Name	Description	Process Instance	Owner	Received
	✓	Create Sales Order		OrderProcess	ADMIN	14.06.2004 16:30:30
		Approve order		OrderProcess	ADMIN	14.06.2004 16:30:41

Figure 10-5 Reviewing the work list

To find information about the actual server, have the WebSphere MQ Workflow Administration component installed. We have installed it on the management client machine and also on the actual server. To start the administration utility, select **Start** → **Programs** → **IBM WebSphere MQ Workflow** → **WebSphere MQ Workflow Administration Utility - FMC**.

After logging on, you are presented with the main menu (Figure 10-6).

```
-  FMC16006I Administration Utility started.
     System group name   : [FMCGRP] FMCGRP
     System name         : [FMCSYS] FMCSYS
     Userid              : [ADMIN] ADMIN
     Password            : [] ********
=  FMC16110I Receive thread for userID 'ADMIN' at system 'FMCSYS' started.
-  FMC16301I UserID 'ADMIN' connected to system 'FMCSYS'.
   FMC15010I Main Menu:
     s ... System Commands Menu
     m ... Select Server Menu
     e ... Errorlog Commands Menu
     l ... Systemlog Commands Menu
     u ... User Commands Menu
     x ... Exit Main Menu
```

Figure 10-6 Main menu of the administration utility

On Windows, the information that is available through the Systemlog Commands Menu option is also available in the Windows Event Viewer. Besides reviewing the contents of the system log, this menu option enables you to control the system log. For example, you can clear the contents or change the retention period for log entries. Note that these settings can also be changed in Buildtime, but those changes must be exported and imported before they become active. Using the administration utility, you can make those changes directly to the runtime server.

Figure 10-7 shows the System Commands menu, with which you can see what server components are running. You can also stop the server or connect to a different server, if the WebSphere MQ Workflow system group contains more than one system.

```
FMC15040I System Commands Menu:
     c ... Connect
     i ... Info
     d ... Shutdown
     q ... Query
     w ... Wait
     x ... Exit System Commands Menu
q
-  FMC16220I Administration Server is 'active'.
-  FMC16220I Cleanup Server is 'inactive'.
-  FMC16221I Execution Server is 'active' (2 instance(s) running).
-  FMC16220I Scheduling Server is 'inactive'.
```

Figure 10-7 System Commands Menu

Figure 10-7 also shows the current status of the system. The administration server is active, as are two execution servers. The execution server is the actual process that navigates through the process model and assigns activities to users. The cleanup and scheduling server are not active. Note that the scheduling server is required if you use any time-based settings in a process model. An example of a timing-based activity could be the Approve Order activity. You could, for example, model the process such that the approver has to act on a work item within two hours. It is then the task of the scheduling server to monitor that this activity is indeed completed within two hours and if it is not, the scheduling server will perform the appropriate action, such as notifying the process administrator.

Figure 10-8 on page 445 shows the Server Menu, which contains options for the different server components. For each server component, you can query its current status, stop or start it, and obtain runtime parameters. To change those parameters, review the properties in Buildtime on the Network tab.

```
m
   FMC15050I Select Server Menu:
     a ... Administration Server Commands Menu
     e ... Execution Server Commands Menu
     s ... Scheduling Server Commands Menu
     c ... Cleanup Server Commands Menu
     x ... Exit Select Server Menu
a
   FMC15051I Administration Server Commands Menu:
     i ... Info
     d ... Shutdown
     q ... Query
     w ... Wait
     x ... Exit Administration Server Commands Menu
```

Figure 10-8 Server Menu and Administration Server Menu options

Figure 10-9 shows the error log, which can contain runtime errors specific to a process instance or overall runtime errors. An example of an error that is specific to a process instance is a badly constructed XML response message. When the runtime engine reads an XML message from its input queue, any error for that XML message is reported to the error log.

Figure 10-9 shows an error log entry that is related to a communication problem between the runtime engine and the database server.

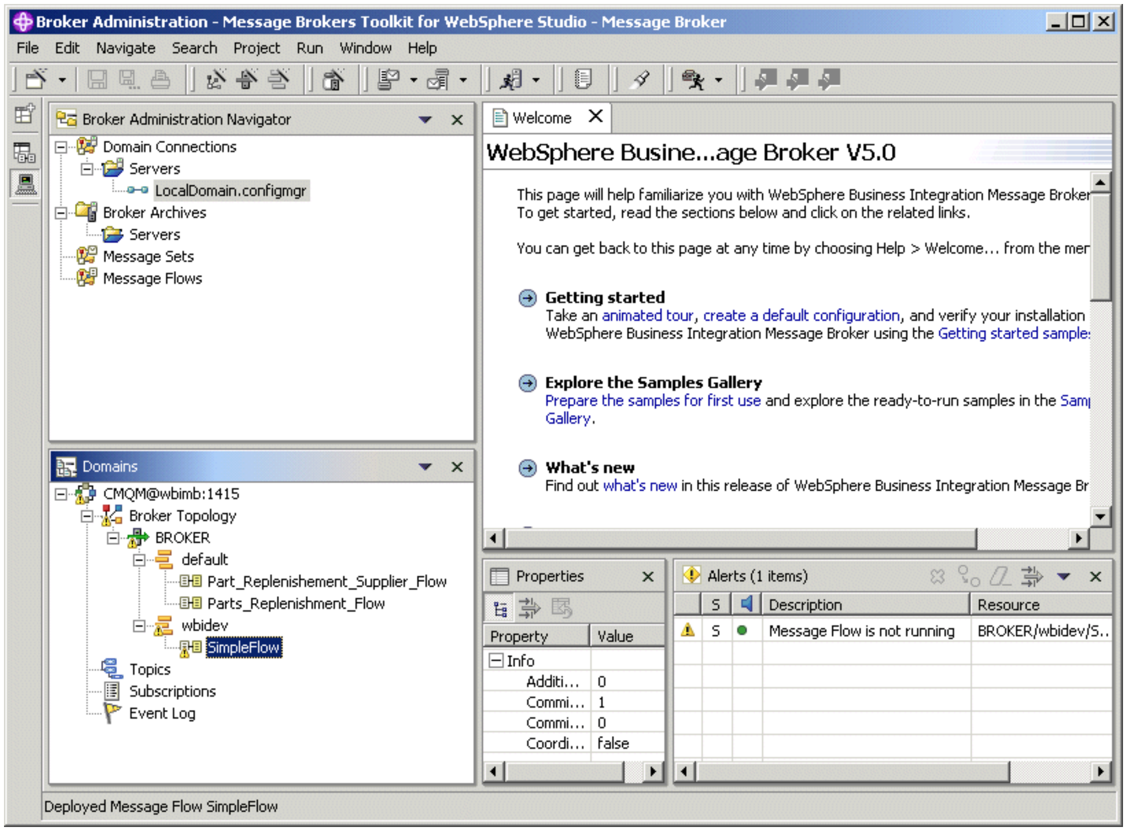
```
 FMC15010I Main Menu:
     s ... System Commands Menu
     m ... Select Server Menu
     e ... Errorlog Commands Menu
     l ... Systemlog Commands Menu
     u ... User Commands Menu
     x ... Exit Main Menu
e
   FMC15060I Errorlog Commands Menu:
     i ... Info
     l ... List
     p ... Purge
     x ... Exit Errorlog Commands Menu
l
-  6/14/2004 3:25:17 PM FmcSQLException, Sqlcode=-30081,
Sqlerrmc=10054ÿ*ÿ0ÿTCP/IPÿSOCKETSÿ192.168.192.136ÿrecvÿ, Sqlerrp=SQLJCMN ,
Sqlerrd[0]=-2127167470, Sqlerrd[1]=18, Sqlerrd[2]=0, Sqlerrd[3]=0,
Sqlerrd[4]=0, Sqlerrd[5]=0, Sqlwarn= , Sqlstate=08001
```

Figure 10-9 Errorlog Commands Menu

WebSphere Business Integration Message Broker

To manage the broker and find actual status information about the broker, its execution groups, and the message flows, you can use the Broker Toolkit. Within the Broker Administration perspective, the Domains view provides information about what is deployed and running. From here, you can stop and start message flows as well.

The Alerts view lists messages about resources that are not running, whether intentionally or as the result of an error condition. Figure 10-10 shows that the SimpleFlow message flow is not running.



WebSphere Busine...age Broker V5.0

This page will help familiarize you with WebSphere Business Integration Message Broker
To get started, read the sections below and click on the related links.

You can get back to this page at any time by choosing Help > Welcome... from the mer

Getting started
Take an animated tour, create a default configuration, and verify your installation
WebSphere Business Integration Message Broker using the Getting started sample:

Explore the Samples Gallery
Prepare the samples for first use and explore the ready-to-run samples in the Sam
Gallery.

What's new
Find out what's new in this release of WebSphere Business Integration Message Br

Property	Value
Info	
Additi...	0
Commi...	1
Commi...	0
Coordi...	false

	S		Description	Resource
⚠	S	●	Message Flow is not running	BROKER/wbidev/S..

Deployed Message Flow SimpleFlow

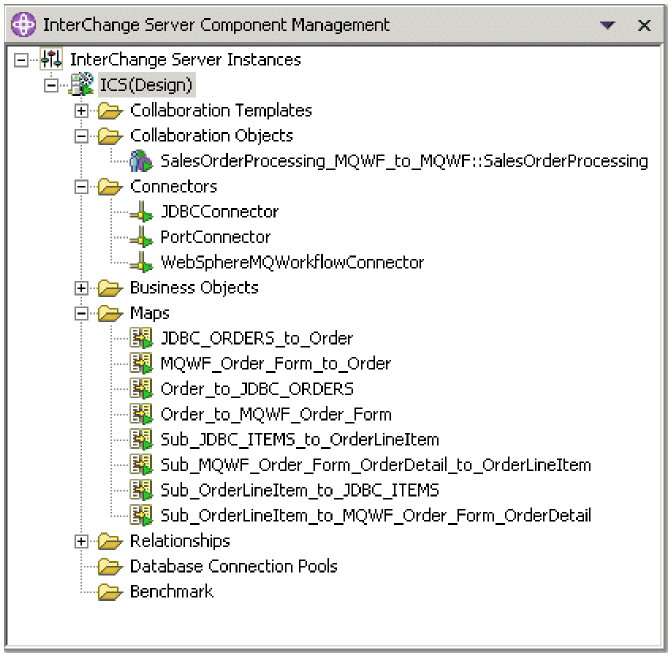
Figure 10-10 Broker administration tools in the Broker Toolkit

Runtime errors are by default reported to an operating system–specific logging infrastructure. On Windows, this is the Event Viewer. On UNIX systems, this is the system logger daemon. For z/OS® brokers, errors are reported to the system logger. Usually, for a runtime problem, the messages in those logging facilities provide quite a bit of information about the nature of the problem.

WebSphere InterChange Server

Runtime information is available through two facilities: a Web-based System Monitor and the Component Management view in the System Manager. Both tools offer more or less the same functionality; the main difference is the reach of the tool. The Web-based System Monitor can be used on any system with a browser. To use the System Manager, you must be on a system that has System Manager installed.

Figure 10-11 shows the Component Management view in System Manager. This view gives an immediate report of the status of collaborations, connector controllers, and maps. The context menu for each of these options provides more details about what is going on in the system.

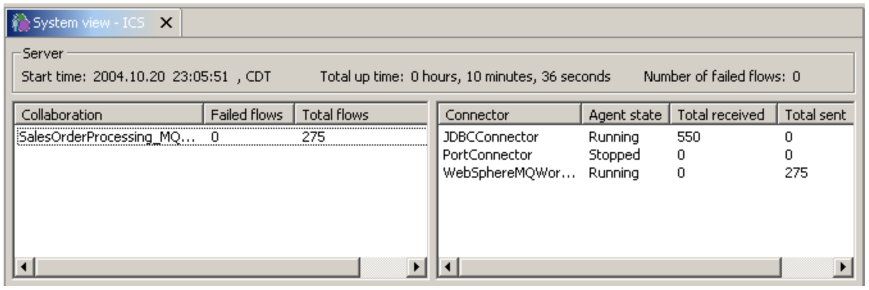


InterChange Server Component Management

- InterChange Server Instances
  - ICS(Design)
    - Collaboration Templates
    - Collaboration Objects
      - SalesOrderProcessing_MQWF_to_MQWF::SalesOrderProcessing
    - Connectors
      - JDBCConnector
      - PortConnector
      - WebSphereMQWorkflowConnector
    - Business Objects
    - Maps
      - JDBC_ORDERS_to_Order
      - MQWF_Order_Form_to_Order
      - Order_to_JDBC_ORDERS
      - Order_to_MQWF_Order_Form
      - Sub_JDBC_ITEMS_to_OrderLineItem
      - Sub_MQWF_Order_Form_OrderDetail_to_OrderLineItem
      - Sub_OrderLineItem_to_JDBC_ITEMS
      - Sub_OrderLineItem_to_MQWF_Order_Form_OrderDetail
    - Relationships
    - Database Connection Pools
    - Benchmark

Figure 10-11 Component Management view in System Manager

Right-click the **InterChange Server** instance and select **System View**, shown in Figure 10-12 on page 448. This view provides real-time status of the connectors and the collaborations, and shows which connector agents are running and how many business objects they have processed.



Server

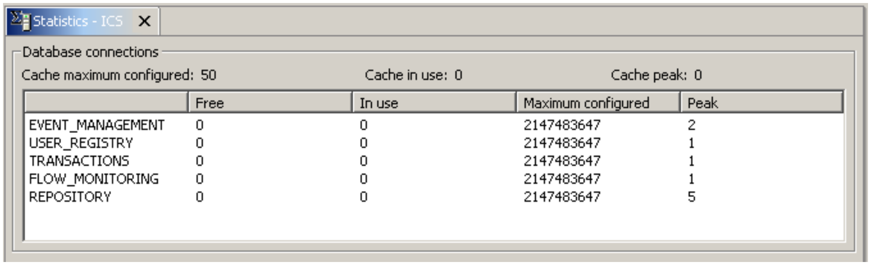
Start time: 2004.10.20 23:05:51 , CDT   Total up time: 0 hours, 10 minutes, 36 seconds   Number of failed flows: 0

Collaboration	Failed flows	Total flows
SalesOrderProcessing_MQ...	0	275

Connector	Agent state	Total received	Total sent
JDBCConnector	Running	550	0
PortConnector	Stopped	0	0
WebSphereMQWor...	Running	0	275

Figure 10-12 System View

Right-click the instance again and select **System Statistics** (Figure 10-13). This view provides information about system-level database services of the InterChange Server.



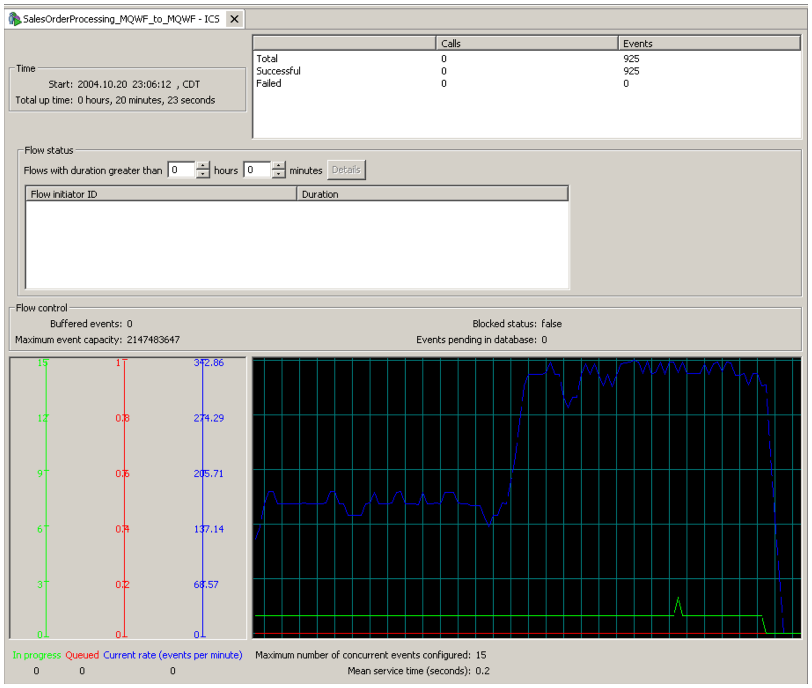
Database connections

Cache maximum configured: 50   Cache in use: 0   Cache peak: 0

	Free	In use	Maximum configured	Peak
EVENT_MANAGEMENT	0	0	2147483647	2
USER_REGISTRY	0	0	2147483647	1
TRANSACTIONS	0	0	2147483647	1
FLOW_MONITORING	0	0	2147483647	1
REPOSITORY	0	0	2147483647	5

Figure 10-13 System Statistics

Right-click the collaboration and select **Statistics** for the view shown in Figure 10-14 on page 449. It provides information about processed service calls and events but also about queued events. For any active collaborations, it provides the elapsed time of execution for that collaboration and the originator.



SalesOrderProcessing_MQWF_to_MQWF - ICS

Time
Start: 2004.10.20 23:06:12 , CDT
Total up time: 0 hours, 20 minutes, 23 seconds

	Calls	Events
Total	0	925
Successful	0	925
Failed	0	0

Flow status
Flows with duration greater than 0 hours 0 minutes Details

Flow initiator ID	Duration

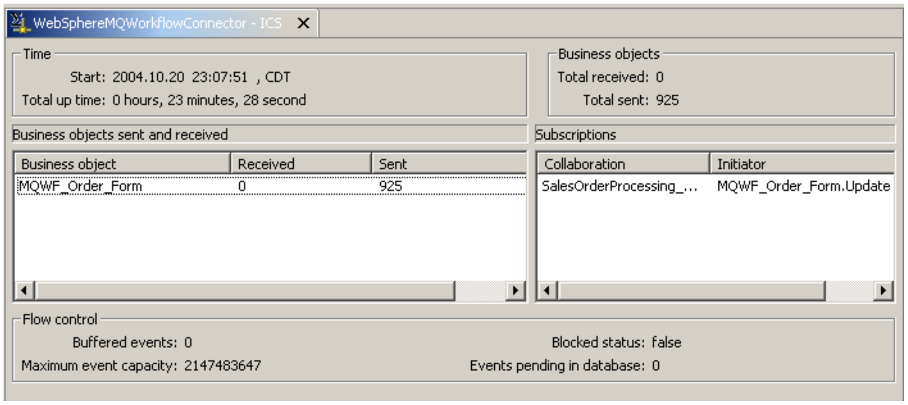
Flow control
Buffered events: 0
Blocked status: false
Maximum event capacity: 2147483647
Events pending in database: 0

In progress	Queued	Current rate (events per minute)
0	0	0

Maximum number of concurrent events configured: 15
Mean service time (seconds): 0.2

Figure 10-14 Collaboration statistics

Statistics are also available for each connector, as shown in Figure 10-15. This view provides information about sent and received business objects and what collaboration has subscribed to these business objects.



WebSphereMQWorkflowConnector - ICS

Time
Start: 2004.10.20 23:07:51 , CDT
Total up time: 0 hours, 23 minutes, 28 second

Business objects
Total received: 0
Total sent: 925

Business objects sent and received

Business object	Received	Sent
MQWF_Order_Form	0	925

Subscriptions

Collaboration	Initiator
SalesOrderProcessing_...	MQWF_Order_Form.Update

Flow control
Buffered events: 0
Blocked status: false
Maximum event capacity: 2147483647
Events pending in database: 0

Figure 10-15 Adapter statistics

WebSphere Application Server

WebSphere Application Server uses a Web-based administration tool, called the Administrative Console, to manage the server. The Administrative Console can be used only if WebSphere Application Server itself is started. To be more correct, start the application server called **server1** to operate the Administrative Console. However, you can use this application to manage other servers, including the ICSMonitor server.

To solve problems related to WebSphere Application Server, the server logs are the first place to look. These server logs are accessible through the Administrative Console, but usually people prefer to open the log files directly if they have access to the server that hosts WebSphere Application Server. These log files are located in these directories:

```
\WebSphere\Application Server\logs\<server name>
```

Two files are important: SystemOut.txt and SystemError.txt.

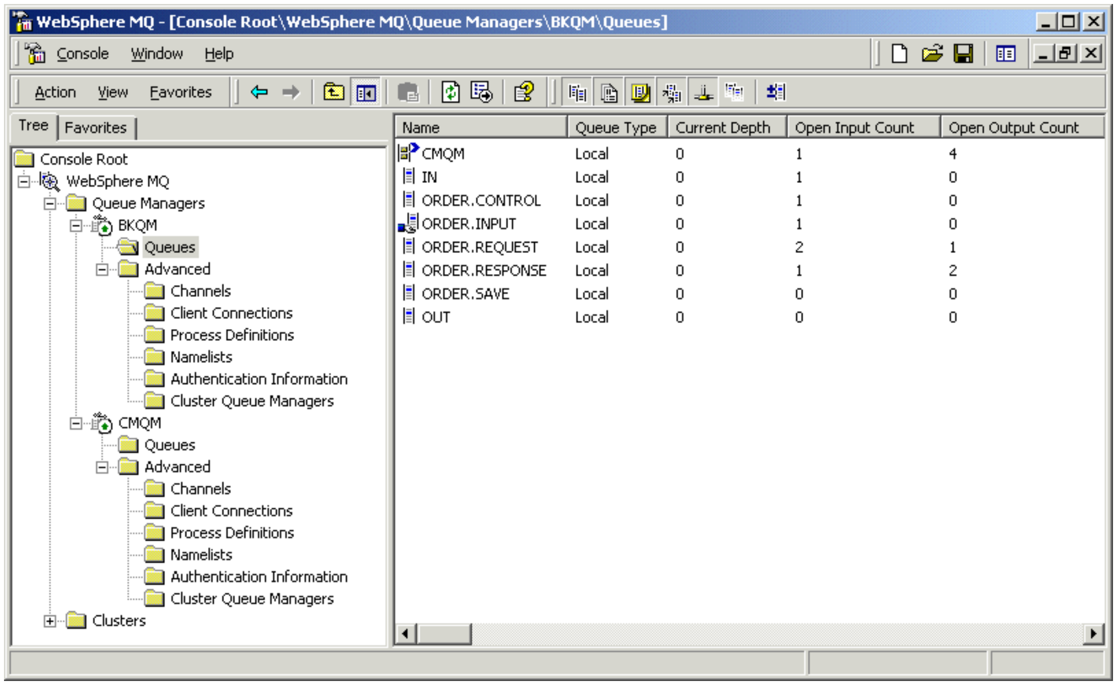
WebSphere MQ

On Windows, WebSphere MQ provides two graphical tools to interact with queue managers and their resources.

One tool, WebSphere MQ Explorer, can be used to manage objects, alter objects, stop and start channels, review queue depth, and browse the actual messages. It can also be used to discover who is using a certain queue. While it is a Windows-specific tool, it can be used to interact with queue managers in other platforms such as UNIX.

Figure 10-16 on page 451 shows the nonsystem queues for the BKQM queue manager. The right pane is customized to show the most important properties:

- Name
- Current depth
- Open input count (how many threads or applications have opened this queue to retrieve messages)
- Open output count (how many threads or applications have opened this queue to write messages)

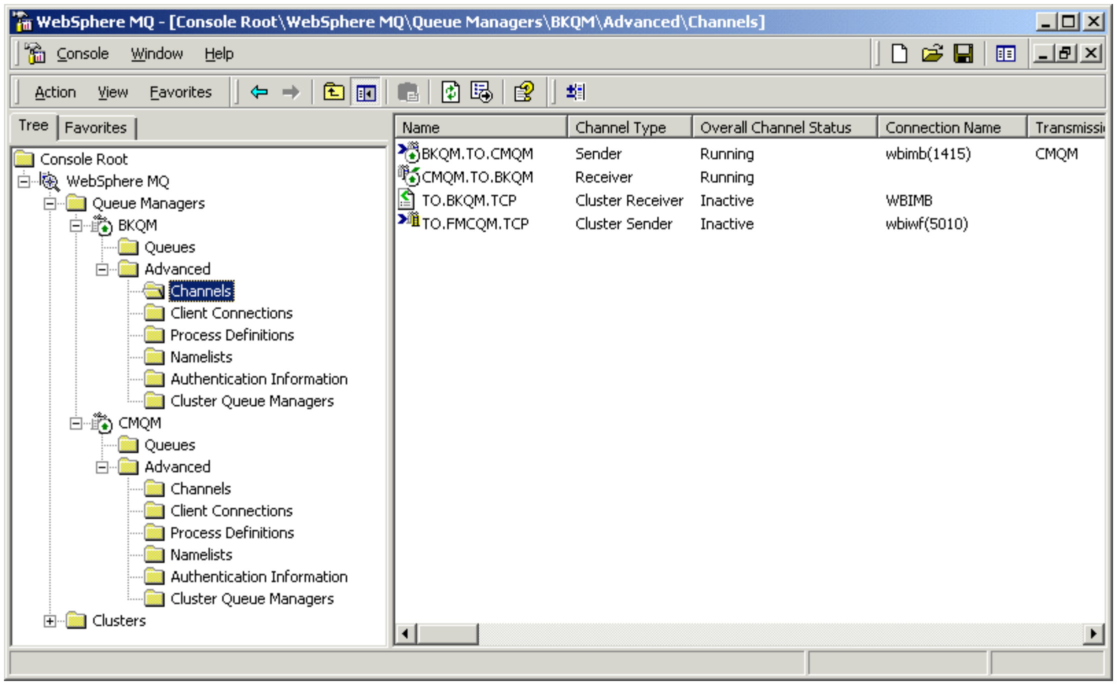


Name	Queue Type	Current Depth	Open Input Count	Open Output Count
CMQM	Local	0	1	4
IN	Local	0	1	0
ORDER.CONTROL	Local	0	1	0
ORDER.INPUT	Local	0	1	0
ORDER.REQUEST	Local	0	2	1
ORDER.RESPONSE	Local	0	1	2
ORDER.SAVE	Local	0	0	0
OUT	Local	0	0	0

Figure 10-16 List of queues in WebSphere MQ Explorer

To customize the right pane, right-click the Queues folder in the left pane and select **View** → **Select Columns**. You can then select which columns to show and their order of appearance. By double-clicking a queue, you can display the list of queued messages and to browse a portion of each message.

Figure 10-17 shows the list of non-system channels for this same queue manager. The list and order of displayed columns can be altered.



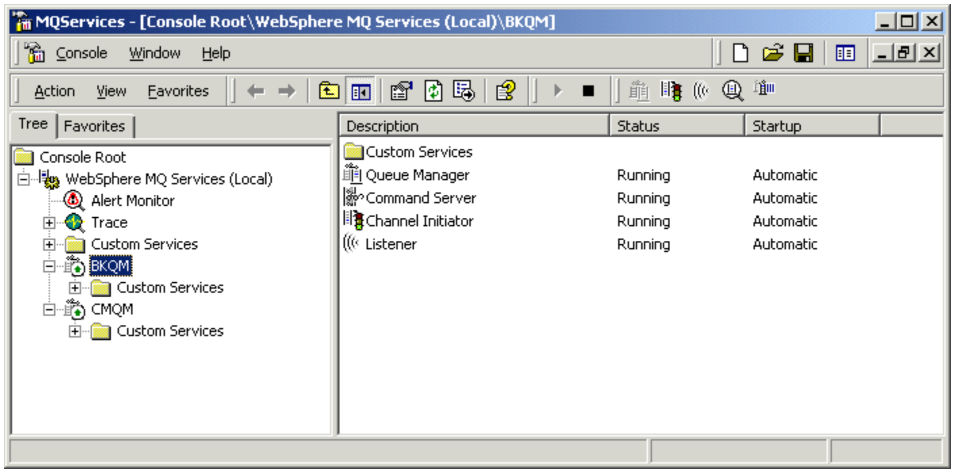
Name	Channel Type	Overall Channel Status	Connection Name	Transmissi
BKQM.TO.CMQM	Sender	Running	wbimb(1415)	CMQM
CMQM.TO.BKQM	Receiver	Running		
TO.BKQM.TCP	Cluster Receiver	Inactive	WBIMB	
TO.FMCQM.TCP	Cluster Sender	Inactive	wbiwf(5010)	

Figure 10-17 List of channels in WebSphere MQ Explorer

When using WebSphere MQ Explorer, always make sure that you see the actual state. The tool will cache objects and their state. If you think that an object status in the GUI is not consistent with its runtime state, perform a refresh.

The other tool, WebSphere MQ services, is used to manage the IBM MQSeries service on a Windows platform. IBM MQSeries service is actually called after the old name of the product. If you open the Windows Services application, you see a service called IBM MQSeries. This service does not map to a single queue manager. It can consist of several queue managers and related components such as listeners and channel initiators. To manage the contents of the IBM MQSeries service and to manage the state, use WebSphere MQ Services.

Figure 10-18 shows the components that are associated with the BKQM queue manager. Note that the start-up of each component is `Automatic`, which means that the component is started as soon as the Windows service IBM MQSeries is started. By using this application, you can of course stop and start components without being dependent on the state of the Windows service IBM MQSeries.



Description	Status	Startup
Custom Services		
Queue Manager	Running	Automatic
Command Server	Running	Automatic
Channel Initiator	Running	Automatic
Listener	Running	Automatic

Figure 10-18 WebSphere MQ Services application

WebSphere MQ can use platform-specific error reporting tools as well as its own facilities. On Windows platforms, the Event Viewer is a good source of information. Depending on the nature of the problem, additional information might be reported to log files called AMQERRxx.LOG. These log files are available in three different locations, depending on the type of problem and where it is being detected. These locations are:

- \WebSphere MQ\qmgrs\<name of queue manager>\errors
- \WebSphere MQ\qmgrs\@SYSTEM\errors
- \WebSphere MQ\errors

Critical errors also might result in an error report in the \WebSphere MQ\errors folder. These error files have the extension FDC. Often, understanding these error reports requires advanced knowledge of the product. They are often required when reporting errors to IBM.

System design and application design documentation

A source of information that is often forgotten during problem analysis is proper system design and application design information. Even more, very often this information is barely available. A simple diagram of the information flows with the names of the components can reduce problem resolution time significantly. If the integration infrastructure is managed by a group of people, it helps to include the names of the responsible people in that diagram.

10.2.2 Obtaining additional information

While all runtime components provide many tools and sources of information, sometimes you still need to dig deeper. In this section we briefly discuss the options that increase the level of information at runtime.

WebSphere MQ Workflow

When problems occur that cannot be solved with the information that is provided through the process monitor or the administration utility, you might have to activate tracing. Several types of tracing exist; all are activated using the fmczchk configuration checker utility and are configurable with system environment variables. Besides operating the trace component, this utility can be used to check the current configuration. This utility can also be used on client platforms.

Example 10-3 on page 455 shows the output of this utility when it is used on the WebSphere MQ Workflow server machine by the Windows Administrator user ID. It reveals, for example, that the configuration of the database server is insufficient. The agent stack size and the query heap size must be incremented.

Because we are using the user ID administrator to run the tool, and as this user ID is not authorized to connect to the remote runtime and buildtime databases, the utility reports errors when trying to access these databases. The user ID that is configured to access the runtime database in our environment is `wbiadmin`.

Example 10-3 Output of the configuration check utility

```
FMC34010I: Configuration checker version 3.5.0.116 started.
FMC34120I: Command-line options specified: -y FMC

FMC34012I: ===> General checks.
FMC34011I: Performing WebSphere MQ Workflow Version 3.5.0 checks.
FMC34101I: The Operating System is Windows 2000 5.0 (Service Pack 4).
FMC34103I: Local time is 2004-10-20 23:50:12.
FMC34104I: Universal time coordinated (UTC) is 2004-10-21 04:50:12.
FMC34301I: WinSock version: WinSock 2.0.
FMC34314I: The local IP address is 192.168.10.129.
FMC34131I: The WebSphere MQ Workflow installation directory is
c:\ibm\webspheremqwf\bin.
FMC34154I: WebSphere MQ Workflow Version 3.5.0.2 is installed.
FMC34143I: The default configuration is 'FMC' (from the general configuration
profile).
FMC34121I: The current configuration is 'FMC'.
FMC34145I: The following configurations are available: FMC.
FMC34113I: Language specification is 'ENU' (from the installation profile).
FMC34115I: The message catalog is c:\ibm\websph~4\bin\fmckmenu.cat.
FMC34132I: The XPG4 locale is EN_US.IBM-1252.
FMC34133I: The ANSI code page is 1252, the OEM code page is 437.
FMC34021W: The LOCPATH directory 'C:\IBM\LDAP\bin\locale' does not exist.
FMC34021W: The LOCPATH directory 'C:\IBM\LDAP\bin\locale' does not exist.
FMC34134I: The locale file used is
c:\ibm\websph~4\bin\locale\en_us\ibm-1252.lcl.

FMC34013I: ===> Admin Server found, checks started.
FMC34122I: Environment variable DB2_RR_TO_RS is set to 'YES'.
FMC34500I: ==> Database Manager configuration for DB2.
FMC34508I: Agent stack size is 96.
FMC34508I: Query heap size is 1500.
FMC34501I: ==> Database configuration for FMCDB.
FMC34508I: Log file size is 4095.
FMC34508I: Log primary is 4.
FMC34508I: Log second is 124.
FMC34508I: Database heap is 400.
FMC34508I: Default application heap is 512.
FMC34508I: Number of page cleaners is 2.
FMC34508I: Log buffer size is 16.
FMC34508I: Lock list is 1000.
FMC34502I: ==> Database connection to FMCDB.
FMC34508I: Buffer pool size is 2000.
FMC34515I: Connect to FMCDB was successful (user ID 'WBIADMIN').
FMC34510I: The DBMS name is DB2/NT (version 08.02.0000).
FMC34511I: The database driver is db2cli.dll (Version 08.02.0000).
FMC34512I: The database name for alias 'FMCDB' is 'fmcdb'.
FMC34700I: ==> Service Control Manager configuration.
```

```
FMC34nnnI: Service DB2NTSECSERVER: Startup automatic, account LocalSystem
FMC34nnnI: Service MQSeriesServices: Startup automatic, account LocalSystem
FMC34535I: Service 'DB2' does not exist, assuming DB2 client installation.
FMC34nnnI: Service MQ Workflow - FMC: Startup manual, account LocalSystem

FMC34013I: ===> Server message device found, checks started.
FMC34303I: TCP port fmclFMCQM5010 found at number 5010.
FMC34306I: The local listener for port 5010 is active.
FMC34031I: Successfully loaded TP Monitor library mqmax.dll.
FMC34404I: The MQSeries Server 5.3.0.7 is installed.
FMC34nnnI: System for FMCQM is FMC.FMCGRP.FMCSYS
FMC34031I: Successfully loaded QM switch file library db2swit.dll.
FMC34nnnI: Settings for FMCQM from Queue manager FMCQM registry
FMC34nnnI:   XAResourceManager.XAOpenString = DB=FMCDB, TPM=MQ, toc=p,
UID=wbiadmin, PWD=******
FMC34nnnI:   Channels.MaxChannels = <null>
FMC34nnnI:   Channels.MaxActiveChannels = <null>
FMC34nnnI: Queue Manager wics43.queue.manager not used by WebSphere MQ Workflow
FMC34nnnI: Queue Manager WBRK_QM not used by WebSphere MQ Workflow
FMC34nnnI: Default queue manager is wics43.queue.manager

FMC34013I: ===> Execution Server found, checks started.

FMC34013I: ===> Client message layer (server API) found, checks started.
FMC34nnnI: FMLConnectName is 'FMC.FMCGRP.FMCSYS,FMCQM'
FMC34nnnI: Channel definition file is c:\ibm\webspheremqwf\chltabs\mqwfchl.tab
FMC34410I: Channel table c:\ibm\webspheremqwf\chltabs\mqwfchl.tab has 1
entries.
FMC34411I:   0: Active; Queue Manager FMCQM connecting to WBIServer(5010)
through TCP.
FMC34nnnI: This client can connect to an OS/390 server

FMC34013I: ===> Client message layer (client API) found, checks started.

FMC34013I: ===> WebSphere MQ Workflow C/C++ API found, checks started.
FMC34nnnI: API will use queue manager 'FMCQM' (queue prefix FMC)
FMC34324I: Ping WBIServer successful.

FMC34013I: ===> Java Agent found, checks started.
FMC34nnnI: Agent.Reaper.Cycle is 300000ms
FMC34nnnI: Agent.Reaper.Threshold is 1000 objects
FMC34nnnI: Agent.Reaper.Ratio is 90%

FMC34013I: ===> Buildtime found, checks started.
FMC34130I: The default font for the graphical user interface is 'MS Shell Dlg'.
FMC34129I: The screen resolution is set to 1400x1050.
FMC34nnnI: B/T help file is c:\ibm\webspheremqwf\bin\fmcbhenu.hlp
FMC34nnnI: HTML template file is c:\ibm\webspheremqwf\bin\fmcfdhtm.tem
FMC34031I: Successfully loaded BT resource library fmcbrenu.dll.
```

```
FMC34502I: ==> Database connection to FMCBTDB.
FMC34515I: Connect to FMCBTDB was successful (user ID 'WBIADMIN').
FMC34510I: The DBMS name is DB2/NT (version 08.02.0000).
FMC34511I: The database driver is db2cli.dll (Version 08.02.0000).
FMC34512I: The database name for alias 'FMCBTDB' is 'fmcbtdb'.
FMC34nnnI: Unified logon for 'wbiadmin' not possible
FMC34523I: DB2 Enterprise Server Edition 8.1.7 has fixpack WR21342 installed.
FMC34519I: QueryTimeoutInterval is '<null>' in the Common section of
c:\ibm\sqllib\db2cli.ini.
FMC34531I: DB2 authentication is through the server.

FMC34100I: Messages have been written to c:\documents and
settings\wbiadmin\fmczchk.log.
FMC34998I: See fmczchk.htm for more information on the messages.
FMC34999I: Configuration checker ended: 0 error(s), 2 warning(s), rc = 0.
```

For a more detailed discussion of the use of this tool for debugging runtime problems, refer to *WebSphere MQ Workflow Administration Guide*, SH12-6289, which can be obtained from the following Web site:

http://www-306.ibm.com/software/integration/wmqwf/library/manuals/wmqwf34.html

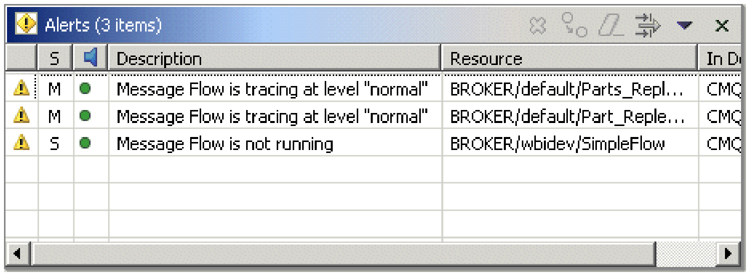
Note: The manuals for WebSphere MQ Workflow v3.4 are applicable to the WebSphere MQ Workflow v3.5 used in this book. At the time of printing this book, there were no manuals for the WebSphere MQ Workflow v3.5 release.

WebSphere Business Integration Message Broker

The message broker provides a user trace option that can be turned on dynamically. This facility can also be used by the message flow developers so that the message flow generates additional information about what is happening in the system. To use this facility as a developer, add a Trace node at crucial points in the message flow. Within the properties of the Trace node, you can provide free text or patterns that are known to the broker to produce additional information. A typical Trace pattern is:

```
${Root}
${LocalEnvironment}
${ExceptionList}
```

To turn on the user trace for one or more message flows, right-click the flow in the Domains view in the Broker Toolkit and select **User Trace** → **Normal**. The Alerts view lists the message flows for which user tracing is turned on.


Alerts (3 items)

	S		Description	Resource	In D
⚠	M	●	Message Flow is tracing at level "normal"	BROKER/default/Parts_Repl...	CMQ
⚠	M	●	Message Flow is tracing at level "normal"	BROKER/default/Part_Reple...	CMQ
⚠	S	●	Message Flow is not running	BROKER/wbidev/SimpleFlow	CMQ

Figure 10-19 User trace activated

The output of the user trace is in binary format. Reading the entries requires two steps:

1. Request the log entries from the broker with a command similar to this:

   ```
   mqsireadlog BROKER -e default -o messageflow.log -u
   ```

   This command requests the user trace entries for the default execution group of the broker BROKER.

2. The trace entries are stored in the messageflow.log file. This file is readable but it is still better to format the log entries in a more readable way by executing the command:

   ```
   mqsiformatlog -i messageflow.log -o messageflow.txt
   ```

   Figure 10-20 shows a sample log message that reports that an unexpected message has arrived at the AggregateReply node.

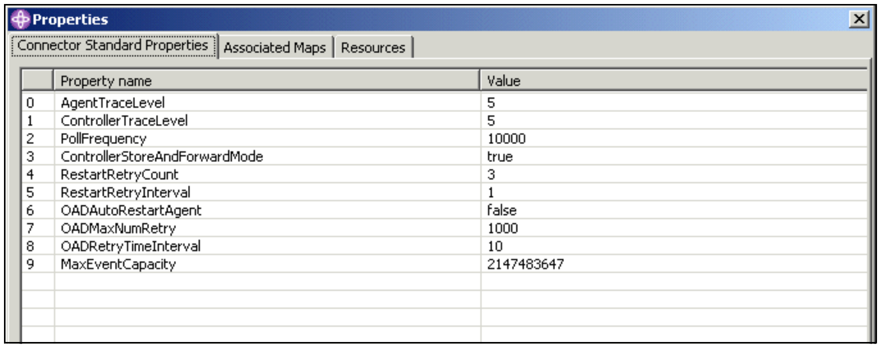
```
Timestamps are formatted in local time, 240 minutes before GMT.

BIP4412I: Corresponding request record not found for the reply message.
          An AggregateReply node has received a message at its 'in' terminal. No
corresponding record of a request message being sent could be found in the database. See
subsequent messages to determine how this situation has been handled.
          It is possible that extraneous messages are arriving at the AggregateReply node
'in' terminal. Check your flow to ensure that the only messages arriving here are replies to
request messages previously sent out and passed through an AggregateRequest node. It is
possible that this message is a valid reply but part of an aggregation which previously
timed out. It is possible that this is a reply to a message which has not yet been recorded
by an AggregateRequest node. This can happen if request messages are sent outside of
transactional control. Adjust your transaction settings to ensure that messages are sent
under transactional control.

Threads encountered in this trace:
  1928  2024
```

Figure 10-20 Sample user trace entry

WebSphere InterChange Server

The level of tracing can be increased dynamically for each component that provides trace functionality. Within the System Manager, select the Component Management view. Select, for example, the properties of a connector, as shown in Figure 10-21. You can increase the level of tracing for either the agent or the controller. Similar options exist for maps and collaborations.

The trace level of a collaboration can also be exploited by the developer of that collaboration. As such, the developer can make sure that additional information is made available when it is needed.



	Property name	Value
0	AgentTraceLevel	5
1	ControllerTraceLevel	5
2	PollFrequency	10000
3	ControllerStoreAndForwardMode	true
4	RestartRetryCount	3
5	RestartRetryInterval	1
6	OADAutoRestartAgent	false
7	OADMaxNumRetry	1000
8	OADRetryTimeInterval	10
9	MaxEventCapacity	2147483647

Figure 10-21 Updating the trace options for a connector

To increase the system trace level, right-click the ICS instance and select **Edit configuration**. The system editor opens. Select the **Trace/Log Files** tab to manage tracing and logging parameters.

WebSphere MQ trace

If tracing and logging information in the other runtime servers cannot help you to locate the source of the problem, it might be helpful to run a WebSphere MQ trace. A useful level of tracing is the API trace, which writes to a file with the invocation and return of every API call from a program that access the WebSphere MQ API. The API that is being traced is the C API, which sits underneath the Java and JMS APIs that are used by the WebSphere InterChange Server, for example. To read the output, you must be familiar with this C API and with the structures that are passed along API calls such as MQGET or MQOPEN. To start this trace, you can use the following command:

```
strmqtrc -t api -t detail
```

11

Tuning a WebSphere BI Server infrastructure

This chapter presents a lessons-learned discussion about performance-related settings for the major software components used in the WebSphere InterChange Server environment. Available tuning parameters for these deployment components, which through experience were observed to have most often affected performance, are identified and discussed.

11.1 Introduction

As with any business process integration project, several major software components work together to comprise the complete business integration processing environment. Clearly, a performance bottleneck in one component is likely to affect system throughput as a whole. Although overall system throughput and response time remain the primary measurements of interest, site administrators should possess some level of confidence that all of the major component parts are functioning at acceptable levels of performance.

In a situation in which end-user performance does not meet customer objectives, it is usually necessary to perform system-level monitoring in order to isolate the cause of the performance issue. The tools and techniques that are used for system-level monitoring differ from platform to platform. To find additional information about system-level monitoring, consult the system documentation and the manufacturer's Web site.

As an example, the WebSphere InterChange Server supports three different database managers. The documentation for each of these database products contains a wealth of information regarding performance, capacity planning, and configuration. This documentation offers the best guidance for performance considerations in a variety of operating environments, assuming that all of these issues have been addressed from the product's perspective. Additional levels of performance implications are introduced at the interface between these products and the WebSphere InterChange Server.

Several configuration parameters are available to the WebSphere InterChange Server administrator. This chapter identifies specific parameters that have been observed to affect performance, but not *all* available configuration parameters of the WebSphere InterChange Server. For a complete list of configuration parameters and possible settings, see Appendix A of *WebSphere InterChange Server V4.3.0 System Installation Guide*, which can be downloaded from:

http://www.ibm.com/software/integration/wbiserver/ics/library/infocenter/

11.2 General performance checklist

This section begins with a tuning checklist that enumerates the major components and their associated tuning concepts. The subsections that follow address each in more detail, describing performance-related design concepts and discussing the tuning parameters and their suggested settings (where appropriate), as well as ways to determine potential settings for a particular configuration. Although there is no guarantee that stepping through this checklist provides immediate, acceptable performance, it is likely that degraded performance can be expected if any of these parameters is set incorrectly.

The checklist includes:

- WebSphere InterChange Server
  - Configure threads in collaborations and in adapter controllers
  - Use caches for maps and collaborations (instance reuse)
  - Configure threading for CORBA / IIOP
  - Configure database connection pools
  - Setting flow control queue sizes
  - Turn off component tracing
  - Turn off event sequencing (if appropriate)
- WebSphere Business Integration Adapters
  - Configure polling frequency and quantity
- Database (general)
  - Place database tablespaces on fast disk subsystem
  - Size database cross-referencing tables correctly
  - Place logs on separate device from tablespaces
- Database (DB2-specific)
  - Maintain current indexes on tables
  - Update catalog statistics
  - Set buffer pool size correctly
- Database (Oracle-specific)
  - Set buffer, block, and shared pool area sizes correctly
  - Set processes, Open_Cursors, and IO_Slaves
  - Use dedicated connection if possible
  - Query optimization
- WebSphere MQ
  - Configure MQ log files, buffer pages
  - Monitor message queue depth
  - Place MQ logs on fast disk subsystem
- Java Virtual Machine
  - Set the heap and nursery size to handle garbage collection efficiently
  - Set AIX® threading parameters
  - Use HotSpot Server instead of client
  - Set thread stack size if using many threads
  - Reduce or increase heap size if java.lang.OutofMemory occurs
  - Set minimum and maximum heap size appropriately for peak loads for both server and adapter JVMs

- Large objects
  - Configure Java heap to appropriate size
  - Reduce concurrent activity when processing large objects
  - Process large objects as a set of smaller objects when possible

> **Note:** WebSphere InterChange Server V4.3.0 has new support for larger object sizes. This does not reduce the need for appropriate architecting of solutions that contain large objects.

11.3 WebSphere InterChange Server

This section discusses areas where you can impact performance for WebSphere Interchange Server.

11.3.1 Configure threads in collaborations and adapter controllers

With the WebSphere InterChange Server, you can control the maximum number of adapter controller and collaboration threads that are created. It also gives you limited control over when these threads are created and destroyed. Setting these parameters correctly can have an impact on performance. A basic knowledge of the InterChange Server threading model is required to best understand what the appropriate settings should be. The following description uses IBM WebSphere MQ as the source application delivery transport mechanism and describes the threading model for asynchronous access (not Server Access Mode). Server Access (synchronous) mode and the associated thread settings available are discussed in 11.3.3, “Configure threading for CORBA / IIOP” on page 469.

Within the WebSphere InterChange Server, each adapter controller and running collaboration is implemented with an associated thread pool. The thread pool is simply a set of worker threads that wait on an in-memory queue (mailbox) for work to perform.

After a source application adapter agent places an application-specific business object (ASBO) on a WebSphere MQ message queue, an MQ listener thread gets the source application generated objects (messages) off the message queue and puts them in the adapter controller mailbox. Threads in the adapter controller thread pool remove the messages from the mailbox and perform the mapping operation that translates an ASBO to generic business object (GBO), and publishes the GBO to the correct collaboration. Multiple adapter controller threads can be active at a time. Note that the ability to configure multiple MQ listener threads for adapters using MQ or JMS as the transport mechanism was added recently and can supply a significant boost in performance. This feature is explored in more detail in 11.4.2, “Multiple WebSphere MQ and JMS listener threads” on page 477.

A thread running in the collaboration pool dequeues the published GBO and executes the context of the collaboration. As is the case with the adapter controller threads, multiple collaboration threads can process business objects concurrently. When business object processing is complete, the destination adapter is invoked on this thread (with service call), which performs the outbound GBO to ASBO mapping operation. Note that the same collaboration thread is used through the mapping operation and any other processing that occurs within the WebSphere InterChange Server portion of the destination adapter. This thread then waits on an IIOP call as the service call completes. (Note that the execution flow is different for Long Lived Business Processing.) It is returned to the collaboration thread pool upon completion of any destination processing and subsequent post processing by the collaboration.

As stated above, the WebSphere InterChange Server enables the user to configure the limit on the total number of threads in any given active connector controller or collaboration thread pool. Additionally, the value for this parameter can be set differently for each object instance.

Ideally, the number of threads that are created would be sufficient to do the work required at peak load, but no more. Obviously, configuring the exact number of required threads is not feasible, but you can follow certain guidelines. For instance, if an adapter controller is only used for consume operations (service call requests), the adapter thread pool will not be used and the number of threads should be set to 1. For adapters that are used for input, the number of threads should be configured to avoid a bottleneck to the collaboration. Typical values are from 10 to 25 threads.

Setting the maximum number of threads in the collaboration depends on the length of service calls. The threads perform the service calls synchronously, so they have to wait for a return from the adapter. If the service call is very long, more threads might be needed. However, having too many threads wastes resources and has also been observed to increase variable behavior. As always, monitoring the running workload is the best way to set this value. Typical values for this thread pool range from 25 to 75. Active collaboration threads can be viewed in either the System Monitor or the System Manager. See the *Administrators Guide* for information regarding the use of these tools.

If the WebSphere InterChange Server is running multiple adapters and collaborations, then the total number of active threads might become an issue. There are costs associated with maintaining available threads. For example, thread limits can be enforced by the operating system on a per-process basis. Additionally, threads also consume space in the Java heap and in process virtual memory. (See 11.9.4, “Setting thread stack size if using many threads” on page 492.) To deal with this issue, an InterChange Server thread pool is made up of two types of threads: permanent and transient. *Permanent* threads are created when a thread pool is initialized. They remain active for the entire life of the thread pool. *Transient* threads are created and destroyed depending on the availability of work.

The existence of transient threads keeps the total number of active threads lower. By default, a transient thread exits if there is no work in the mailbox. On some workloads, it makes more sense for a transient thread to wait a specified amount of time for work to appear before exiting. This improves performance by not incurring the overhead of thread creation and deletion operations. The time to wait for work to be added to the mailbox before exiting can be managed by setting the global parameter `ThreadPoolLingeringTime` given in milliseconds in the InterchangeSystem.cfg file, as shown in Example 11-1. The heading Performance in this example is case-sensitive and is not an available option in the Configuration View of the System Manager. The full XML stanza is shown here and it must be added with a text editor.

Note: Changes to the InterchangeSystem.cfg file will not take affect until the WebSphere InterChange Server is restarted.

Example 11-1 Details of the configuration option for performance

```
<tns:property>
  <tns:name>Performance</tns:name>
  <tns:isEncrypted>false</tns:isEncrypted>
  <tns:updateMethod>system restart</tns:updateMethod>
  <tns:location>
    <tns:reposController>false</tns:reposController>
    <tns:reposAgent>false</tns:reposAgent>
    <tns:localConfig>true</tns:localConfig>
  </tns:location>
  <tns:property>
    <tns:name>ThreadPoolLingeringTime</tns:name>
    <tns:value xml:space="preserve">60000</tns:value>
    <tns:isEncrypted>false</tns:isEncrypted>
    <tns:updateMethod>system restart</tns:updateMethod>
    <tns:location>
      <tns:reposController>false</tns:reposController>
      <tns:reposAgent>false</tns:reposAgent>
      <tns:localConfig>true</tns:localConfig>
    </tns:location>
  </tns:property>
</tns:property>
```

Thread configuration example

A methodology describing how a system can be optimized for end-to-end throughput is beyond the scope of this book. System administration skills are required for performance monitoring and analysis of a wide range of software components, and sufficient performance analysis skills are needed to understand the profiling and trace data that is collected for this purpose. However, the topic is important enough to warrant some attention. In this section, we present a small thread-tuning scenario using the background information that we previously presented. Our scenario touches on possible causes of performance bottlenecks in the WebSphere InterChange Server system and how such bottlenecks can be detected and addressed by introducing some general performance-related concepts and methods.

Before beginning such an exercise, it would be useful to have at least an idea of what the expected system throughput should be. As an example, suppose a collaboration thread pool is configured with 15 threads and it calls a destination adapter that takes 200 milliseconds for service. Each thread in the collaboration thread pool can execute a maximum of five calls per second (1 sec / 200 ms = 5). With 15 threads, the maximum throughput of the collaboration cannot exceed 75 calls per second. Obviously, the actual throughput will be less than this theoretical maximum, but this should provide a good starting estimate. You can work out similar arithmetic for the adapter threads, if you know how long it takes to convert an ASBO to a GBO.

Several things could limit the maximum throughput in a WebSphere InterChange Server system deployment. For example, if the CPU utilization of the server is close to 100 percent, the throughput could possibly be increased by optimizing the maps and the collaborations in the system, thus shortening the code path. This is more of a development issue, which would have to be addressed by the team who are responsible for the development and integration of the code modules that support the particular deployment.

A second scenario might show that the CPU utilization and throughput remain low in spite of what is known to be a sufficient amount of work being injected into the system. Such a scenario hints at a performance-related issue, indicating that throughput is limited by some bottleneck in the system. Assuming that surrounding software components are believed to be performing acceptably, we can approach the problem through the perspective of the InterChange Server.

Within the InterChange Server, bottlenecks can be identified by monitoring the adapter and collaboration queues in the system when it is under *stress*, when more work is being injected than can be processed. If it is found that an adapter queue is continually growing, it implies that processing the maps is the bottleneck. You should then make sure that there are enough threads in the thread pool associated with the adapter queue. The

ConcurrentEventTriggeredFlows property on the Standard Properties tab of the Connector Configurator tool specifies the number of threads in the pool.

If the collaboration queue is growing continually, then collaboration processing or *service call time,* application latency, is a likely candidate. To mitigate the problem, increase the size of the thread pool is associated with the collaboration pool by modifying Maximum Number of Concurrent Events in the Collaboration general properties. Right-click the collaboration object in the System Manager and select **Properties**. As noted, a collaboration queue that grows continually can also be indicative of long service times or application latencies. If this is the case, then the destination application should be examined.

The AgentConnections property in the destination adapter is a parameter that has been known to cause significant confusion due to the property name chosen. This parameter should not be changed.

If the adapter and collaboration queues are not growing in the stressed system, then consider the adapter itself. In this example, by monitoring the MQ message queue for the source adapter and seeing that it is continually growing, it might indicate that more listener threads are necessary. Increase the ListenerConcurrency property on the Standard Properties tab of the Connector Configurator tool. MQ and DB2 accesses are made from these listeners, so it is imperative to optimize the WebSphere MQ and database accesses. See the relevant MQ and database sections for more details.

11.3.2 Use caches for maps and collaborations (instance reuse)

The WebSphere InterChange Server provides the user with the ability to cache collaboration and map objects. This instance reuse can reduce the number of objects that are created and initialized and, thus, improve performance.

Instance reuse for maps can be enabled by editing the map properties for a given map and checking the Instance Reuse box. You can also set the map cache size here. If the map is used for subscription delivery, it will be executed by a thread in the adapter controller pool and the map cache should be set to the same size as the number of threads in the adapter controller (see previous section). If the map is for consuming or being executed after the return from the service call then the size of the cache should be set to the number of threads in the collaboration.

Enable Instance reuse for collaborations by defining EnableInstanceReuse as a property in the collaboration template and setting the value to `true`.

11.3.3 Configure threading for CORBA / IIOP

The VisiBroker Object Request Broker (ORB) that shipped with all the previous versions of WebSphere InterChange Server is replaced in WebSphere InterChange Server V4.2.2 with the IBM ORB on all platforms, regardless of the underlying JVM™. Though it is functionally equivalent, some changes have been made in the way that ORB services and classes are designated at server startup, adapter startup, or both. Most of these changes are transparent to the user, because they are primarily related to system configuration and administration.

The most visible difference is the inclusion of the *IBM transient name server*, the CORBA naming registry service that replaces the function previously provided by the VisiBroker osagent. While the previous naming service was dynamic in nature, the transient name server is static, meaning that each client machine must be directed to the naming server to obtain required object references.

Another service that is provided by the VisiBroker ORB is the Object Activation Daemon, which enables you to restart remote agents from the System Manager. An equivalent function is now provided through WebSphere MQ triggering. See the product documentation for a description and how to configure it.

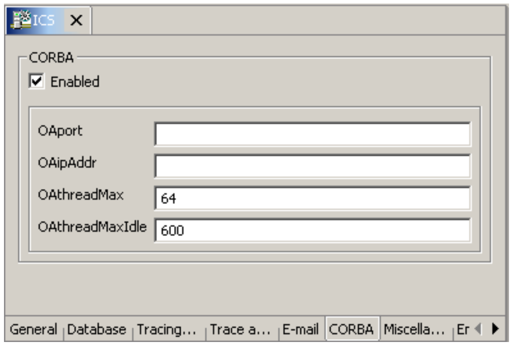
Previous versions of this information described the importance of managing the ORB thread pool, particularly when using the Server Access Interface. The same concepts still hold true, but there are differences in both the syntax and initialization of parameters that control the ORB worker threads starting in WebSphere InterChange Server V4.2.2 and still present in V4.3.0. Like the previous ORB, the IBM ORB also maintains a pool of threads used to service synchronous access client requests through the entire flow, including service calls. Limiting the number of threads created as well as allowing these threads to remain active for a designated period after use can produce a noticeable performance improvement. The idea is to tune the thread pool such that a sufficient number of worker threads exist and delay exiting long enough to be reused, thereby decreasing the overhead associated with thread creation and termination.

Table 11-1 identifies previous and current properties that are associated with the ORB thread pool.

Table 11-1 Properties used in previous and current versions for the ORB

Previous property (in InterChangeSystem.cfg)	**Setting property for WebSphere InterChange Server V4.3.0 and Corresponding adapters (passed as -Dargument to JVM)**	**Typical value**
OAthreadMax	com.ibm.CORBA.ThreadPool.MaximumSize	64
OAthreadMaxIdle	com.ibm.CORBA.ThreadPool.InactivityTimeout	600

The System Manager configuration view was updated in the WebSphere InterChange Server V4.3 to allow for modification of these CORBA settings, as shown in Figure 11-1. An alternative is to include all desired ORB properties in the ORB_PROPERTY variable defined in these system-specific environment variable files.



CORBA
Enabled [✓]

OAport	
OAipAddr	
OAthreadMax	64
OAthreadMaxIdle	600

Figure 11-1 CORBA configuration via System Manager

See “Administering the Object Request Broker” in the Administering problem scenarios section of the *System Administration Guide* for a complete description of all available parameters and how to configure them. This guide is available at:

http://publib.boulder.ibm.com/infocenter/wbihelp/index.jsp

11.3.4 Configure database connection pools

During the course of flow execution, multiple accesses are made to the WebSphere InterChange Server databases. These accesses are used to implement recovery, event sequencing, dynamic relationships, flow monitoring, and transactional control. The number of database connections that the WebSphere InterChange Server uses varies greatly, based on the configured services. In an active runtime environment, the event management service is busy storing state information about events that arrive at the WebSphere InterChange Server. Collaborations might also add to the traffic by reading tables in the repository to make application-routing decisions, or by saving and retrieving transaction state information.

Establishing a new connection to a database is a time-consuming operation. In order to minimize the time taken to effect a database transaction and improve performance, the WebSphere InterChange Server maintains dynamic pools of database connections for each of the following:

- Event Management
- Transactions

- Repository
- Flow Monitoring
- Relationships

These pools maintain active database connections that may be used by collaborations and mapping processes within the WebSphere InterChange Server.

The maximum number of database connections allowed can be set individually for each pool or globally for all pools. If all of the listed services are to a single data source, one database for the WebSphere InterChange Server, then MAX_CONNECTIONS should be set at the global level. To set this level, it is best to accept the default at installation time, then monitor connection usage and adjust accordingly, erring on the side of too many, rather than not enough, connections.

However, note that a connection limit can exist in the DBMS server configuration. If the number of connections requested by the WebSphere InterChange Server is larger than that for which the database is configured, then errors will occur. Consversly, when the WebSphere InterChange Server cannot meet a connection request, the server's action varies according to why it needed the connection. In some cases, the server might simply log an error message; in others, it might stop completely. For this reason, it is important to avoid restricting the number of connections so much that the WebSphere InterChange Server cannot meet the workload. For information about how to check the log for connection failures, see the *System Administration Guide*.

To summarize, for best stability and performance, keep enough connections active in the pool to handle the largest possible load and to make sure that the database is configured to provide this number of connections.

Note that the maximum number of connection pools that will be created (MAX_CONNECTION_POOLS) may also be set. By default, the WebSphere InterChange Server requires a minimum of three connection pools, one each for event management, transactions, and the repository. The default value of 10 for this parameter should be sufficient in most cases.

11.3.5 Setting flow control queue sizes

The primary objective of the flow control feature is to prevent the WebSphere InterChange Server from becoming overwhelmed and potentially halting due to events backing up in the system and exhausting the Java heap. This situation could arise, for example, if a destination application is offline but a source application continues to inject new business objects for processing. In this simple case, the queues associated with the collaborations would grow unchecked, and the WebSphere InterChange Server would eventually run out of heap memory, leading to a system halt.

Beginning with WebSphere InterChange Server Version 4.2.0, the user can associate a maximum event queue size with each collaboration and connector queue. If a queue grows to its maximum size, no more events are accepted (a *blocking* queue), or the new events are persisted to the database (a *nonblocking* queue). Events are requeued only after consumption has resumed and the size of the queue shrinks below a certain threshold. For the case where a destination application is offline, a collaboration queue configured for flow control would reach its maximum capacity and stop accepting objects. The input connector would not be able to queue any events for processing until the destination application was back online.

In this section, we present guidelines that can be used to set the primary flow control-related variables: queue depth and threshold.

Recommended values for flow control queue size, threshold

The recommended setting methodology is straightforward. Each queue feeds a collaboration or a connector. These components are implemented as thread pools. It is recommended that the queue depths be set to three times (3x) the number of threads in the corresponding connector. For example, if a collaboration has 30 threads, the queue depths should be set to 90.

The logic behind this is also straightforward. The queues are set to small values so that there is always work for each of the threads in a connector or collaboration to perform. Performance experiments indicate that a threshold of 0.67 works well with the 3x value:

- Queue depth: three times the number of threads in a collaboration or connector pool
- Threshold: 0.67

Current flow control queue depths for the connectors and collaborations in the WebSphere InterChange Server can be monitored using either the System Monitor or System Manager. To change the flow control parameters, consult the product documentation.

11.3.6 Turn off component tracing

The ability to configure event tracing at different levels for a variety of system components has proven be extremely valuable during periods of system analysis or debugging. However, use of this feature can affect, in some cases significantly, overall system performance and throughput. It is recommended that all unnecessary tracing be turned off to ensure optimal performance.

11.3.7 Turn off event sequencing where applicable

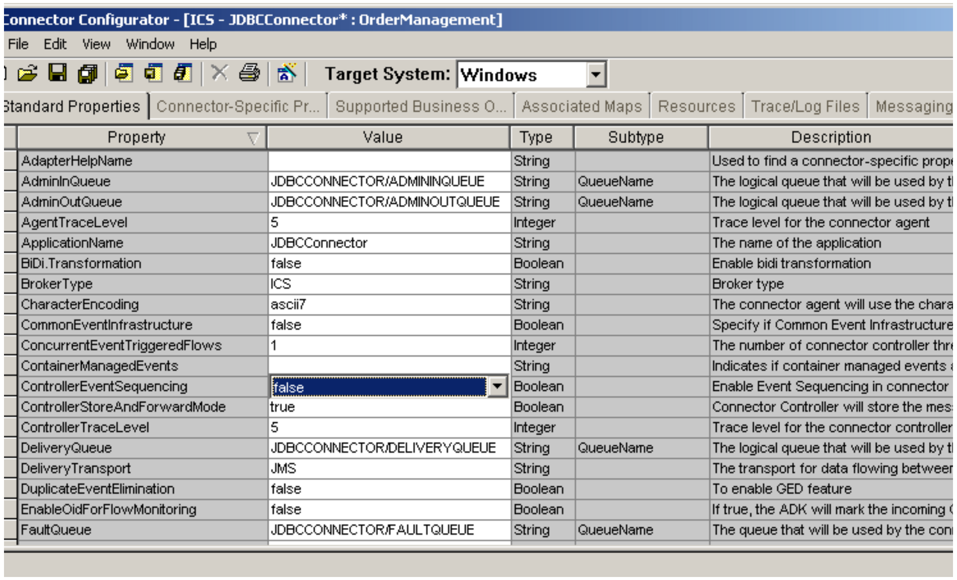
The WebSphere InterChange Server is able to process new events in both an asynchronous and synchronous nature. *Asynchronous* processing occurs when receiving new events from adapters with the asynchronous delivery mechanism of the adapter framework. *Synchronous* event processing occurs when receiving events either through the WebSphere InterChange Server Access (previously the Server Access Interface) or through the synchronous delivery mechanism of the adapter framework.

Because it is recommended to configure the server to process new events concurrently, it is imperative to have event sequencing and event isolation. Where event sequencing ensures that two threads of the *same collaboration* do not work on the same data concurrently and event isolation ensures that *two or more collaborations* do not work on the same data concurrently. What is not documented in the product manuals is the fact that event sequencing also exists within adapter controllers, which prevents maps from working on the same data at the same time (this happens when Current Event Triggered Flows is set to a value greater than one). By default, the WebSphere InterChange Server has both event sequencing and event isolation enabled for all incoming events for collaborations and event sequencing for adapter controllers.

In the past, it was thought that the majority of integrations using asynchronous events required event sequencing, while the majority of integrations using synchronous events did not require event sequencing. It has been found that these past assumptions are not practical, because many customer implementations use both asynchronous and synchronous events with different requirements. Thus, new in WebSphere InterChange Server V4.3.0, is the ability for implementors to disable event sequencing and event isolation. The purpose of exposing this feature is that if the integration requirements do not require event sequencing and event isolation, performance gains can be achieved. This feature is implemented by the use of two boolean attributes and can be accessed at a very granular level, namely with individual adapter controllers and collaborations.

- *ControllerEventSequencing* (Figure 11-2 on page 474) as an adapter controller attribute
- *Event Isolation* (Figure 11-3 on page 474) as a collaboration object property

Note: The Event Isolation attribute in the collaboration object properties disables both event sequencing and event isolation for that collaboration.


Connector Configurator - [ICS - JDBCConnector* : OrderManagement]

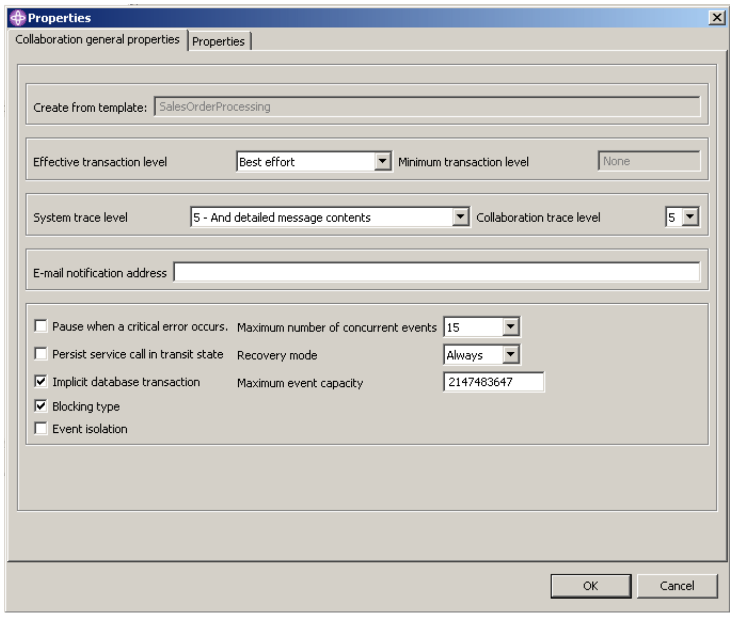
File Edit View Window Help

Target System: Windows

Standard Properties | Connector-Specific Pr... | Supported Business O... | Associated Maps | Resources | Trace/Log Files | Messaging

Property	Value	Type	Subtype	Description
AdapterHelpName		String		Used to find a connector-specific prope
AdminInQueue	JDBCCONNECTOR/ADMININQUEUE	String	QueueName	The logical queue that will be used by t
AdminOutQueue	JDBCCONNECTOR/ADMINOUTQUEUE	String	QueueName	The logical queue that will be used by t
AgentTraceLevel	5	Integer		Trace level for the connector agent
ApplicationName	JDBCConnector	String		The name of the application
BiDi.Transformation	false	Boolean		Enable bidi transformation
BrokerType	ICS	String		Broker type
CharacterEncoding	ascii7	String		The connector agent will use the chara
CommonEventInfrastructure	false	Boolean		Specify if Common Event Infrastructure
ConcurrentEventTriggeredFlows	1	Integer		The number of connector controller thr
ContainerManagedEvents		String		Indicates if container managed events
ControllerEventSequencing	false	Boolean		Enable Event Sequencing in connector
ControllerStoreAndForwardMode	true	Boolean		Connector Controller will store the mes
ControllerTraceLevel	5	Integer		Trace level for the connector controller
DeliveryQueue	JDBCCONNECTOR/DELIVERYQUEUE	String	QueueName	The logical queue that will be used by t
DeliveryTransport	JMS	String		The transport for data flowing betweer
DuplicateEventElimination	false	Boolean		To enable GED feature
EnableOidForFlowMonitoring	false	Boolean		If true, the ADK will mark the incoming
FaultQueue	JDBCCONNECTOR/FAULTQUEUE	String	QueueName	The queue that will be used by the con

Figure 11-2 Adapter ControllerEventSequencing attribute


Properties

Collaboration general properties | Properties

Create from template: SalesOrderProcessing

Effective transaction level: Best effort Minimum transaction level: None

System trace level: 5 - And detailed message contents Collaboration trace level: 5

E-mail notification address:

- [ ] Pause when a critical error occurs.
- [ ] Persist service call in transit state
- [x] Implicit database transaction
- [x] Blocking type
- [ ] Event isolation

Maximum number of concurrent events: 15

Recovery mode: Always

Maximum event capacity: 2147483647

OK Cancel

Figure 11-3 Collaboration object properties Event Isolation attribute

It is recommended to use this feature only when it is very obvious that event sequencing and event isolation is not required. These attributes can be configured dynamically but will only affect new events, not events currently in progress. By default, both of the mentioned attributes are set to true, to enable event sequencing and event isolation. In upgrading from a previous version, these two attributes will be added to your configuration and set to the default values.

Note: To correctly use this feature, you must ensure that all the event sequencing switches on the path of the event flow are set the same.

11.4 WebSphere Business Integration Adapters

This section will discuss areas where you can impact performance for WebSphere Business Integration Adapters.

11.4.1 Configure poll frequency and poll quantity

Two of the most important configuration parameters for the WebSphere BI Adapters are Poll Frequency and Poll Quantity:

- Poll Frequency specifies the amount of time in milliseconds between the end of one polling action, and the start of the next polling action.
- Poll Quantity specifies the maximum number of objects to process during a polling action.

These parameters control the rate and amount of work that an adapter processes, so the combination of Poll Frequency and Poll Quantity regulate the number of transactions that are processed first by the adapter, and then by the broker (for example, the WebSphere InterChange Server). As such, these parameters influence the performance of the entire solution, not just the adapter.

Nonoptimal values for Poll Frequency and Poll Quantity can result in either low system throughput (Poll Frequency is too long, Poll Quantity is too low, or both). Can cause excessive memory usage (and potentially OutOfMemory exceptions) if the parameters are configured to deliver events to the system at rates that are higher than the solution is implemented to handle (if Poll Frequency is too short, Poll Quantity is too high, or both). Both of these conditions dramatically affect overall system performance, so appropriate settings for Poll Frequency and Poll Quantity are critical and should be explicitly configured to support the level of throughput a solution is designed to handle.

In general, the recommendation is to configure Poll Frequency and Poll Quantity to enable events to be retrieved and processed at a level that matches the peak throughput of the WebSphere BI solution. This is discussed in more detail below.

The definition of Poll Frequency merits further clarification because it is sometimes misunderstood. As stated previously, this parameter defines the amount of time between the end of one polling action and the start of the next polling action. Note that this is not the interval between polling actions. Rather, the logic is:

- Poll to obtain (up to) Poll Quantity number of objects
- Process these objects (for some adapters, some of this work is done on separate threads, which execute asynchronously to the next polling action)
- Delay for the interval specified by Poll Frequency
- Repeat

Note that the default setting for Poll Frequency is 10 seconds (10,000 ms). For workloads with reasonably high throughput requirements, this default should be changed to enable more frequent polling.

On the other hand, Poll Quantity is an adapter-specific parameter and the default is different for different adapters. For example, the default for Poll Quantity for the JText adapter is 25 and the default for the WebSphere MQ adapter is 1. This parameter should be checked for each adapter and set accordingly.

As an example, if the peak throughput rate of a solution is 20 events per second, appropriate settings could be Poll Frequency = 1000 ms and Poll Quantity = 20. This supports the required peak throughput, while requiring a relatively small number of events to be concurrently held in the adapter process. Factors that may require an adjustment to these values include:

- The size of the object being processed

  For larger objects, a good general rule is to use a lower Poll Quantity and longer Poll Frequency. This does not generally apply for relatively small objects (100 KB or less). However, for larger objects, it is important to minimize the number of objects that are held concurrently in the adapter process, in order to avoid potential OutOfMemory exceptions. To extend the example, if the object size is 1 MB and the throughput rate of the solution is 10 events per second, appropriate settings could be Poll Frequency = 200 ms and Poll Quantity = 2.
- The Java heap size and physical memory available on the system

  In general, the larger the heap, the higher Poll Quantity can be set. However, several factors are involved in setting the heap size. One very important factor is to ensure that the heap is not set so large that paging results. Paging the Java heap dramatically reduces system performance. See the Java configuration information later in this section for a detailed discussion about setting the Java heap sizes appropriately.

- The uniformity of event arrival rates

  The previous examples assume that events arrive at a relatively constant rate. This might not be true for many solutions; event arrival is sometimes very uneven. In these cases, care must be taken to balance processing events in a timely manner to handle the occasional high arrival rates, while also not holding too many events in memory concurrently, and potentially encountering OutOfMemory exceptions.

11.4.2 Multiple WebSphere MQ and JMS listener threads

Prior versions of the WebSphere InterChange Server had known limitations to the rate at which work could be accepted from a source application when using WebSphere MQ or JMS as the adapter transport. Each transport contained a single listener thread that monitored the MQ and JMS mailbox respectively, in essence providing a single point through which all input objects had to pass.

In WebSphere InterChange Server V4.2.1, the ability to specify the number of MQ listener threads was introduced, producing significant effect on overall system throughput. Thus, the recommended adapter transport for events was WebSphere MQ, but new in WebSphere InterChange Server V4.3.0 is the ability to specify multiple JMS listener threads allowing the JMS transport to perform equal to benchmarks established by the same number MQ listener threads.

- When using MQ as the adapter transport, the attribute *ListenerConcurrency* can be configured through the Connector Configurator.
- When using JMS as the adapter transport, the attribute *jms.TransportOptimized* can be configured to `true`, followed by the attribute *jms.ListenerConcurrency* to the number of listeners desired.

If it is determined that the single listener thread is the bottleneck, the number of listener threads should be set to the number of processors in the system, but pay close attention to the CPU usage. If the CPU usage appears high, experiment by reducing the number listener threads.

11.5 General database performance

This section discusses general areas where you can improve database performance. The following sections discuss ways to improve the performance of specific databases.

11.5.1 Place database tablespaces on a fast disk subsystem

The database tablespaces should be placed on a fast disk subsystem with write back cache. This has a direct bearing on the database performance.

11.5.2 Size database cross-referencing tables correctly

Relationship cross-reference tables for identity relationships and other dynamic relationships grow continuously and can become quite large in a production environment. Care must be taken to size these tables accordingly. It is recommended that the first extent of the table contains as many rows as possible, even the entire table. This will avoid extent inter-leaving and enable faster database service times on relationship lookups.

Currently there is no mechanism to specify extent sizes and physical storage attributes for these tables through the System Manager. Therefore, if these tables are expected to be large, the recommended method to achieve this is to export the data in these tables, drop them, r-create them according to the desired storage parameters, and then reload the data.

Determining which tables in the WebSphere InterChange Server database are the cross-reference tables is straightforward. Each relationship participant has a cross-reference table dedicated to itself. For example, if there is a relationship definition for a dynamic relationship named Customer with participants SAP, Clarify, and PeopleSoft, there are three distinct tables in the database for each participant. The name of the table is included in the relationship definition.

Many of the other tables in the WebSphere InterChange Server database do not grow continuously. but are used for temporary storage. These database tables grow and shrink. They include: CxWIPObjects, CxWIPBlobs, CxFailedEventKeys, CxPBusObjState, and CxPBusObjMessage. Additionally, if transactional collaborations are used, the following tables also grow and shrink: CxAStateBusObjs, CxCStateBusObjs, CxTransBlobs, CxCompBusObjs, and CxReposAttributes.

11.5.3 Place logs on separate device from table spaces

A basic strategy for all database storage configurations is to place the database logs on separate devices from the tablespace containers. This prevents I/O to tablespace containers from contending with the I/O to the database logs.

11.6 Database: DB2-specific

Providing a comprehensive DB2 tuning guide is beyond the scope of this book. On the other hand, a few general rules can assist in improving the performance of DB2 environments. In the paragraphs that follow, we discuss these rules and provide pointers to more detailed information. A quick reference for DB2 performance tuning can be found at the following Web site:

http://www-106.ibm.com/developerworks/db2/library/techarticle/0205parlapalli/0205parlapalli.html

11.6.1 Maintain current indexes on tables

While the WebSphere InterChange Server creates a set of database indexes that are appropriate for many installations, additional indexes might be required in some circumstances. A database environment that requires additional indexes often exhibits performance degradation over time; in some cases the performance degradation can be profound. Environments that need additional indexes often exhibit heavy read I/O on devices holding the tablespace containers. To assist in determining which additional indexes could improve performance, DB2 provides the Design Advisor. The Design Advisor is available from the DB2 Control Center, or can be started from a command line processor. Design Advisor has the capability to help define and design indexes suitable for a particular workload.

11.6.2 Update catalog statistics

It is important to update the DB2 catalog statistics on a regular basis. These are statistics that are used by the DB2 query optimizer to determine the access plan for evaluating a query. Statistics are maintained on tables and indexes. Examples of statistics include the number of rows in a table, and the number of distinct values in a certain column of a table. These statistics are not maintained by DB2 in real time; statistics are updated by DB2 commands that are typically run by the DBA. If statistics are not updated on a regular basis, the DB2 query optimizer may create poor-performing access plans for evaluating queries. The following command can be used to update statistics on all tables in the database:

```
db2 -v reorgchk update statistics on table all
```

More information about maintaining catalog statistics can be found in the document “DB2 Administration Guide: Performance,” which is available here:

ftp://ftp.software.ibm.com/ps/products/db2/info/vr8/pdf/letter/db2d3e80.pdf

11.6.3 Set bufferpool size correctly

Although there are many DB2 configuration parameters, one parameter that is critical to performance and often requires modification in a new installation is the bufferpool size (BUFFPAGE parameter from the database configuration). In general, the default value of this parameter is too small to run efficiently; on the other hand, the parameter should be set small enough so that the database bufferpool can coexist along with other structures and applications, without exhausting the real memory on the system.

11.7 Database: Oracle-specific

Compiling a comprehensive Oracle tuning section is beyond the scope of this book. The guidelines provided here for tuning are generic and the objective is to highlight areas that require attention while monitoring and tuning an Oracle database. Database accesses can vary significantly by workload. Optimal configuration and parameter values depend on the specific WebSphere InterChange Server installation and can only be arrived at by monitoring the V$ dynamic performance views for status of files and memory.

11.7.1 Set buffer, block, and shared pool area sizes correctly

The following parameters are important to WebSphere InterChange Server performance with Oracle. They are set in the init<sid>.ora file.

DB_BLOCK_BUFFERS or DB_CACHE_SIZE

Increase the appropriate parameter to reduce the number of physical reads from disk. As a general rule, set to 25% of total memory allocated to Oracle processes. The DB_CACHE_SIZE (Oracle 9) parameter is the actual memory allocated. If DB_BLOCK_BUFFERS (Orracle 8) is specified, the total memory allocated is equal to DB_BLOCK_SIZE times DB_BLOCK_BUFFERS. The DB_CACHE_SIZE divided by the DB_BLOCK_SIZE equals the number of available database buffers.

DB_BLOCK_SIZE and DB_FILE_MULTIBLOCK_READ_COUNT

These values should be tuned in conjunction with database memory buffer allocation (DB_BLOCK_BUFFERS or DB_CACHE_SIZE). They control the amount of data fetched in one single read from the disk. It should be noted that changing DB_BLOCK_SIZE requires re-creation of the Oracle instance. So this value must be determined at installation time.

SHARED_POOL_SIZE

The shared pool area must be large enough to accommodate the most frequently used SQL statements and Data Dictionary information. Use V$ views to monitor shared pool area hit ratio.

LOG_BUFFER

Increase the LOG_BUFFER value to reduce physical reads when processing redo entries in the redo log file.

LOG_CHECKPOINT_INTERVAL

This is the interval, equal to the number of redo blocks written, before the next checkpoint. It should be set to `0` to reduce disk writes.

11.7.2 Set processes, Open_Cursors, and IO_Slaves

Consider the values of the following settings:

- PROCESSES specifies the number of operating system user processes that can connect to an Oracle database server. It should be set to `1000` for WebSphere InterChange Server usage.
- OPEN_CURSORS should be set to a value of `1200` or more to ensure enough available cursors for various connections.
- DBWR_IO_SLAVES configures the number of slaves for disk writes. As a general rule, one slave per disk is recommended.

11.7.3 Use a dedicated connection

With respect to Oracle, if the database happens to be dedicated to the WebSphere InterChange Server and fewer than 200 connections are active, it is recommended that a dedicated connection mechanism be used.

11.7.4 Query optimization

OPTIMIZER_MODE specifies the mode that is used to determine how to access data to fulfill an SQL query. The default setting is `choose`, which uses a rule-based optimization if no table statistics are available, or a cost-based optimization if statistics are available. A setting of `rule` forces rule-based optimization. The performance impact of the setting varies with the type of workload. A production environment, with complex queries, in which up-to-date table statistics are available, might benefit from a cost-based approach.

11.8 WebSphere MQ

Because WebSphere MQ is one of the primary transports for WebSphere Business Integration Adapters, it is important to configure it correctly. There are many documents available about this subject and some are referenced in the links below. The reader is strongly encouraged to pursue these links for more detailed information. The purpose of this section is to document those tuning options that we found particularly useful in maximizing the throughput.

11.8.1 Place MQ logs on fast disk subsystem

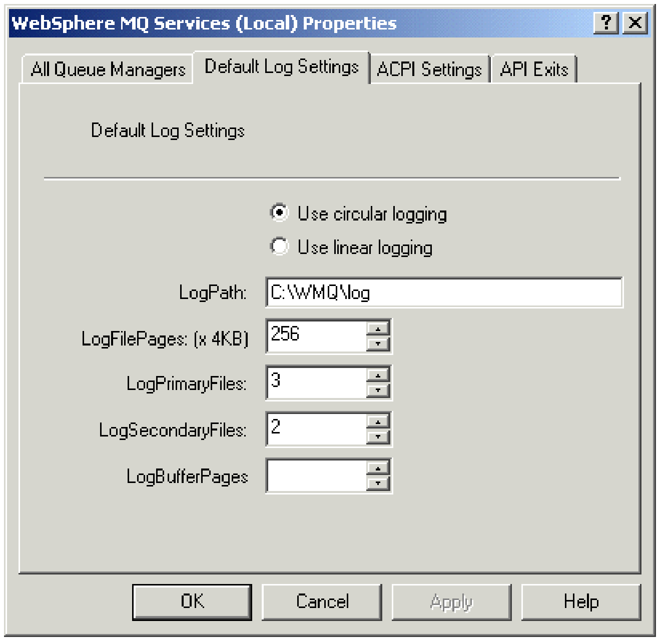
WebSphere Business Integration Adapters use persistent messages, which require significant disk I/O. This disk I/O comes from two sources: the queue manager's log and queue data itself. If care is not taken, these I/Os can become a severe system bottleneck. One way to mitigate this issue is to use disk drives and adapters with fast write-back caches. This cache could be on the drives themselves (for example, certain SSA disks) or on the disk adapter (for example, one particular model of the IBM ServeRaid adapter used had 32 MB of onboard NVRAM).

The steps that ensure that MQ-related I/O is directed to a fast storage device are:

- Install fast-write disks or adapters on the MQ system. If using a RAID adapter, configure a Raid-0 array of one or more physical drives. Note that the MQ log, being sequential, does not get any benefit from disk striping, so multiple-drive arrays are not of much benefit here. The primary use of the RAID adapter is for its fast-write caching capability.
- Create a filesystem (AIX, Solaris™) or partition (Windows) on the fast disk or array, and a directory on it to store the MQ logs and queues.
- Configure MQ to use this directory by default for logs and queue data (Figure 11-4 on page 483). On Windows, this location can be specified at installation. After installation, this default can be changed using the WebSphere MQ Services application.

  Right-click the WebSphere MQ Services (Local) folder and select **Properties**. Change the values on the All Queue Managers tab to change the default queue folder. You can also change the default log folder.

  These changes do not affect existing queue managers. It is possible to edit registry entries and move directory trees to the desired locations. However, it is probably safer to recreate the queue manager after changes have been made to these properties.



Default Log Settings

(●) Use circular logging
( ) Use linear logging

LogPath:	C:\WMQ\log
LogFilePages: (x 4KB)	256
LogPrimaryFiles:	3
LogSecondaryFiles:	2
LogBufferPages	

Figure 11-4 Changing the properties for all queue managers

On AIX, the LogDefaultPath and DefaultPrefix values can be modified in the /var/mqm/mqs.ini file.

Additionally, the default log path can be overridden when creating a queue manager using the **`-ld`** option of the **`crtmqm`** command.

If you must recreate a queue manager, it can be helpful to use SupportPac™ MS03, which enables you to extract object definitions from a queue manager so that it can be used in another queue manager or when the queue manager is re-created. SupportPac MS03 can be downloaded from:

`http://www-306.ibm.com/software/integration/support/supportpacs/`

11.8.2 Monitor message queue depth

Each message queue has an associated maximum depth, which is the maximum number of elements that can be in the queue at any one time. This value can be modified in the MQ Explorer by modifying the Maximum Queue Depth property of the queue of interest.

Care should be taken to monitor the current depth of the queues of interest during system operation. This can be done either from the MQ Explorer Queues view, or the System Manager Server Statistics view. If the queue depths grow over time, it is likely the case that the server is not processing events fast enough for that adapter. A particularly insidious effect here is that increasing queue depth tends to make MQGET operations slower, so as a server gets further and further behind, the service times to read from the input queue get longer and longer.

11.8.3 Configure WebSphere MQ log files and buffer pages

The following WebSphere MQ log file and buffer page settings should be examined for their applicability to any configuration from a performance standpoint. The settings and a brief description are given below:

LogFilePages

WebSphere MQ log data is held in a series of files called *log* files. The log file size is specified in units of 4 KB pages. A larger value enables bigger log files. This is not really a performance knob, but it does aid in recoverability. Its value is set during the creation of the queue manager and cannot be altered.

- Recommended value: `2048`
- Default on Windows: `256`
- Default on UNIX: `1024`

LogPrimaryFiles

Primary log files are the log files that are allocated during queue manager creation. Its value can be changed for an existing queue manager.

- Maximum value: 62
- Default value: 3

LogSecondaryFiles

Secondary log files are the log files that are allocated when the primary files are exhausted. This value can be changed for an existing queue manager.

- Maximum: `61`
- Default: **`2`**

LogBufferPages

This is the amount of memory that is allocated to buffer records for writing. The size of the buffers is specified in units of 4 KB pages. Larger values lead to higher throughput, especially for larger messages, because this reduces the number of I/Os but increases the size of each I/O.

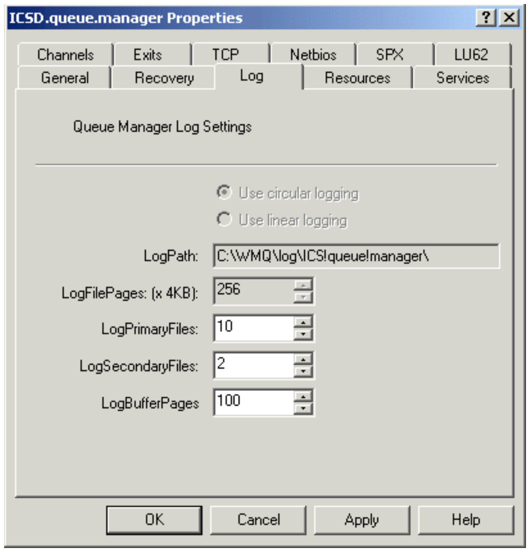
- Maximum value: `512`
- Default value: `18`

LogWriteIntegrity

This parameter controls the reliability with which log files are written. There is a performance trade-off, with the default providing the highest level of reliability, and a value of `SingleWrite` providing the best performance.

- Default: `TripleWrite`

For attributes that can be changed, right-click the queue manager in the WebSphere MQ Services application and select **Properties**. Select the **Log** tab (Figure 11-5). For UNIX platforms, edit the `qm.ini` file for that queue manager.



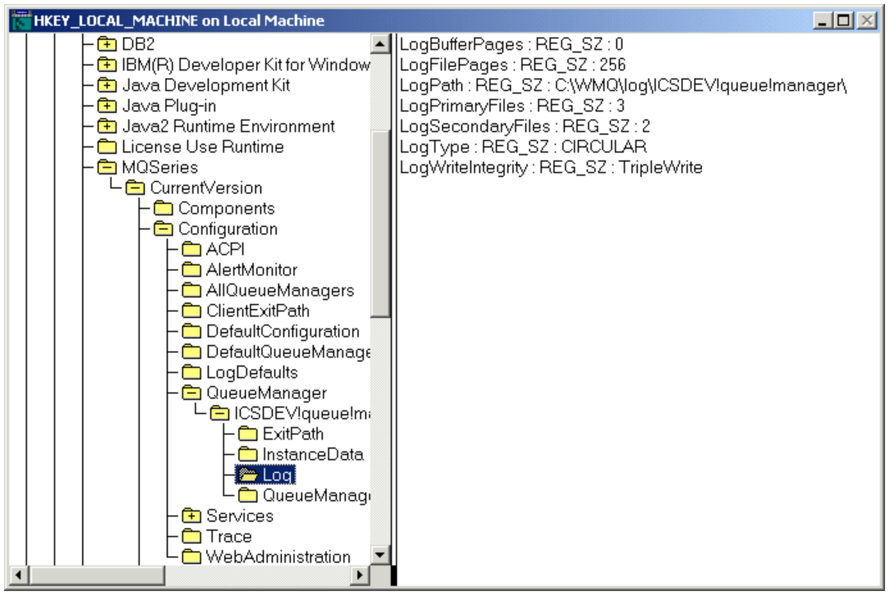
Queue Manager Log Settings

(●) Use circular logging
( ) Use linear logging

LogPath: C:\WMQ\log\ICS!queue!manager\
LogFilePages: (x 4KB): 256
LogPrimaryFiles: 10
LogSecondaryFiles: 2
LogBufferPages 100

Figure 11-5 Updating the logging parameters for a queue manager

The parameter LogWriteIntegrity (Figure 11-6 on page 486) is not exposed in the WebSphere MQ Services application on Windows. You can change this parameter using the registry editor.



```
LogBufferPages : REG_SZ : 0
LogFilePages : REG_SZ : 256
LogPath : REG_SZ : C:\WMQ\log\ICSDEV!queue!manager\
LogPrimaryFiles : REG_SZ : 3
LogSecondaryFiles : REG_SZ : 2
LogType : REG_SZ : CIRCULAR
LogWriteIntegrity : REG_SZ : TripleWrite
```

Figure 11-6 Reviewing the parameter LogWriteIntegrity

Consult the IBM WebSphere MQ documentation for a description of all WebSphere MQ log file and buffer pages tuning parameters. The WebSphere MQ documentation is available at:

`http://www-306.ibm.com/software/integration/mqfamily/library/manualsa/`

11.9 Java

Because the WebSphere InterChange Server and its components, such as maps and collaborations, are written in Java, the performance of the Java Virtual Machine (JVM) has a significant impact on the performance that is delivered by a WebSphere InterChange Server application. JVMs externalize multiple tuning parameters that can be used to improve WebSphere InterChange Server performance. The most important of these are related to garbage collection, setting the Java heap size, and configuring threading parameters. This section deals with these topics in detail.

Note: WebSphere InterChange Server V4.3.0 now requires the use of JRE V1.4.2. Also, as of the WebSphere InterChange Server V4.2.2 release, the product ships with the IBM JVM on Win32® and AIX platforms, and the Sun™ JVM on Solaris and HP systems. Vendor-specific JVM implementation details and settings are discussed as appropriate.

While there are more tuning parameters than are discussed in this section, most are for specific situations and are not of general use. The following URL provides a useful summary of JVM options:

http://java.sun.com/docs/hotspot/VMOptions.html

For a detailed description of the IBM JVM, consult the Java Performance issue of the *IBM Systems Journal*, Vol. 1, 2000:

http://www.research.ibm.com/journal/sj39-1.html

For a useful FAQ about the Sun HotSpot JVM, vread this page:

http://java.sun.com/docs/hotspot/PerformanceFAQ.html#20

11.9.1 Set heap size and nursery size for efficient garbage collection

Garbage collection (GC) is the process of freeing unused objects so that portions of the JVM heap can be reused. Because the Java language specification does not provide explicit delete() or free() byte codes, it is imperative to occasionally detect and delete objects that no longer have active references and free that space for reuse.

Garbage collection is triggered automatically when there is a request for memory, such as for object creation, and the request cannot be readily satisfied from the free memory that is available in the heap (allocation failure). Garbage collection can also be programatically activated using a Java class library System.gc() call. In this case, garbage collection occurs immediately and synchronously.

While the function that is provided by the SUN HotSpot and IBM garbage collectors is the same, the underlying technology is different. For both JVMs, garbage collection takes place in three phases: mark, sweep, and an optional compact phase. The implementation of the garbage collection phases is very different. This is mainly due to the fact that the Sun HotSpot engine is what is known as a *generational collector* while the IBM JVM is not.

A detailed discussion of the HotSpot generational GC can be found here:

http://java.sun.com/docs/hotspot/gc/index.html

With the IBM JVM, the full heap is consumed before a garbage collection is triggered. With the SUN JVM a garbage collection is triggered when either the nursery or the full heap is consumed. Whether a full-heap GC or nursery GC is being performed, the first phase is to mark all referenced objects in the region that is being collected. This leaves all unreferenced objects unmarked and leaves the space they occupy free to be collected and reused. Following the mark phase, free chunks of memory are added to a free list. This phase is referred to as *sweeping*.

Occasionally, following the sweep phase, a compact phase is performed. The compaction moves objects closer together to create larger contiguous free chunks. Several triggers can cause a compaction. For instance, if after sweep there is still not a large enough contiguous chunk of memory, then compaction executes. Also, for most System.gc() calls, a compaction is performed. Relative to the other phases involved, compaction can be a time-consuming process and should be avoided if possible. The IBM JVM has been optimized to avoid compactions, as this is an expensive process.

Monitoring garbage collection

In order to set the heap correctly, you must first determine how the heap is being used. This is easily done by collecting a *verbosegc trace*, which prints garbage collection actions and statistics to stderr. The verbosegc trace is activated by using the Java run-time option of verbose:gc. Output from verbosegc is different for the IBM JVMs and Sun HotSpot, as shown by Example 11-2 and Example 11-3 on page 489.

Example 11-2 IBM JVM verbosegc trace output

```
<AF[8]: Allocation Failure. need 1572744 bytes 5875 ms since last AF>
<AF[8]: managing allocation failure, action=1 (23393256)/131070968) (2096880/3145728)>
<GC: Tue Dec 18 17:32:26 2001
<GC(12): freed 75350432 bytes in 168 ms, 75% free (100840568)/134216696)>
<GC(12): mark: 129 ms, sweep: 39 ms, compact: 0 ms>
<GC(12): refs: soft 0 (age >= 32), weak 0, final 0 , phantom 0>
<AF[8]: completed in 203 ms>
```

Using the IBM JVM output shown in Example 11-2 as an example, the metric following the word `need` is the size of the allocation that caused the garbage collection. On the same line, the amount of time in milliseconds since the last allocation failure is given. The next line with the `<AF[8]>` tag displays the amount of free space in the heap and in the wilderness. In tExample 11-3 on page 489, the line reports `23393256` free bytes out of a possible `131070968` bytes. The `(2096880 / 3145728)` refers to wilderness area free, which is usually ignored.

The next set of lines provides information about the garbage collection that was caused to satisfy the allocation failure. The first line is a time stamp. This is followed by a line that includes the time to complete the GC, `168 ms`, and the amount of free space after the GC, `75%`. Both of these metrics are extremely useful in understanding the efficiency of the garbage collection and the heap usage. Following this line is a line describing the time for the different components of the GC. You should look to make sure that the number following compact is normally 0. That is, a well-tuned heap will avoid compactions. Finally, for the GC, there is a line about soft, weak, and phantom references, as well as a count of finalizers. This is then bracketed by a line with a time for the full allocation failure.

Example 11-3 SUN JVM verbosgc trace output (young and old)

```
[GC 325816K->83372K(776768K), 0.2454258 secs]
[Full GC 267628K->83769K <- live data (776768K), 1.8479984 secs]
```

Setting heap size for most configurations

This section contains guidelines for determining the appropriate Java heap size for most WebSphere InterChange Server configurations. If your configuration requires that many JVMs run concurrently on the same system (for example, if you run multiple adapters on the same physical system as the InterChange Server), then you should also read the next section, “Setting heap size when running multiple JVMs on one system” on page 490.

For many applications, the default heap size setting for the IBM JVM is sufficient for good performance. In general, the HotSpot JVM default heap and nursery size is too small and should be increased. We show you how to set these parameters later. For optimal performance and for applications with unpredictable loads or large live sets, the heap size should be optimized.

There are several approaches to setting the optimal heap sizes. Here, we describe the approach that most applications should use when running the IBM JVM on AIX. The essentials can be applied to other systems. First, we provide more background. There is a feature in the IBM JVMs that deals with dynamically growing the heap that is referred to as *rate-trigger* heap growth. This process attempts to set the size of the heap so that pauses are not too long and GC does not take too much time. This is done dynamically, and it adjusts with the workload. If too much time is being used in GC (fraction of execution time spent in GC > .13), the heap grows. If the heap is mostly free, the heap can shrink.

In order to use rate-trigger heap growth effectively, set the initial heap size (**-ms** option) to something reasonable (for example, 64 MB or 96 MB), and the maximum heap size (**-mx**) option to something reasonable, but large (for example, 256-512 MB). Of course, the maximum heap size should never force the heap to page. It is imperative that the heap always stays in physical memory. The JVM then tries to keep the GC time to something reasonable behind the covers by growing and shrinking the heap. The output from verbosegc should then be used to monitor the GC actions.

A similar process can be used to set the size of HotSpot heaps. In addition to setting the minimum and maximum heap size, you should also increase the nursery size to approximately 1/4 of the heap size. Note that you should never increase the nursery to more than 1/2 the full heap. The nursery size is set using the MaxNewSize and NewSize parameters (`XX:MaxNewSize=128m`, `XX:NewSize=128m`).

After the heap sizes are set, verbosegc traces should be used to monitor the GC actions. If you find something unpleasant from the verbosegc trace, you can modify the heap settings accordingly.

For example, if the percentage of time in GC is too high (greater than 10% of the total time), and the heap has grown to its maximum size, increase that size. Note that this does not always solve the problem because it is normally a memory-over-usage problem. If the pause time is too long, decrease the heap size. If both problems are observed, an analysis of the application heap usage is required.

Setting heap size when running multiple JVMs on one system

Each running Java program has a heap associated with it. Therefore, if you have a configuration in which several Java programs are running on a single physical system, setting the heap sizes appropriately is of particular importance. An example of one such configuration is when many adapters are on the same physical system as the InterChange Server. Each adapter is a separate Java program that has its own Java heap, as does the InterChange Server. If the sum of all of the Java heap sizes plus all other virtual memory usage exceeds the size of physical memory, the heap will be subject to paging. As previously noted, this causes the performance to degrade significantly. To minimize the possibility of this occurring, use the following guidelines:

- Collect a verbosegc trace for each running JVM.

  Based on the verbosegc trace output, set the initial heap size to a relatively low value. For example, assume that the verbosegc trace output shows that the heap size grows quickly to 128 MB, and then grows more slowly to 200 MB. Based on this, set the initial heap size to 128 MB (`ms128m`).

  Based on the verbosegc trace output, set the maximum heap size appropriately. Care must be taken to not set this value too low, or Out Of Memory errors will occur. The maximum heap size must be large enough to allow for peak throughput. Using the above example, a maximum heap size of 256 MB might be appropriate (`mx256m`).

  Be careful to not set the heap sizes too low, or garbage collections will occur frequently, which might reduce throughput. In general, each adapter should have a heap of at least 128 MB. Again, a verbosegc trace will assist in determining this. A balance must be struck so that the heap sizes are large enough that garbage collections do not occur too often, while still ensuring that the heap sizes are not cumulatively so large as to cause the heap to page. This balancing act will, of course, be configuration-dependent.

- Enabling large heaps

  Utilizing Java V1.4.2 (shipped with WebSphere InterChange Server V4.3.0) on AIX it has been shown to achieve a Java heap size over 3 GB, a dramatic improvement over the heap sizes achieved with Java V1.3.1.

  Refer to the following link for details regarding making this AIX configuration change:

  `http://www.ibm.com/servers/esdd/articles/aix4java`

Summary: setting heap sizes

The following list summarizes our recommendations about setting heap sizes:

- Make sure that the heap never pages. That is, on a given system, the sum of all of the JVMs maximum heap sizes must fit in physical memory.
- Collect and analyze a verbosegc trace in order to optimize memory usage.
- Aim for less than 10% execution time spent in garbage collection (GC). Analyze the verbosegc trace in order to determine the GC execution time. Object reuse and heap size tuning can help in this area.
- For optimal performance, the heap should be run with less than 60%, possibly even 50%, occupancy. This is readily determined from the verbosegc trace output.
- Avoid finalizers. A developer can never be guaranteed when a finalizer will run, and often they lead to problems. If you do use finalizers, try to avoid allocating objects in the finalizer code. An IBM JVM verbosegc trace shows if finalizers are being called.
- Avoid compaction where possible. A verbosegc trace shows when compaction is occurring.
- Analyze requests for large memory allocations and then devise a method for reusing the object.
- Increase the size of the nursery for the Sun HotSpot JVM. A good rule is to set the nursery to 25% of the size of the heap.

11.9.2 Set AIX threading parameters

The IBM JVM threading and synchronization components are based on the AIX POSIX–compliant Pthread implementation. The following environment variables in Example 11-4 on page 492 have been found to improve Java performance in many situations and have been used for the workloads in this document. The variables control the mapping of Java threads to AIX Native threads, turn off mapping information, and allow for spinning on mutex locks.

Example 11-4 AIX environment variables

```
export AIXTHREAD_COND_DEBUG=OFF
export AIXTHREAD_MUTEX_DEBUG=OFF
export AIXTHREAD_RWLOCK_DEBUG=OFF
export AIXTHREAD_SCOPE=S
export SPINLOOPTIME=2000
```

More information about AIX-specific Java tuning information can be found here:

http://www.ibm.com/developerworks/eserver/articles/JavaPart1.html
http://www.ibm.com/developerworks/eserver/articles/JavaPart2.html

11.9.3 Use HotSpot server instead of client

The Sun HotSpot JVM can be configured to run as a server or as a client. When configured as a server the Just-In-Time Compiler (JIT) uses extra processor cycles and memory to create more highly optimized code. Because the server is a long-running process, the extra time and memory spent on JIT at initial instantiation is well worth the increased performance during run time.

Therefore, on platforms that ship with the Sun HotSpot JVM, the WebSphere InterChange Server should always be run in server mode. To do this, add the server parameter to the Java invocation.

11.9.4 Setting thread stack size if using many threads

As mentioned in the section about WebSphere InterChange Server threading, Java threads consume memory in the heap. In addition, the threads themselves use virtual memory for their thread stacks. If a configuration is using an excessive number of threads, memory in either place can become a problem. The JVM enables a user to configure the amount of virtual memory to set aside for the thread stack. The default thread stack size is different, depending on the JVM version and the operating system. However, the mechanism to set the value is the same. To set the thread stack size to 128 KB, the parameter -ss128k is passed in on the invocation of the JVM. Care should be taken not to set this value to `small`. It is recommended that at least 128 KB be given to each thread stack, although the system can operate successfully with a lower setting.

11.9.5 Reduce or increase heap size for out-of-memory errors

The java.lang.OutOfMemory exception is used by the JVM in a variety of circumstances, making it sometimes difficult to track down the source of the exception. There is no conclusive mechanism for telling the difference between these potential error sources, but a good start is to collect a trace using verbosegc. If the problem is a lack of memory in the heap, then this is easily seen in this output. Many garbage collections resulting in very little free heap space will occur preceding the exception. If this is the problem, increase the size of the heap.

If, however, there is enough free memory when the java.lang.OutofMemory exception is thrown, the next item to check is the finalized count from the verbosegc (only the IBM JVM will give this information). If these appear high then a subtle effect might be occurring whereby resources outside the heap are held by objects within the heap and being cleaned by finalizers. Reducing the size of the heap can alleviate this situation, by increasing the frequency with which finalizers are run.

11.10 Large objects

A common tuning issue is trying to identify the largest object size that both the WebSphere InterChange Server and the corresponding adapters can process effectively and efficiently. Because there is no one simple answer to this question, we present both a discussion of the issues involved and some practical guidelines for the current release of the WebSphere InterChange Server and the WebSphere Business Integration Adapters.

WebSphere InterChange Server V4.3.0 includes support for larger object sizes. This is made possible by segmenting large business objects into more managable pieces when allocating those objects to memory. For example, segmenting a 5MB object into five 1MB objects in memory.

Controlled laboratory tests have been performed to find the current upper boundary to the maximum object size that the WebSphere InterChange Server V4.3.0 is capable of consistently processing. Observation shows that the breaking point differs, depending primarily on the internal state of the WebSphere InterChange Server and adapters. These tests have shown that in a standalone environment where the particular InterChange Server is dedicated to specific actions, up to a 100 MB object could successfully be processed. Testing in a stressed environment produced consistent results of processing up to a 28 MB object. These tests show a 2X improvement in maximum object size and a 10X improvement in normal environment large object size.

Note: It is recommended that you architect integration solutions to reduce object size whenever possible. Ideally, the solution design should be architected to break the object into multiple objects sized at or below 1 MB, and process them individually. This approach generally produces a more robust and a higher performing solution. When this is not possible, it is recommended to avoid using object sizes in excess of 28 MBs in a stressed production environment. Also, it is important to understand that even though the InterChange Server can support large objects, the adapters might not be able to support the same size objects.

11.10.1 Factors affecting large object size processing

Stated at a high level, the object size capacity for a given installation depends on the size of the Java heap and the load placed on that heap by the current level of incoming work. The larger the heap, the larger the business object that can be processed successfully.

To apply this technically accurate but somewhat general statement, one must first understand that the business object size limit is based on three fundamental implementation facts of Java Virtual Machines.

Java heap size limitations

The limit to the size of the Java heap is operating system dependent. Further details of heap sizes are given later in this section (see Java heap limitations under Section 11.10.2, “Mitigating large object issues” on page 495), but it is not unusual to have a heap size limit of around 1.4 GB. The maximum object size experiments mentioned above were achieved with a heap size on the order of 1.6 GB to 2 GB.

Java heap fragmentation

In a production WebSphere InterChange Server environment, the server and adapter agents may have been operating for a long period before a large object arrives. All JVMs implement a heap compaction function that limits but cannot totally eliminate heap fragmentation. Some objects on the Java heap cannot be moved. As a result, fragmentation will always exist in the heap, potentially making the amount of contiguous memory available in a single block much less than the overall total of available memory. This is normally not an issue, because most Java object requests are small and easily allocated in a fragmented heap.

Contiguity of Java objects

If a large Java object is requested and, after compaction, there is not a sufficiently large contiguous chunk of Java heap available, then Java throws an out-of-memory exception. This is a typical scenario that is encountered with very large business objects, and it brings to light two very important facts:

- First, even though there is plenty of free space available in the heap, an out-of-memory exception can still occur. We have seen instances in the field where half of a 1.6 GB heap was free (800 MB), but a 38 MB allocation caused an out-of-memory exception. This is due to the fact that the Java heap becomes fragmented over time. The impact of this tendency is somewhat alleviated by WebSphere InterChange Server V4.3.0's ability to do Business Object Chunking; but the overall concern still exists.
- Second, there is an issue concerning the relationship between a WebSphere InterChange Server business object (ASBO, GBO) and the Java objects that represent them. An ASBO which appears as a 10 MB message on the MQ input transport might result in the allocation of many larger Java objects as it flows through WebSphere InterChange Server and the adapters. That is, an ASBO will be 10 MB on the input message queue, but might result in allocations of 20 MB to 30 MB on the Java heap. The fact that an ASBO or GBO consumes more of the Java heap than the actual size of the corresponding message is due to the implementation details of the WebSphere InterChange Server (and WebSphere Business Integration Adapters) as well as possible growth in the mappings of objects.

11.10.2 Mitigating large object issues

It has been observed that the use of large objects in an architected solution is frequently associated with batch-oriented business processes, where a large number of smaller transactions are assembled and submitted as a single object for processing. Given that, one must address the three issues outlined in the previous section (Java heap size limitations, Java heap fragmentation, and Contiguity of Java objects),

Java heap limitations

Increase the Java heap to its maximum. This change can be implemented in different ways depending on the operating system.

Windows

On the Windows platforms, WebSphere InterChange Server V4.3.0 and WebSphere Business Integration Adapters V2.6.0 ship with IBM Java V1.4.2 for Windows.

Because of adress space limitations in the Windows operating system, the largest heap that can be obtained is around 1.4 GB to 1.6 GB.

AIX

WebSphere InterChange Server V4.3.0 and WebSphere Business Integration Adapters V2.6.0 support the 32-bit version of IBM Java V1.4.2 for AIX.

Utilizing the normal Java heap settings for IBM Java V1.4.2 for AIX, it has been shown that the IBM JVM V1.4.2 supports a Java heap size in excess of 3 GB. For details, refer to the article *Getting more memory in AIX for your Java applications* available here:

http://www.ibm.com/developerworks/eserver/articles/aix4java/

Solaris

On Solaris, WebSphere InterChange Server V4.3.0 and WebSphere Business Integration Adapters V2.6.0 ship with the 32-bit version of Java V1.4.2 from Sun Microsystems. Laboratory experiments indicate a maximum of approximately a 3.5 GB heap for WebSphere InterChange Server V4.3.0 workloads.

Java heap fragmentation

The only way to reduce significantly the fragmentation in the Java heap and allow for larger object sizes is to limit the concurrent processing activity occurring within the JVM. Obviously, you should not expect to be able to process a steady stream of the largest objects possible concurrently with other WebSphere InterChange Server activities. The operational assumption that must be made when considering large objects is that not all objects will be large and that large objects will not arrive very often, perhaps once or twice per day. If more than one large object is being processed by the WebSphere InterChange Server concurrently, the likelihood of failure increases dramatically.

The size and number of the normally arriving smaller objects affect the amount of Java heap memory fragmentation that exists and also play a role in the processing of large objects. Generally speaking, the heavier the load on a system when a large object is being processed, the more likely it is that memory problems will be encountered.

For adapter agents, the amount of concurrent processing can be influenced by setting the Poll Frequency and Poll Quantity parameters. To allow for larger object sizes, set a relatively high value for Poll Frequency and low value for Poll Quantity this will minimize the amount of concurrent processing that occurs. For a detailed discussion about setting these parameters, see Section 11.4.1, “Configure poll frequency and poll quantity” on page 475.

Contiguity of Java objects

If a large object, one greater than 28 MB, must be processed on current WebSphere InterChange Server and adapter technologies, then the solutions engineer must find a way to limit the large Java objects that are being allocated.

While significant enhancements in the internal processing of large business objects have been delivered in WebSphere InterChange Server V4.3.0 and WebSphere Business Integration Adapters V2.6.0, the business process designer is ultimately responsible for managing the maximum object size processed by the system. The primary technique for doing this is to decompose the large object into smaller objects and submit them individually.

If the large objects are actually a collection of small objects as assumed, the solution is to group the smaller objects into objects less than 1 MB in size. If there are temporal dependencies or an all-or-nothing requirement for the individual objects, then the solution becomes more complex. Implementations have shown that dealing with this complexity is worth the effort, as demonstrated by both increased performance and stability.

An exploration of large object processing

Driven by observations of memory usage patterns within solutions using WebSphere InterChange Server and WebSphere Business Integration Adapters, a study of the ways that memory is used within the server and adapters was initiated. As a result of this study, several enhancements have been incorporated into WebSphere InterChange Server V4.3.0 and WebSphere Business Integration Adapters V2.6.0 that greatly improve their ability to process large business objects. In laboratory experiments using the JText adapter (with the Fixed Width Data Handler) and a simple collaboration, the system has successfully processed 100 MB business objects (as measured by the size on the input delivery queue). Some of the issues discovered in the study are detailed here. However, keep in mind that there are several significant differences between the laboratory and a production environment. The purpose of this memory work was to improve general product reliability, *not* to support the processing of business objects of any particular size. In almost every case, a solution is better off limiting the size of the largest object processed. Many of the most powerful techniques for dealing with memory constraints reside in solution design. Refer to Section 11.10.2, “Mitigating large object issues” on page 495 for several important suggestions.

Several of the efficiency improvements incorporated into the server and adapters are equally applicable to the application code running within the server, or custom data handlers developed for adapters. Some of them are detailed here:

- Reduce overall Java memory use.

  Allocation and initialization of Java objects can be expensive. This is particularly true for larger objects. Both the allocation and initialization cost increases as object sizes increase. Whenever possible, an application should reuse a previously allocated Java object, rather than allocating a new one. Reusing previously allocated Java objects reduces stress on the memory allocation subsystem and improves overall system performance. It is interesting to note that the allocation improvements also drove improvements to core adapter and server throughput.

- Reduce the peak live set.

  A Java server receiving requests for more memory than is available in the heap will begin to throw runtime errors (OutOfMemory exceptions). Many of the problems observed in processing large business objects simply result from exhaustion of the Java heap. One technique that helped to reduce the live set involved explicitly releasing a Java reference when it was no longer required. Frequently, Java programmers wait for a local variable to go out of scope before the object it references is available to be freed by the garbage collector. If the reference is nullified, prior to a method invocation for example, the garbage collector might be able to free that object to satisfy the allocation requests within the called method.

- Reduce the size of the largest contiguous object allocated.

  A new class of objects, CxListBuffer, has been created within the WebSphere InterChange Server and WebSphere Business Integration Adapters packages. This class offers a representation of string-like objects that makes more efficient use of available Java heap space. CxListBuffer offers many of the same services as Java's String and StringBuffer classes, but the internal implementation is significantly different. Rather than storing the string contents in a single character array, CxListBuffer uses a collection of character arrays, thereby limiting the maximum size of any of the individual objects. Because each Java object occupies a contiguous memory region within the Java heap, it is typically easier to allocate ten 1 MB objects than one 10 MB object.

In spite of the enhancements introduced in these releases, some limitations exist in the overall system capacity for processing large objects. One common limitation to the ability to design for maximum capacity occurs at process boundaries. For instance, WebSphere InterChange Server and WebSphere Business Integration Adapters use the WebSphere MQ messaging server for interprocess communication. WebSphere MQ imposes a limit of 100 MB to the size of any message placed on a queue. Additionally, business object expansion can occur in the mappings within a data handler or collaboration. Cases have been observed in which the size of the business event delivered to a queue is significantly different from the size of the internal representation of a business object, or the size of the event on disk (in the case of the JText adapter used for this study). Particular care should be given to the possibility of these expansions at solution design time.

The process boundary limitations might be even more severe when using the JMS delivery transport because the 100 MB message size limitation remains, but there are additional constraints imposed by the JMS protocol itself. As a result, the maximum object size observed when using the JMS delivery transport was less than that seen with the MQ delivery transport.

11.11 Tuning other WebSphere BI runtime components

For information about performance and tuning for WebSphere Business Integration Message Broker and WebSphere MQ Workflow, refer to the performance section of the SupportPacs Web site. New reports are added frequently and existing reports are updated to reflect new functionality in the different runtime engines. These reports can be found here:

`http://www-306.ibm.com/software/integration/support/supportpacs/perfreppacs.html`

Part 4

Appendixes

A

Our hardware and software configuration

This appendix provides information about the hardware and software that was used to create the scenario described in this redbook.

Configuration of client machines

The development client and management client machines were created on the following hardware configuration:

- IBM NetVista PC, Model 6792-MHU
- Pentium® 4 processor running at 1800 MHz
- 1 GB memory
- 40 GB hard disk

Configuration of runtime servers

The runtime servers, including the Web server and the database server, were implemented on the following hardware configuration:

- IBM @server® xSeries® 230, Model 8658-61Y
- Pentium III processor running at 1GHz
- 2 GB memory
- 24 GB hard disk

The software setup for these runtime servers was:

1. Database server

   IBM DB2 V8.2 Server (8.1 with FixPak 7a (WR12342)).

2. WebSphere Application Server

   WebSphere Application Server V5.1
   WebSphere MQ V5.3 and CSD7
   IBM DB2 V8.2 Client (8.1 with FixPak 7a (WR12342))
   WebSphere MQ Workflow V3.5 Web Client feature with service pack 1

3. WebSphere MQ Workflow runtime server

   WebSphere MQ V5.3 and CSD7
   IBM DB2 V8.2 Client (8.1 with FixPak 7a (WR12342))
   WebSphere MQ Workflow V3.5 Web Client feature with service pack 1

4. WebSphere Business Integration Message Broker server

   WebSphere MQ V5.3 and CSD7
   IBM DB2 V8.2 Server (8.1 with FixPak 7a (WR12342))
   WebSphere Business Integration Message Broker V5 with fix pack 4(498270)

5. WebSphere InterChange Server

   WebSphere MQ V5.3 and CSD7
   IBM DB2 V8.2 Client (8.1 with FixPak 7a (WR12342))
   WebSphere InterChange Server V4.3

WebSphere Business Integration Adapters Framework V2.6
WebSphere Business Integration Adapters for JDBC
WebSphere Business Integration Adapters for WebSphere MQ Workflow

B

Configuring LDAP for use with RBAC

This appendix provides instructions for configuring LDAP for use with role-based access control in our scenario.

Configuring an LDAP server as a user registry for RBAC

WebSphere InterChange Server Version 4.3 introduces a new feature called role-based access control (RBAC) that allows one to create multiple users, define roles and operations and assign those roles to users. RBAC can be configured to use a repository such as a database to store the user registry; or it can be configured to work with an existing Lightweight Directory Access Protocol (LDAP) server. In our scenario, we are using IBM Tivoli Directory Server as the LDAP server implementation.

In this section, we will explore the process involved in configuring an LDAP server as a user registry for role-based access control.

The following steps describe how to setup LDAP for RBAC.

1. Install and configure the LDAP server.

   For more information about installing the LDAP server see the IBM Redbook *Understanding LDAP - Design and Implementation*, SG24-4986

   After installing LDAP, configure all the generic items properly.

2. Create a realm called realm1 in the LDAP server, we'll use this realm as the userbase in WebSphere InterChange Server. In addition, we created a user called John Doe to be the realm administrator. Create a user schema which inherits from inetOrgPerson.

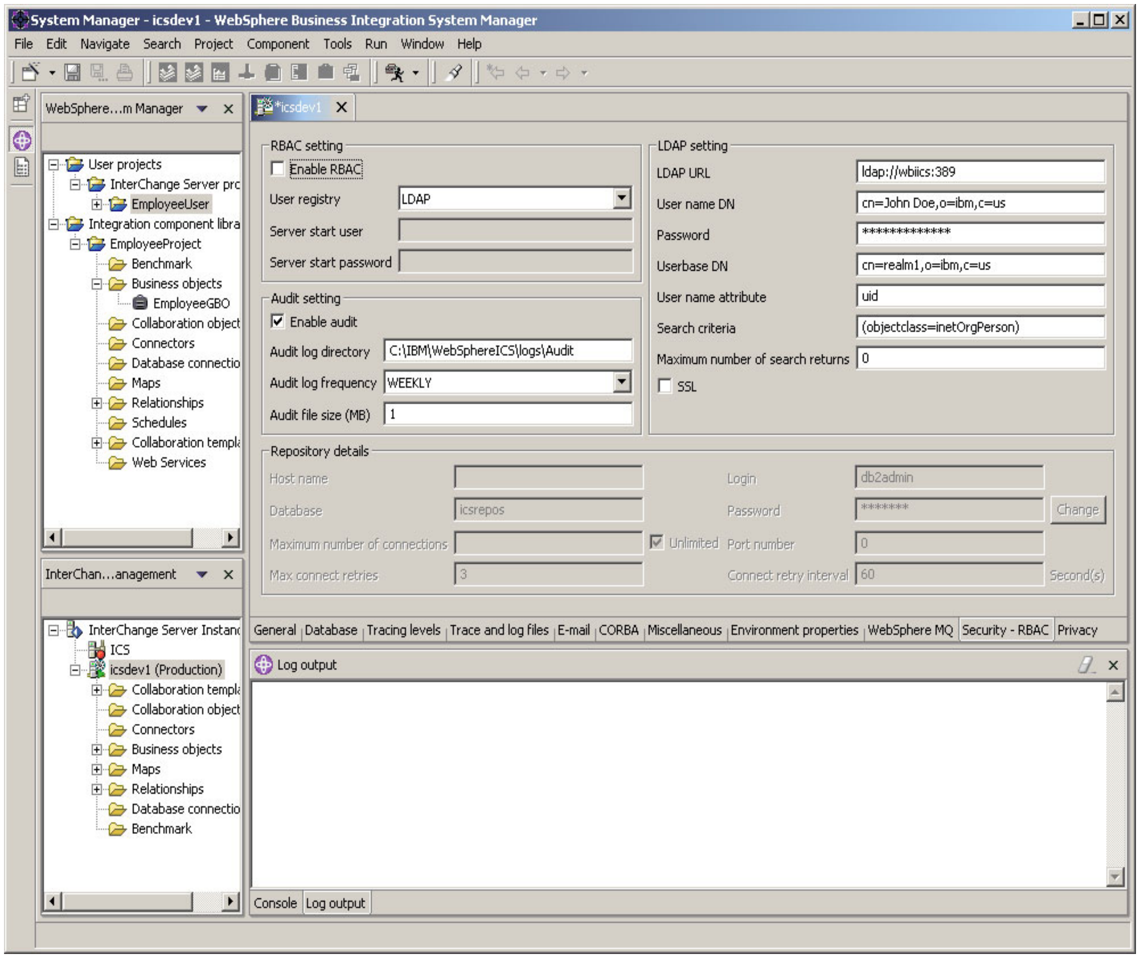
3. Configure RBAC. Start the InterChange Server and connect to the server using the System Manager. In the System Manager server view, right-click the server and select Edit Configuration. On the configuration page, select the Security-RBAC page. (See Figure 11-7 on page 509.)

   a. Configure the LDAP URL property; for example, ldap://localhost:389

   b. Configure the LDAP user name/password: This username and password are used to connect to the LDAP server and retrieve/create user information. This user should already exist and have proper privileges to access the LDAP server. In our scenario, we are using cn=John Doe to connect to LDAP server.

   c. Configure the user base for InterChange Server. The userbase is used to retrieve/create users, it should exist in the LDAP server. We created a user with the following relative distinguished name (RDN™): cn=DEF,cn=realm1,o=ibm,c=us

   d. Configure other optional items.



RBAC setting
- [ ] Enable RBAC
- User registry: LDAP
- Server start user:
- Server start password:

Audit setting
- [✓] Enable audit
- Audit log directory: C:\IBM\WebSphereICS\logs\Audit
- Audit log frequency: WEEKLY
- Audit file size (MB): 1

LDAP setting
- LDAP URL: ldap://wbiics:389
- User name DN: cn=John Doe,o=ibm,c=us
- Password: *************
- Userbase DN: cn=realm1,o=ibm,c=us
- User name attribute: uid
- Search criteria: (objectclass=inetOrgPerson)
- Maximum number of search returns: 0
- [ ] SSL

Repository details
- Host name:
- Database: icsrepos
- Maximum number of connections: [✓] Unlimited
- Max connect retries: 3
- Login: db2admin
- Password: ******* Change
- Port number: 0
- Connect retry interval: 60 Second(s)

Figure 11-7 Configuring LDAP for RBAC

4. Save the configuration.
5. Shutdown the server and restart it.
6. Using the System Manager, connect to the Interchange Server using default user id **admin** and password **null**.
7. Create a user. Right-click the server view and select **Users and Roles.** (See Figure 11-8 on page 510.) Now add a user **DEF** as shown in Figure 11-9 on page 511. Save the page. The new user will be created on LDAP. The user relative distinguished name (RDN) should be cn=DEF,cn=realm1,o=ibm,c=us in our scenario.

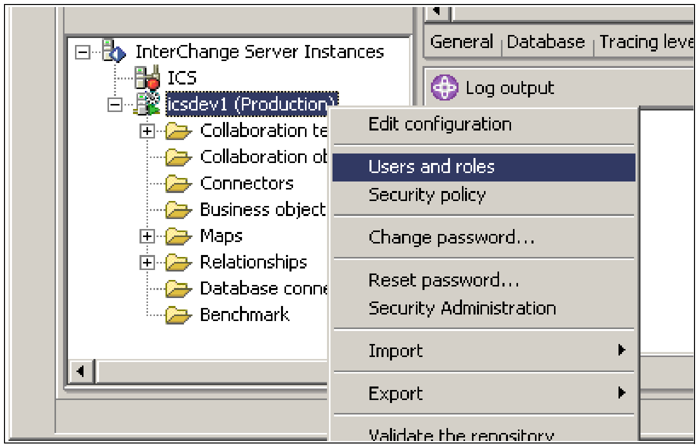


Figure 11-8 RBAC: Getting to Users and roles in System Manager

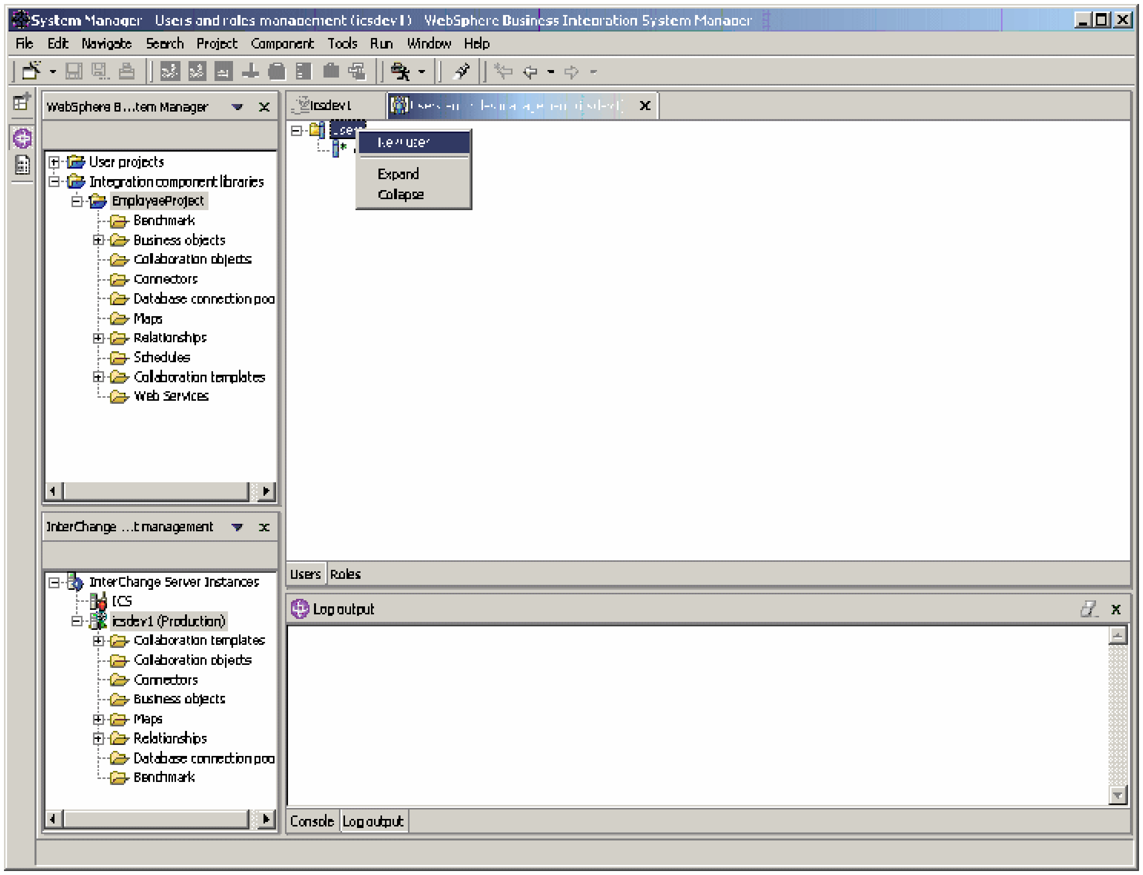
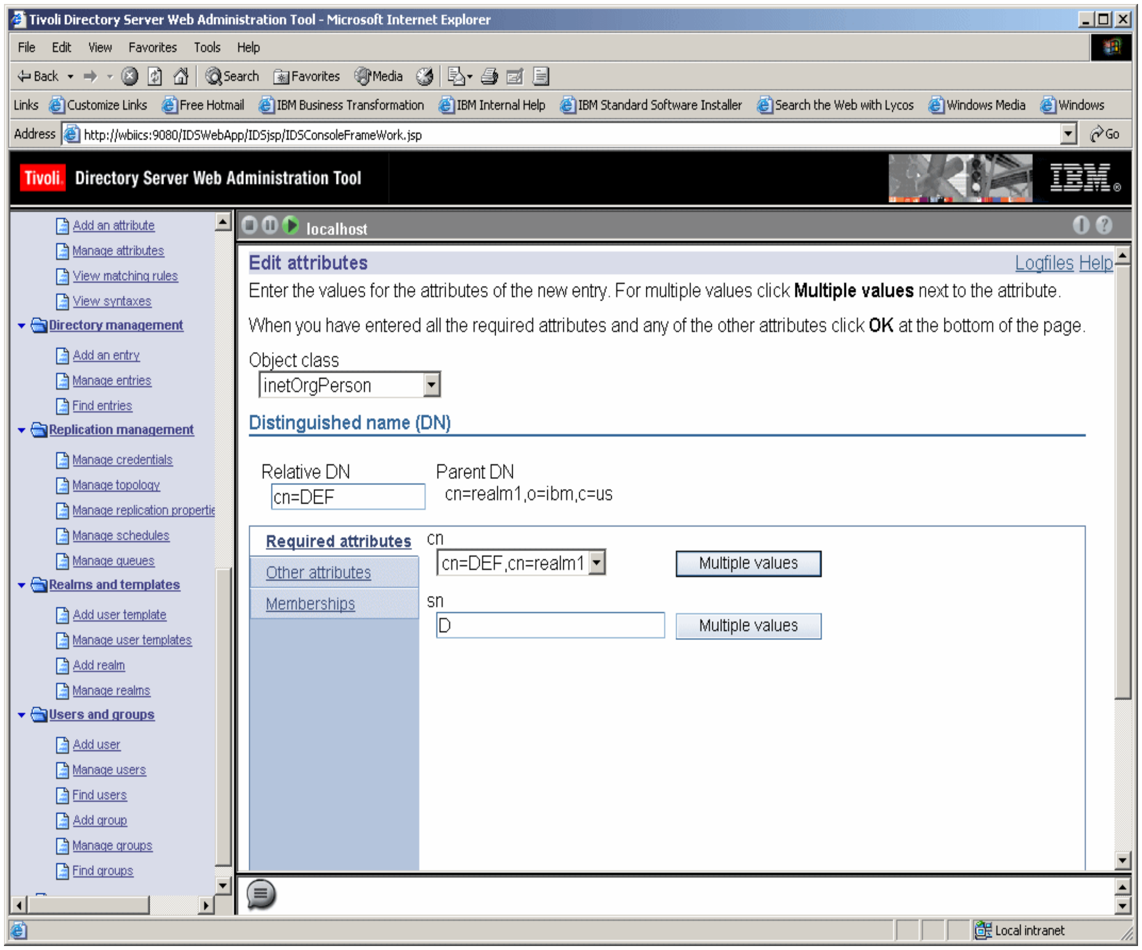


Figure 11-9 RBAC: Add a user

After the user has been created, you can use the Tivoli Directory Server Web Administration Tool to view the newly created user. (See Figure 11-10 on page 512.)



Edit attributes

Enter the values for the attributes of the new entry. For multiple values click **Multiple values** next to the attribute.

When you have entered all the required attributes and any of the other attributes click **OK** at the bottom of the page.

Object class: inetOrgPerson

Distinguished name (DN)

Relative DN: cn=DEF

Parent DN: cn=realm1,o=ibm,c=us

cn: cn=DEF,cn=realm1

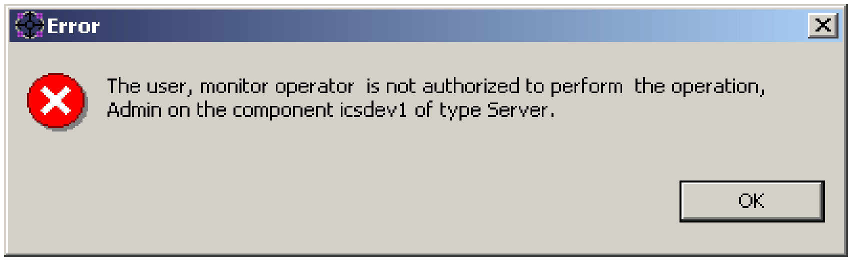
sn: D

Figure 11-10 Verify user created in Tivoli Directory Server

8. Using system manager, add the new user **DEF** to the administrative role.
9. Right-click the server view and select **Edit Configuration.** Click on the Security-RBAC page of the configuration, assign the new user we created to **Server Start User** and **Server Start Password**. Now enable the RBAC by checking the **Enable RBAC** box.
10. Restart the InterChange Server.
11. Open System Manager and connect to the InterChange Server as the new user **DEF.**

12. Now you can add new users and assign the various roles. For example if you create a user called **monitor user** and assign him the **Monitor** role, the user cannot, for example, modify the configuration.

    Figure 11-11 shows the error that is generated if a user with only the Monitor role attempts to modify the configuration.



The user, monitor operator is not authorized to perform the operation, Admin on the component icsdev1 of type Server.

Figure 11-11 A user with Monitor role cannot modify the configuration

Our configuration of LDAP to provide RBAC support for InterChange Server is now complete.

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 516. Note that some of the documents referenced here may be available in softcopy only.

- *Business Integration Management using WebSphere BI Modeler and Monitor: A Real-World Case Study,* SG24-7024
- *Migration to WebSphere Business Integration Message Broker V5,* SG24-6995
- *Patterns: Serial Process Flows for Intra- and Inter-enterprise*, SG24-6305

Other publications

These publications are also relevant as further information sources:

- *Getting more memory in AIX for your Java applications,* written by Sumit Chawla in September 2003 and available from IBM developerWorks at:

  `http://www.ibm.com/developerworks/eserver/articles/aix4java/`
- *IBM WebSphere InterChange Server System Installation Guide for UNIX Version 4.3.0,* (September 2004). See:

  `ftp://ftp.software.ibm.com/software/websphere/integration/wicserver/library/doc/wics43/pdf/installation_unix_30sept04.pdf`

Online resources

These Web sites and URLs are also relevant as further information sources:

- WebSphere MQ Family library

  `http://www.ibm.com/software/integration/mqfamily/library/manualsa/`
- WebSphere Business Integration InfoCenter

http://publib.boulder.ibm.com/infocenter/wbihelp/index.jsp

- WebSphere MQ family SupportPacs

  http://www.ibm.com/software/integration/support/supportpacs

- WebSphere MQ Workflow Support Pacs

  http://www.ibm.com/software/integration/support/supportpacs/product.html#wmqwf

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

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM

Redbooks

Administering and Implementing WebSphere Business Integration Server V4.3

(1.0” spine)
0.875”<->1.498”
460 <-> 788 pages

IBM®

Administering and Implementing WebSphere Business Integration Server V4.3

Redbooks

Implement a business integration infrastructure

Develop and deploy solution components

Manage WebSphere BI infrastructure

This IBM Redbook describes three major phases in a WebSphere Business Integration (BI) project.

- We discuss the planning and system design for a WebSphere BI infrastructure designed to support several business integration projects. We extend the real-life scenario written for another IBM Redbook. Following planning and design, we discuss the implementation of the run-time engines available in IBM WebSphere Business Integration Server V4.3.
- The next phase is developing and testing a business integration solution within our infrastructure. The integration solution combines three run-time engines of WebSphere Business Integration Server V4.3. These engines provide for human interaction, straight-through processing, and message brokering and aggregation.
- The final phase of our WebSphere BI project involves deploying the solution into the production environment, and how to manage this solution. We address issues such as how to coordinate stopping and starting components, and troubleshooting run-time problems. We end by discussing performance tuning in WebSphere Business Integration Server V4.3.

SG24-6647-00 ISBN 0738497266