IBM WebSphere Business Integration Server
Express and Express Plus

IBM

System Implementation Guide

V4.3

IBM WebSphere Business Integration Server
Express and Express Plus

System Implementation Guide

V4.3

Note!

Before using this information and the product it supports, read the information in “Notices” on page 285.

14May2004

This edition of this document applies to IBM WebSphere Business Integration Server Express, version 4.3, IBM WebSphere Business Integration Server Express Plus, version 4.3, and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, e-mail doc-comments@us.ibm.com. We look forward to hearing from you.

Contents

New in this release

New in release 4.3

This is the first release of this guide.

About this document

The products IBM(R)WebSphere Business Integration Server Express and IBM(R) WebSphere Business Integration Server Express Plus are made up of the following components: InterChange Server Express, the associated Toolset Express, CollaborationFoundation, and a set of software integration adapters. The tools in the Toolset help you to create, modify, and manage business processes. You can choose from among the prepackaged adapters for your business processes that span applications. The standard processes template--CollaborationFoundation--allows you to quickly create customized processes.

Except where noted, all the information in this guide applies to both IBM WebSphere Business Integration Server Express and IBM WebSphere Business Integration Server Express Plus. The term WebSphere Business Integration Server Express and its variants refer to both products.

This document provides an overview of the tasks for implementing a business integration system with WebSphere InterChange Server Express.

Audience

This document is for developers and others responsible for developing and implementing the components of a business integration system.

Related documents

The complete set of documentation available with this product describes the features and components common to all WebSphere Business Integration Server Express installations, and includes reference material on specific components.

You can download, install, and view the documentation at the following site: http://www.ibm.com/websphere/wbiserverexpress/infocenter

Note: Important information about this product may be available in Technical Support Technotes and Flashes issued after this document was published. These can be found on the WebSphere Business Integration Support Web site. Select the component area of interest and browse the Technotes and Flashes sections: http://www.ibm.com/software/integration/websphere/support

Typographic conventions

This document uses the following conventions:

`courier font`	Indicates a literal value, such as a command name, filename, information that you type, or information that the system prints on the screen.
bold	Indicates a new term the first time that it appears.
italic, *`italic`*	Indicates a variable name or a cross-reference.
blue text	A blue outline, which is visible only when you view the manual online, indicates a cross-reference hyperlink. Click inside the outline to jump to the object of the reference.

`{ }`	In a syntax line, curly braces surround a set of options from which you must choose one and only one.
`[ ]`	In a syntax line, square brackets surround an optional parameter.
`...`	In a syntax line, ellipses indicate a repetition of the previous parameter. For example, `option[,...]` means that you can enter multiple, comma-separated options.
`< >`	In a naming convention, angle brackets surround individual elements of a name to distinguish them from each other, as in `<server_name><`*`connector_name`*`>tmp.log`.
`/, \`	In this document, backslashes (\) are used as the convention for directory paths. All IBM WebSphere Business Integration Server Expres product pathnames are relative to the directory where the WebSphere Business Integration Server Express product is installed on your system.
`%text%` and *`$text`*	Text within percent (%) signs indicates the value of the Windows `text` system variable or user variable.

Using this guide

This guide is designed to:

- Introduce the high level stages involved in an implementation of WebSphere InterChange Server Express, from assessment of customer requirements to deployment of a live system
- Provide a sequential guideline to the tasks that developers perform in developing and configuring the components and interfaces of a WebSphere InterChange Server Express solution
- Provide a hub document from which to navigate to appropriate development or system documents for specific development tasks

For some tasks that might be performed by both developers of WebSphere InterChange Server Express components and administrators of a live, deployed system, this guide refers to the *System Administration Guide*.

Chapter 1. Overview of IBM WebSphere Business Integration Server Express

This chapter provides an overview of the architecture, components, and processing flow of a business integration system that uses IBM WebSphere Business Integration Server Express or Express Plus. It contains the following sections:

The InterChange Server Express model

At the highest level, the business integration system implemented by IBM WebSphere Business Integration Express and Express Plus is a hub-and-spoke infrastructure. Application-independent business logic resides at the hub, in an integration broker named InterChange Server Express. At the spokes are a set of adapters and the applications or technologies to which they connect. The heterogeneous business applications at the spokes exchange data in the form of business objects, which are passed through business logic at the hub.

The design utilizes modular components, many of which are customizable. Tools are provided both for developing and customizing components, and for administering a system after deployment.

The subsequent topics in this chapter describe the components and flow of information.

Collaborations, business objects, and connectivity

The following list describes the primary roles of the components of the IBM WebSphere Business Integration Server Express system:

- Business-process logic resides in **collaborations** at the hub.

  Collaborations are software modules that contain logic that describes a distributed business process. There are different collaborations for different fundamental business processes—for example, a ContactManager collaboration, or an InventoryMovement collaboration. Collaborations coordinate the functionality of business processes for disparate applications and enable data exchange between them. Collaborations are the hub; through them, data in the form of business objects is exchanged with the spokes.
- Data is exchanged between the hub and the spokes in the form of **business objects**.

Business objects are the messages used by the IBM WebSphere Business Integration Server Express system for exchanging data. **Data handlers** are used to transform serial application data into business objects, and **maps** are used between a business object that is structured for the data model of a specific application and a business object that is generically structured for use by collaborations at the hub.

- **Adapters** supply connectivity to applications or other programmatic entities at the spokes.

  Certain adapters, used for interactions with specific technologies, are supplied with both IBM WebSphere Business Integration Server Express and Express Plus. Other adapters, used for interactions with specific applications, can be obtained from the Adapter Capacity Pack. The Adapter Capacity Pack is separately available as an optional addition to IBM WebSphere Business Integration Server Express Plus.

  Each individual adapter includes a **connector** component that is responsible for linking an application or technology to InterChange Server Express. Many adapters also include an Object Discovery Agent (ODA), which you can use to generate business objects that are specific to the application with which the connector interacts.

  Each connector consists of two parts—the **connector controller** and the **connector agent.** The connector controller interacts directly with collaboration objects and resides on a server that has implemented the IBM WebSphere Business Integration Server Express system (the hub in a hub-and-spoke relationship). The connector agent interacts directly with an application, and can reside with that application on any server on the network.

  Application adapters—for example, the Adapter for PeopleSoft— are intermediaries between collaborations and applications. These connectors transform data from the application into business objects that can be manipulated by the collaborations, and transform business objects from the collaborations into data that can be received by the specific application.

  Technology adapters, such as the Adapter for XML, are designed for interactions that conform to specific technology standards.

- The **Server Access Interface** makes it possible for remote spoke sites that do not implement InterChange Server Express to use **access clients**, which make calls over the Internet to a hub site that does have InterChange Server Express.

  The Server Access Interface is part of the InterChange Server Express. It is a CORBA-compliant API that accepts synchronous data transfers from either internally networked or external sources. The data is then transformed into business objects that can be manipulated by a collaboration. The Server Access Interface makes it possible to receive calls from external entities—for example, from web browsers at remote sites—that do not come through connector agents, but instead come through web servlets into the Server Access Interface.

  The Server Access Interface and the connectors both make use of data handlers. In the IBM WebSphere Business Integration Server Express environment, new data handlers can be created from a modular group of base classes called the **Data Handler Framework**. Also provided is a **Protocol Handler Framework**. These frameworks make it easier to customize solutions and add connectivity for additional data formats and protocols in the future.

Example implementation solutions

A typical IBM WebSphere Business Integration Server Express solution includes one or more collaborations and a set of business objects that represent business information relevant to an enterprise. The collaborations and business objects are used with connectors, with the Server Access Interface, or with both. Solutions can be implemented for applications distributed on a network, and depending on the requirements and components available, for applications that reside across the Internet.

Integration on a network

In this example, the goal is to automatically update an enterprise resource planning (ERP) application when customer information changes in a customer interaction management (CIM) application that resides on the same network. The IBM WebSphere Business Integration Server Express solution might consist of the hypothetical CustomerSync collaboration, connectors for the CIM application and the ERP application, and definitions of business objects that represent customer information. Figure 1 illustrates that solution.

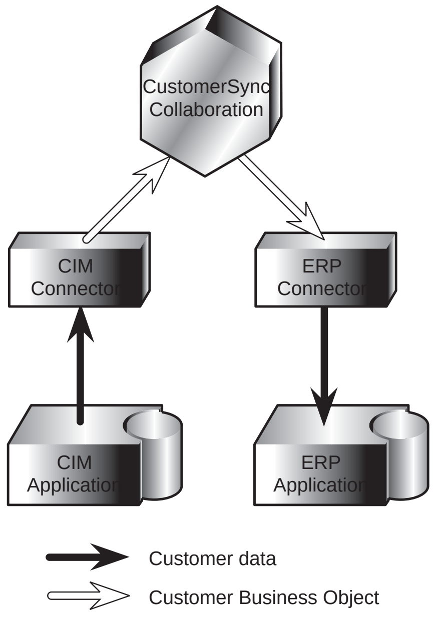


Figure 1. CIM-to-ERP Customer Data solution

Connectivity over the Internet

Data exchange across the Internet can be implemented using the Server Access Interface and certain technology adapters. In this approach, the Server Access Interface is used to pass synchronous calls into the IBM WebSphere Business Integration Server Express; connectors that use Internet technology standards are used to send data from the IBM WebSphere Business Integration Server Express:

- Calls from external browsers (or other sources over the Internet) can be received by a web servlet that sends the calls through the Server Access Interface to trigger business processes in collaborations.
- On behalf of collaborations, connectors for specific Internet technology standards (such as XML) engage in request/response interactions with external destinations that understand the specific data format.

The Server Access Interface resides at a hub site on the InterChange Server Express (ICS). When the Server Access Interface receives a call, it sends the data to a handler for that specific data format (such as the XML data handler). The data handler transforms the data into a generic business object. The Server Access Interface then sends the business object to a collaboration. The collaboration performs its processes on the business object and responds, and that response is transformed back into the specific data format that was used at the beginning of the process.

To accept calls from external processes and send the calls as business objects to a collaboration, the Server Access Interface requires the following:

- A web server that uses a servlet or other process to send data to the Server Access Interface. The data must be in a MIME type for which data handlers have been configured in the InterChange Server Express. The servlet uses the Server Access Interface to call a data handler that converts the data into a business object format, and then uses the Server Access Interface to send the business object as a call into a collaboration.
- A collaboration that has been configured to handle requests that are received as calls through the Server Access Interface.

In the example shown in Figure 2,, at a site that has not implemented either the IBM WebSphere Business Integration Server Express system or a connector, a customer representative uses a web browser to obtain the status of a purchase order over the Internet from an ERP application (SAP in the example) that resides at a site that has implemented the IBM WebSphere Business Integration Server Express system. To enable this, the IBM WebSphere Business Integration Server Express uses the Server Access Interface, together with a collaboration (hypothetical in this example) for purchase-order business logic, an SAP connector, and definitions of business objects that represent purchase-order status information.

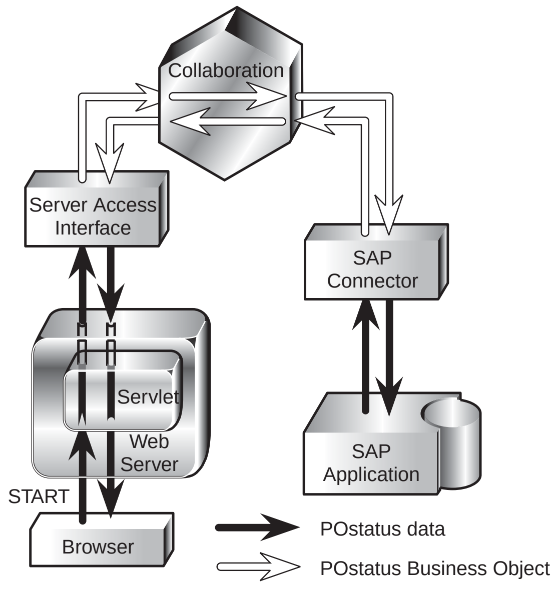


Figure 2. Execution of a call through the Server Access Interface

Data flow in an InterChange Server Express implementation

A data flow is initiated by either of two types of interactions: publish-and-subscribe interactions or service call interactions. Both types of interaction supply triggers that start the execution of a collaboration's business processes. The collaboration then uses a third type of interaction—request/response —to complete the exchange of data with the intended destination.

Publish-and-subscribe interactions

Connectors and collaborations use a **publish-and-subscribe** interaction to move information about application events into the IBM WebSphere Business Integration Server Express for processing.

In the publish-and-subscribe interaction, a collaboration begins its business process when it is triggered to do so by receipt of a business object for a particular type of **triggering event**—for example, Employee.Create—that represents an application operation. The name of the business object (*Employee*) indicates a type of business entity. The verb (*Create*) indicates an operation that occurred on that entity. Therefore, the Employee.Create event reports the creation of an employee entity.

The publish-and-subscribe interaction enables a triggering event to reach a collaboration as follows:

- A collaboration **subscribes** to the event that can trigger its execution. A collaboration subscribes to an event by requesting it, and then waiting for it. A collaboration that subscribes to Employee.Create starts executing when the business object for the Employee.Create event arrives.
- An event occurs in an application, and the event is detected by the application connector's **event notification** mechanism. The connector supplies the event to one or more collaborations by publishing it—making it available as a business object.

Depending on the connector, an event can be published to a collaboration either asynchronously or synchronously. In addition, if the long-lived business process feature of the collaboration is enabled, a collaboration can maintain the event in a waiting state, in anticipation of incoming events satisfying pre-defined matching criteria.

Access requests

A collaboration can be designed to be triggered by direct calls that are sent by an **access client**, received by the Server Access Interface, and sent to the collaboration as business objects. In an InterChange Server Express implementation, calls sent to collaborations through the Server Access Interface are referred to as **access requests**. Access requests can originate from external sources or from sources that are configured within the InterChange Server Express implementation.

Access request interactions are useful when synchronous communication is important—as when, for example, a customer representative uses a web browser to request inventory status information over the Internet.

Request/response interactions

A collaboration begins processing data when it is triggered to do by receipt of a triggering business object. The triggering business object can be the result of either an access request or an event notification. Once a collaboration has been triggered, it can make requests of connectors to which it has been bound, and receive responses.

The collaboration makes its requests—referred to as **service call requests**—in the form of generic business objects. The connectors transform the generic business objects into data entities that are understood by the specific application or data format for which the connector is designed.

The responses that the collaboration receives from the connectors—referred to as **service call responses**—can be business objects containing business data (in the case of retrieve requests) or status reports (successful or unsuccessful).

Figure 3 shows a simplified view of a business process that is initiated through a publish-and-subscribe interaction and completed through request/response interactions. The example shows a distributed business process that automatically generates an invoice when a customer service representative finishes working on a case. In this example, a hypothetical service billing collaboration uses connectors to exchange business data with three different applications.

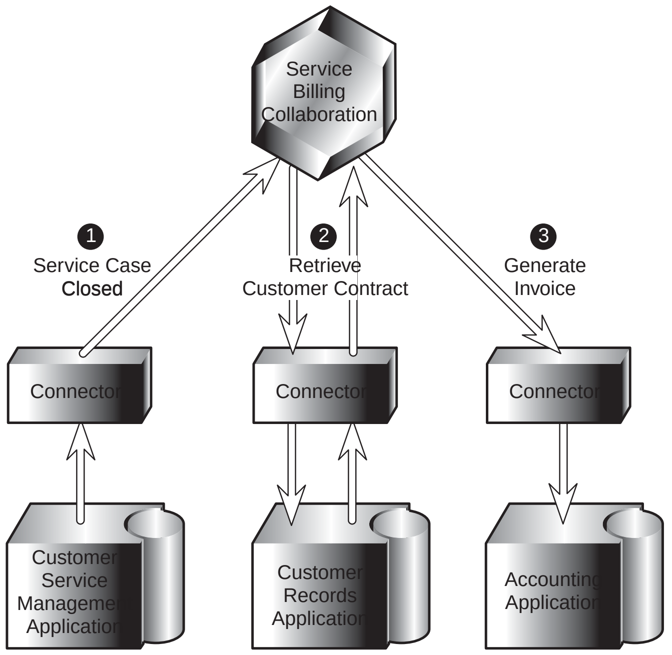

Service Billing Collaboration
1
Service Case Closed
2
Retrieve Customer Contract
3
Generate Invoice
Connector
Connector
Connector
Customer Service Management Application
Customer Records Application
Accounting Application

Figure 3. A publish-and-subscribe, request/response interaction

Figure 3 illustrates the following sequence in the business process:

1. A customer service representative completes work on a case. The connector detects the case closure as an event in the customer service management application and retrieves the relevant case data. The connector then publishes the event, making it available to a collaboration that has subscribed to it. These actions comprise a publish-and-subscribe interaction.
2. In order to compute the invoice amount, the collaboration needs the terms of the customer contract. The collaboration sends a service call request to retrieve the necessary data from the connector for the customer records application. The connector responds to the request. These actions comprise a request/response interaction between the collaboration and the customer records application and connector.
3. Using both the case information and the customer contract, the collaboration produces the information needed to generate an invoice. It sends the invoice creation request to the connector for the accounting application, which forwards the request to the application itself, and responds to the collaboration with a notification of success or failure. These actions comprise a request/response interaction between the collaboration and the accounting application and connector.

A site can tune the closeness with which collaborations and connectors are coupled. The collaboration might, for example, execute on a 24-hour basis, sending requests to a connector that communicates with its application only between midnight and 2:00 A.M. The collaboration could be designed and configured to send the requests without requiring a response and simply process the responses when they come.

Alternatively, a collaboration that has been enabled for long-lived business processes can save the flow context of a request, and send the request with a timeout value, specifying the period of time in which a response can cause the saved processing flow to resume.

Connectors

Connectors are supplied as components of adapters. A connector provides distributed translation services for the IBM WebSphere Business Integration Server Express system, moving data between collaborations and either:

- An application, or
- A programmatic entity—a remote web server, for example—that understands a technology standard, such as XML, that is handled by a connector

A connector has a distributed structure:

- The **connector controller** interacts directly with collaborations and runs as a component within the InterChange Server Express process.
- A **client connector framework** runs as a separate process from the InterChange Server Express and, together with an application-specific component, interacts directly with an application or other programmatic entity. In this guide, the client connector framework and the application-specific component are together referred to as the **connector agent**.

The two parts of a connector can run on the same system or on two different systems. The connector controller runs as part of the InterChange Server Express and so resides on that system. However, the connector agent can reside on any system on the network from which it can communicate with both its application and the connector controller.

Connector communication with applications

There is one connector for each version of an application. Each connector is unique, because it communicates with its application according to the application's interfaces.

To detect application events in which collaborations are interested, a connector polls the application or uses the application's event callback notification mechanism, if there is one. A connector can also interact with the application at the command of a collaboration or to verify the results of its previous requests.

Collaborations

In an IBM Websphere Business Integration Server Express implementation, the term **collaborations** refers to software modules that contain code and business process logic that drives interactions between applications. A collaboration can be simple, consisting of just a few steps, or complex, involving several steps and other collaborations.

A collaboration object can be bound to another collaboration object to form a **collaboration group**. A collaboration group harnesses the power of multiple discrete collaborations to integrate related business processes.

When you install a collaboration, you install a **collaboration template**. A collaboration template contains all of the collaboration's execution logic, but it is not executable. To execute a collaboration, you must first instantiate a **collaboration object** from the template. The collaboration object becomes executable after you configure it by binding it to connectors or to other collaboration objects, and by specifying other configuration properties.

A collaboration has a set of interfaces to the outside world called **ports**. In a collaboration template, each port is a variable that represents a business object that the collaboration object receives or produces at runtime. For example, consider a hypothetical Service Billing collaboration that is triggered by the closing of a case in a customer service management application. The collaboration retrieves the customer's contract as a business object from a customer records application, then uses the information in the contract to send an invoice business object to an accounting application.

Such a collaboration might use three ports, one for each of the connectors with which it interacts. Each port is associated with a particular type of business object, as shown in Figure 2-3:

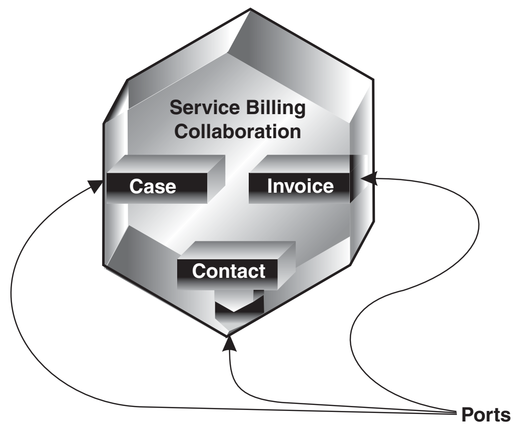


Figure 4. Ports in the Service Billing collaboration

At configuration time, an administrator creates a collaboration object that contains the template's ports. The administrator **binds** the ports for that specific collaboration object, associating each port with a connector, or with another collaboration object. For some collaborations (those whose templates accept the Retrieve verb), a port can be configured instead as *External*, so that it can receive a triggering business object from an access request—an external call on the Server Access Interface.

The ports enable communication between bound entities, so that the collaboration object can accept the business object that triggers its business processes, and then send and receive business objects as requests and responses.

Business objects

Collaborations and connectors interact by sending and receiving **business objects** through InterChange Server Express.

A business object reflects a data entity—a collection of data that can be treated as an operative unit. For example, a data entity can be equivalent to a form, inclusive of all of the form's fields. The form might typically be used in an application, or over the Web, to contain business information about customers, or employees, or invoices.

Business objects are cached in memory during collaboration execution for fast access, and also stored in a persistent transaction state store to provide robust recovery, rollback, and re-execution of collaborations upon server restarts after failures.

The IBM WebSphere Business Integration Server Express system creates business objects that reflect the information contained in entities. In this documentation, a data entity is often referred to in the context of the kind of business information it contains—for example, an *employee entity* or a *customer entity*.

This section provides a first look at business objects. More detail is provided in the *Business Object Development Guide*.

Roles of a business object

A business object can act as an event, a request, or a response.

Event

A business object can report the occurrence of an **application event**, an operation that affected a data entity in an application. The application event might be the creation, deletion, or change in value of that collection of data. When a connector detects an application event and sends a business object to an interested collaboration, the business object has the role of representing the event, and so it is called an **event** in the IBM WebSphere Business Integration Server Express system.

For example, a connector might poll an application for new employee entities on behalf of a collaboration. If the application creates a new employee entity, the connector sends an event business object to the collaboration.

Request

Requests are typically generated in one of two ways:

- A collaboration can send a business object as a **request** to a connector, instructing the connector to insert, change, delete, or retrieve some data in an application. For example, in the service billing collaboration illustrated in Figure 3, the collaboration sends two business objects to connectors, one to retrieve a contract and one to create an invoice. Both are requests.
- The Server Access Interface can send a business object as a request to a collaboration, if that collaboration has been designed or customized to accept the Retrieve verb as a trigger.

Response

When a connector finishes processing a request, it usually returns a **response**. For example, after a connector receives a request to retrieve employee data from an application, it sends a business object containing the employee data.

Structure of a business object

A business object is a self-describing unit that contains a type (its name), processing instructions (a verb), and data (attribute values). Figure 5 is an example of a simple business object, showing its type, verb, and attribute values.

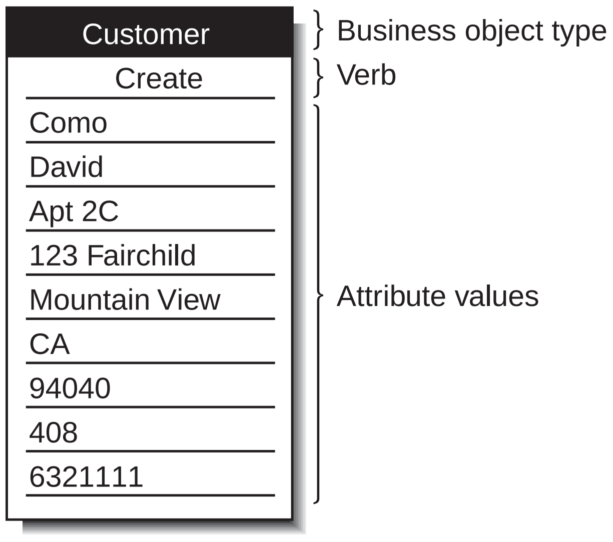


Figure 5. Business object components

The next sections describe these components.

Business object type

Each business object has a type name that identifies it within the IBM WebSphere Business Integration Server Express system. This type is defined by the business object definition. For example, the type might be Customer, Employee, Item, or Contract.

Business object verbs

A business object verb specifies an action in relation to the attribute values. The verb can indicate various types of actions, depending on the role of the business object. Table 1 lists the three business object roles and describes the meaning of the verb in a business object that has each role.

Table 1. Meanings of business object verbs

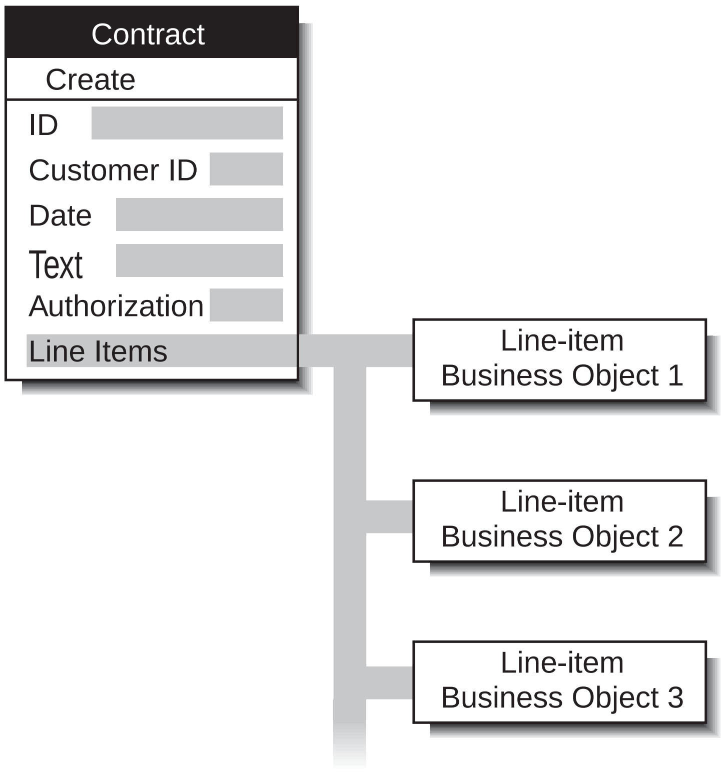
Role of business object	Meaning of verb
Event	Describes what happened in an application. For example, in an event, the Create verb indicates that the source application created a new data entity.
Request	Tells the connector how to interact with the application in order to process the business object. For example, the Update verb is a request to the connector to update the data entity.
Response	Provides the results of a previous request. For example, in a response, the Retrieve verb indicates that the connector obtained the attribute values from the application.

Note: The naming convention is to use the format *business-object-type.verb* to indicate a particular type of business object with a particular verb. For example, Customer.Create is a Customer business object with the Create verb.

Business object attribute values

A business object contains **attribute values** that represent data fields associated with the data entity, such as Last Name, First Name, Employee ID, or Invoice Status.

Some attributes, instead of containing data, contain **child business objects** or **arrays of child business objects**. Figure 6 illustrates the structure of a Contract business object. The Line Item information in the contract is in an array of child business objects.



Contract
Create
ID
Customer ID
Date
Text
Authorization
Line Items
Line-item Business Object 1
Line-item Business Object 2
Line-item Business Object 3

Figure 6. Business object with child business objects

A business object that contains child business objects or arrays of child business objects is a **hierarchical business object**. One whose attributes contain only data is a **flat business object**.

Application-specific and generic business objects

The IBM WebSphere Business Integration Server Express system includes two kinds of business objects: application-specific and generic.

- An **application-specific business object** reflects data entity attributes and the data model of a specific application or other programmatic entity.
- A **generic business object** contains a set of business-related attributes that are common across a wide range of applications, not tied to any specific application's data model

When the connector agent (through its application-specific component) detects an application event such as an update, it retrieves the appropriate data entity from the application and transforms it into an **application-specific business object**.

Note: When this document refers to a business object whose name includes an application name, such as Oracle_Customer, it refers to an application-specific business object. An Oracle_Customer business object, for example, contains the set of information that the Oracle application stores about a customer. In another application, a customer entity might store a somewhat different set of information, store the information in a different order or format, or have a different name.

After the connector agent has built an application-specific business object, it sends the business object to the connector controller in the InterChange Server Express.

The connector controller exchanges business objects between collaborations and the connector agent. Collaborations are generally application-neutral, so the business object that a connector controller exchanges with a collaboration must be a **generic business object.** The use of generic business objects enhances the reusability of the collaboration because its business logic is not bound to specific versions of specific applications.

Note: Generic business object names do not include a company name or product name. Examples include Contact, Employee, and Customer.

Figure 7 shows where the two kinds of business objects fit within the IBM WebSphere Business Integration Server Express system: the collaboration interacts using generic business objects and the connector agent supports business objects designed for specific applications.

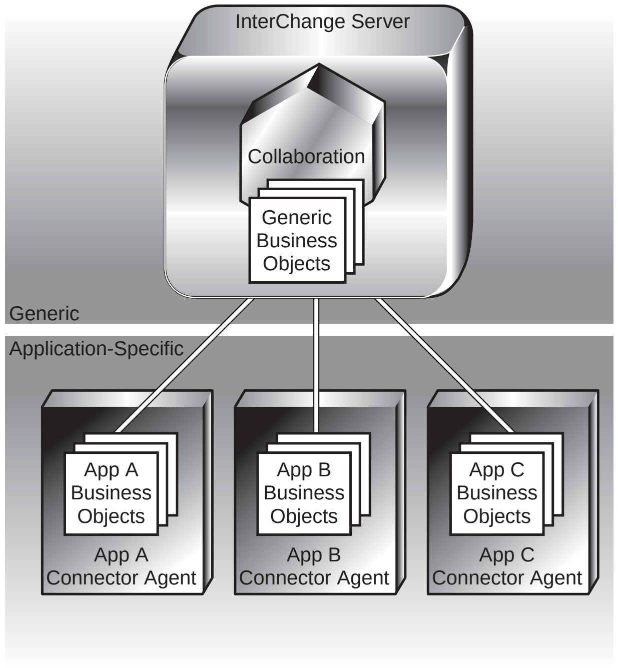


Figure 7. Generic and application-specific business objects

You can use the same connector to run multiple types of collaborations, if the connector supports the business objects used by those collaborations.

Binding between elements

To perform a business process, a collaboration can communicate with connectors, with other collaborations, and with external processes from which it receives access requests through the Server Access Interface. **Binding** is used to set up communications between the collaboration and these elements when the collaboration is configured.

This section describes the concepts for binding. The tools and procedures for creating the bindings are described later in this guide underChapter 9, "Configuring collaboration objects," on page 155).

Binding a trigger

A collaboration begins a processing flow when it is triggered to do so by the arrival of a business object. The trigger can be any of the following:

- A business object from an event published by a connector in a publish-and-subscribe interaction
- An access request received through the Server Access Interface
- A service call from a connector

To allow a collaboration's business processes to be triggered, you must bind the collaboration at configuration time to an element that will supply the triggering event or call.

Binding is done between the collaboration and any element that will participate in that collaboration's business process, but only one element can be bound as the trigger.

Binding for events

When you configure a collaboration at your site to be triggered by events, you bind it to a connector capable of publishing the triggering event. For example, you might specify that the collaboration's Employee.Delete event comes from the PeopleSoft connector.

To look more closely at this relationship, recall that the connector actually consists of a connector controller (which, like the collaboration, resides in the InterChange Server Express) and a connector agent (which includes the client connector framework and an application-specific component, and is separate from the InterChange Server Express). The connector controller maintains the binding information and provides its connector agent with a list of events to which collaborations have subscribed.

When relevant application operations occur, the connector agent publishes the events on that list to the connector controller. The connector agent sends an event to the connector controller without knowing anything about its ultimate destination at a collaboration.

The connector controller is therefore an intermediary between the connector agent and the collaboration, as Figure 8 on page 14 illustrates.

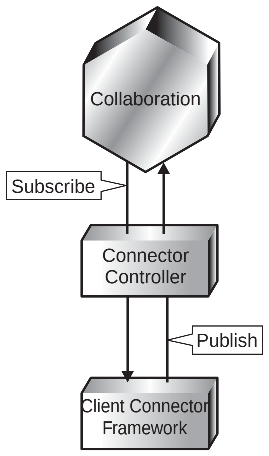


Figure 8. A connector providing a triggering event

Multiple collaborations can subscribe to the same event. When the connector controller publishes the event, it can publish simultaneously to all subscribers.

Binding to receive access requests

Instead of binding the collaboration to a connector for triggering, you can specify that the collaboration will receive access requests from external processes as triggers.

Binding destinations

In addition to binding collaborations to triggering elements, you also bind collaborations to the destination elements with which the collaborations will engage in request/response interactions. Destination elements can be either connectors or other collaborations. A single collaboration can be bound to multiple destination elements.

Data mapping

Figure 7 shows different types of business objects at each application and at the collaborations. The IBM WebSphere Business Integration Server Express system, therefore, must convert business objects into like forms so that it can send events and data across applications and collaborations. **Data mapping** is the process of converting business objects from one type to another. Data mapping is required whenever the IBM WebSphere Business Integration Server Express system sends data between a source and a destination that does not share exactly the source's data model.

Unlike custom application integration solutions that map data directly from one application to another, InterChange Server Express collaborations generally use the generic business object between the application-specific data models. The generic business object serves as a common, cross-application data set. If you change applications in the future, you need only get a new connector and map the new application-specific business object to the generic business object. Collaborations then continue to work as they did previously.

Whenever a collaboration transfers a business object across dissimilar applications, mapping transforms the business object to and from the common data set. Business object transformation takes place:

- From application-specific to generic when business objects pass from connectors to collaborations
- From generic to application-specific when business objects pass from collaborations to connectors

Each connector controller manages the mapping of business objects that pass between its connector agent and the InterChange Server Express. The maps themselves are created through the use of the mapping tools—Map Designer Express and Relationship Designer Express. These tools let you create and modify detailed mapping specifications and execute mapping at runtime.

A connector controller invokes the mapping function when it receives business objects that require mapping. Figure 9 illustrates the invocation of mapping from the connector controller.

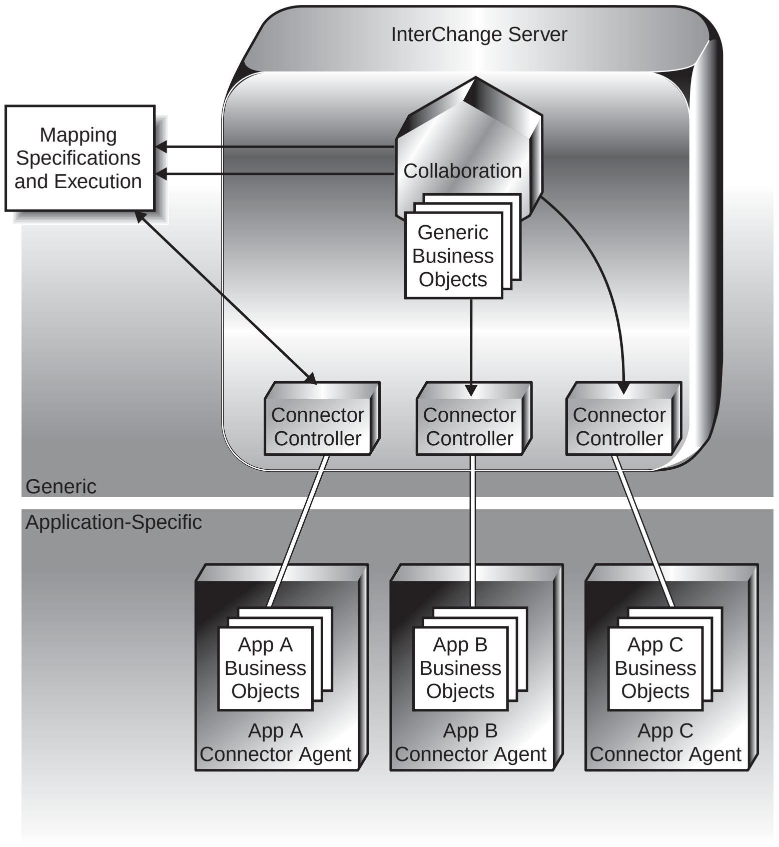


Figure 9. Mapping control and execution

Types of data transformations

A **map** associates a source business object with a destination business object, and contains a series of transformation steps—one for each attribute that is being transformed. Each transformation step contains Java code that calculates the value of the attribute. You need a map for every business object that you intend to transfer between different applications. When you modify business objects, you might also need to modify the associated maps. You typically create one map for each source business object you want to transform.

Within a data map, the conversion of source to destination attributes can be simple, or it can require establishing and maintaining relationships between data entities that are equivalent but which are represented differently in each different application, and so cannot be directly transformed.

For example, for a Country attribute, one application might use a two-letter code (such as US, FR, or EG) while another application uses a numeral (such as 1, 2, or 3).

To associate such attributes between different applications, you create a **relationship definition**, associating the data of the source and destination attributes. Most maps use one or more relationship definitions.

Both map and relationship definitions reside in the repository of InterChange Server Express. Like business object definitions, relationship definitions function as specifications or templates for the instances that are created. Unlike instances of business objects, relationship instances persist, and are stored in special tables for each relationship.

Each time the system receives a request to transform a given business object, it executes the associated map, and, depending upon the purpose of the transformation, creates one or more instances of its associated relationship definitions. Relationship instances created during map execution contain the runtime data from the attributes they associate, and this data is stored in relationship tables.

Communication transport infrastructure

Administrative interactions between distributed InterChange Server Express components are enabled by the Common Object Request Broker Architecture (CORBA).

Content data transfers--that is, the business data that is being exchanged--can utilize either CORBA, or messaging technology provided by the Java Messaging Service (JMS) using WebSphere MQ queues.

CORBA

The Common Object Request Broker Architecture (**CORBA**) defines a set of standards and interfaces for distributed objects on a network. The Object Request Broker (**ORB**) is a set of libraries and other components that client applications and object servers use to communicate. InterChange Server Express uses the IBM Java ORB product.

The ORB makes InterChange Server Express accessible to its clients, the connector agent, and System Manager. An InterChange Server ExpressExpress instance registers with the ORB's name service, from which a client obtains the information it needs to find and start interacting with the server. The client and server perform object-to-object interactions by means of the ORB's Interface Definition Language (IDL). At a transport level, they communicate by means of the Internet Inter-ORB Protocol (IIOP). ORB-based communication is typically used for the following purposes:

- The Server Access Interface uses ORB-based communication to handle calls.
- Optionally, in request/response interactions, collaborations and connectors use ORB-based communication to exchange business objects.
- A connector agent uses ORB-based communication:

  – At startup, when it interacts with the InterChange Server Express to get its initial configuration.
  – During operation, when it receives directives from the connector controller to report its status, pause, stop, or resume.
- Optionally, ORB-based communication can also be used for event delivery in a publish-and-subscribe interaction.

In the IIOP request/response protocol, communication either succeeds or fails immediately, so both programs must be running for the components to communicate. However, for request/response interactions in an InterChange Server Express implementation, you can use a connector property to set a store-and-forward mode, specifying how a connector controller will respond to a collaboration's request in a situation where the connector agent is unavailable:
- If you set the property to True, the connector controller won't fail any collaboration requests even if the connector agent is unavailable. A request is blocked until the connector agent is operational. This causes a collaboration to wait until the connector agent is operational before it completes the processing flow for the request.
- If set to False, the connector controller fails all collaboration requests if the connector agent is unavailable. This causes a collaboration to complete the processing of the request according to its business logic for processing a failed request.

Messaging technologies

Messaging embodies a communication style in which programs asynchronously exchange discrete units of data (messages). Programs that use a messaging transport need not establish connections or wait for messages; each program asynchronously sends and receives messages by interacting with the messaging service. The messaging service provides guaranteed delivery, storing the message if the destination program is unavailable and retrying until it is available.

Java Messaging Service (JMS) is supported as the messaging system.

When JMS is the delivery transport mechanism, data persistence can be provided through the long-lived business processes feature. When this feature is used, a process initiated by a request on a collaboration can be placed in a waiting state with a timeout value, so that the process will be resumed if and when a specified data response is received. Use of this feature requires that the feature be enabled during the creation of the collaboration template.

InterChange Server Express

InterChange Server Express is a multi-threaded, Java-based execution framework for collaborations. InterChange Server Express runs within its own Java Virtual Machine (JVM). This section describes the following services and features of InterChange Server Express:
- "Event management service" on page 18
- "Connector controllers" on page 18
- "Repository" on page 18
- "Database connectivity service" on page 18
- "Database connection pools" on page 18
- "Transactional collaborations" on page 18
- "Recovery features" on page 19

Event management service

The InterChange Server Express persistently stores every business object that it receives during the execution of a collaboration. This allows the InterChange Server Express to recover from an unexpected termination or from the failure of a collaboration without losing event notifications or calls.

Connector controllers

A connector controller is an interface between the client-side of a connector and the InterChange Server Express. A connector controller routes business objects as they traverse the IBM WebSphere Business Integration Server Express system, linking the client-side of connectors to collaborations and managing the mapping process.

Through the connector controller, an administrator can:

- Trace interactions between the InterChange Server Express and the connector agent.
- Enable and disable interactions between the InterChange Server Express and the connector agent.
- Specify the type of mapping to perform for each business object arriving from or departing to the connector agent.

Repository

The InterChange Server Express maintains configuration information and definitions of all objects in a persistent store called the **InterChanger Server repository**, which consists of a set of tables in a relational database. The tables store object definitions and configuration information in the form of XML documents.

Database connectivity service

The **database connectivity service** manages interactions between the InterChange Server Express and the repository. The database connectivity service interacts with the repository by means of the Java Database Connectivity API (JDBC).

Database connection pools

You can use the System Manager tool of the IBM WebSphere Business Integration Server Express system to define database connection pools in InterChange Server Express. User-defined database connection pools make it possible for developers to directly access relational databases from within a collaboration or map. This feature provides support for

- Automated database connection life-cycle management
- Simplified APIs for SQL statements and stored procedure execution
- Container-managed transaction bracketing

Transactional collaborations

The IBM WebSphere Business Integration Server Express system supports services that can run a collaboration as if it were a type of **transaction**.

Transactional qualities are desirable for collaborations where data consistency is important across applications. Like other transactions, a transactional collaboration involves a set of steps. If an error occurs, the InterChange Server Express can undo each completed step, performing a transaction-like rollback.

However, collaborations are different from traditional transactions in some important ways:

- Collaboration actions are distributed, and there is no centralized control over the participating databases.
- Collaborations that respond to events (as in the publish-and-subscribe model) are long-lived; they execute asynchronously, because to isolate application data while they execute would adversely affect application users.
- Applications save data changes caused by collaborations, thereby providing a distributed, cross-application form of durability. However, if a collaboration needs to roll back, it might need to undo previously saved operations.

The techniques that the InterChange Server Express uses to support transactional collaborations, therefore, differ from those that support traditional transactions. The transaction levels associated with collaborations define the rigor with which the InterChange Server Express enforces transactional semantics.

Recovery features

An InterChange Server Express implementation provides features for improving the time it takes ICS to reboot after a failure, for making ICS available for other work before all flows have been recovered, and for controlling the resubmission of failed events:

- Asynchronous recovery

  InterChange Server Express does not wait for collaborations and connectors to recover before it completes boot-up; collaborations and connectors are allowed to recover asynchronously after InterChange Server Express has booted. This makes it possible to use System Manager troubleshooting tools, such as System Monitor, while the connectors and collaborations are still recovering.
- Deferred recovery

  Use of this feature is optional and is configured through the use of collaboration object properties. If you enable this feature for a collaboration, when an ICS failure occurs, the recovery of the collaboration’s WIP flows is deferred until after the server has rebooted, thereby saving the memory usage associated with those flows. After the server has rebooted, you can resubmit the events.
- Persist service call in-transit state

  You may want to prevent a recovery from automatically resubmitting all service calls that were in transit when a failure occurred, to avoid the possibility of a nontransactional collaboration sending duplicate events to a destination application. This is accomplished by configuring the collaboration (prior to the server failure) so that it will persist any service call event in the In-Transit state when a failure and recovery occurs. When InterChange Server Express recovers, the flows that were processing service calls remain in the In-Transit state, and you can examine individual unresolved flows and control when (or if) they are resubmitted.
- Guaranteed event delivery features

  For JMS-enabled connectors (connectors that use JMS as their transport mechanism), the following features can be useful for guaranteed event delivery in recovery situations:
  - Container managed events

    The container managed events feature is valid for JMS-enabled connectors that use a JMS event store. It ensures that if a system crash and recovery occurs, an event that was in process between the event store and the connector framework will be received exactly once by the connector framework, and will not be delivered twice. This feature is optional and configured through connector properties; it is used only with connectors that use JMS as their transport mechanism.

– Duplicate event elimination

The duplicate event elimination feature, valid for JMS-enabled connectors, uses unique event identifiers in the application-specific code of the connector to ensure that events will not be delivered in duplicate to the delivery queue. This feature is optional and configured through connector properties; it is used only with connectors that use JMS as their transport mechanism

Toolset Express

Tools provided with IBM WebSphere Business Integration Server Express, referred to collectively as Toolset Express, include System Manager and sets of tools for both development and administration.

System Manager is the primary mechanism for accessing and manipulating the modular components—referred to as **integration components**—that are used to implement the exchange of business data in the IBM WebSphere Business Integration Server Express environment.

System Manager executes within the context of an Eclipse workbench. The Eclipse platform is an open-source integrated development environment for the creation of tools. The Eclipse platform provides tools developers with a development kit and runtime that enables the developer to write plug-ins that allow the user to work with a particular type of resource.

System Manager is described in detail in Chapter 4, “Using System Manager,” on page 37

Development tools for specific components

Toolset Express supplies several tools for developing specific types of integration components. These tools can be launched either from within System Manager, or from the **Start>Programs>IBM WebSphere Business Integration Express>Toolset Express>Development** menu. These tools include:

- Process Designer, used for developing collaboration templates, and described in detail in the *Collaboration Development Guide*. (Process Designer is available only with the Express Plus version of the product.
- Business Object Designer, used for creating new, or modifying existing business objects, described in detail in the *Business Object Development Guide*.
- Map Designer, used creating or modifying maps, described in detail in the Map Development Guide.
- Relationship Designer, used creating or modifying relationship definitions, described in detail in the Map Development Guide.
- Connector Configurator, used for configuring connectors, described later in this guide.

Administrative and monitoring tools

Several tools are provided for monitoring and administering a system after it has been implemented. Some of these tools can also used during the implementation process. Instructions for accessing and using the administrative and monitoring tools are in the *System Administration Guide*.

Chapter 2. The development model and process

This chapter provides an overview of system implementation and contains the following sections:

- "Concepts for the business integration system"
- "Stages in developing an implementation" on page 22
- "Migration" on page 27

Concepts for the business integration system

This section defines integration concepts and structures that are necessary for understanding the development process in IBM WebSphere Business Integration Server Express implementations.

Integration components

Integration components are low-level, modular artifacts that interact with each other to integrate data. Business objects, collaboration templates and collaboration objects, maps, and connectors are all integration components. They are modular in the sense that they can be created and customized (except for connectors) individually to accomplish specific purposes, and, within licensing restrictions, are potentially reusable. But although they are modular, integration components are designed and configured to work together, as an *interface*.

Interfaces

When you develop or select integration components, you typically think about them in the context of an **interface**. An interface is not a structure or artifact itself, but is a way of thinking about the integration components that will work together to automate your specific business process. For example, you might create one interface to synchronize employee records between PeopleSoft and SAP. Another interface might synchronize customer records between Siebel and SAP.

An interface typically centers around a collaboration object, which is an instance of a collaboration template, with its ports bound to the components that are appropriate for the interface.

Solutions

A **solution** is a collection of components that are designed to address a general business need, such as order management, or customer relationship management. A solution might consist of the integration components that make up one single interface, or it might consist of the integration components that make up multiple interfaces, each interface addressing a specific aspect of the overall business need. For example, for the general purpose of customer relationship management, there might be one interface for integrating billing data about customers, utilizing a collaboration template and business objects designed for that purpose, and another interface integrating customer credit information, utilizing a collaboration template and business objects designed for that specific purpose. Together, the integration components that make up these specific interfaces comprise an overall business solution.

Integration component libraries

Integration component libraries are structural groupings of components in your development environment. You define a library in System Manager and System Manager creates a directory in the file system that represents the library. Within the library directory is a number of subdirectories for each type of integration component. When you create an integration component, System Manager creates a file or group of files and stores them in the proper subdirectory within the directory of the library.

User projects

User projects are structures within System Manager that support the need to organize integration components together so that they are viewed as belonging to an interface. User projects are collections of shortcuts to integration components in one or more libraries, so you can create one user project for each interface. Each user project might have shortcuts to some of the same components in a library because interfaces frequently share components. A customer synchronization interface and an order processing interface that both involved the SAP application, for instance, would both require the Adapter for SAP, so the user projects that correspond to those interfaces would both have shortcuts to the definition for the SAP connector.

InterChange Server Express instances

InterChange Server Express is an integration broker. It is the software infrastructure that provides business process integration and automation. When you install InterChange Server Express on a computer and start the server, you start a server **instance**. A server instance hosts the business integration system that is comprised of all the interfaces that satisfy the integration requirements of the customer. Typically you will have one instance of InterChange Server Express dedicated to developing and testing the business integration system as a whole, and one instance for hosting the production release of the system.

Stages in developing an implementation

Typically, the development of a business integration system involves the following stages:

- “Discovery and assessment of requirements”
- “Preparation of environments” on page 24
- “Design” on page 24
- “Development and configuration” on page 25
- “Validation” on page 26
- “Deployment” on page 27

Discovery and assessment of requirements

This stage begins the implementation process by identifying the business goals for the project, the system requirements, and the overall scope of the development effort.

The process of requirements elicitation is very specific to different organizations. This section presents some of the possible questions you will need to ask, but you should follow whatever method is appropriate for your organization.

Discovery starts at a high level and proceeds to lower levels of detail. It should begin with the following high-level questions:

- What is the specific business problem that must be solved?

  The answer to this question is important, because it establishes the functional requirements for interfaces during the testing stages. If testing later reveals that the business problem has not been solved then the interfaces were not designed or developed properly.
- What enterprise-level business processes need to be integrated or automated to address the business problem?

  Ask the following questions and others that might be relevant:
  - What are the names and specific versions of the applications that require integration?
  - Which are the source applications?
  - Which are the destination applications?
  - Which application is the system of record?
- What is the technology environment—including applications, databases, and APIs—in which the business processes need to be integrated?

  Determine the characteristics of the technology environment. Examine each of the following:
  - Database vendor and version
  - Platform, operating system, and version
  - APIs that exist for the applications
  - Location of all the application client and server platforms
  - Network environment
  - Anticipated transaction volume

To identify the interfaces needed for the implementation and the components that will be used, you must research information in lower levels of detail, identifying and describing the specific business processes that you intend to implement, the business logic and data transformations that are required, and details of the applications and databases that will interact. Your research may include the following information-gathering tasks:

- Identify and describe business processes that need to be integrated or automated in order to solve the business problem. Ask the following questions:
  - What is the current series of automated and manual tasks involved in the business process?
  - What event initiates the process?
  - Who are the people involved in the business process?
  - Who are the people affected by the business process (the end-users)?
  - What are the inputs and outputs?
  - Are there prerequisites or dependencies for the data?
  - Are there filtering requirements?
  - If there are multiple destination applications, what determines how the entity being processed must be routed?
  - Is the interface bi-directional?
  - How frequently does the process occur?
  - Is there a time frame in which the transaction process needs to complete? Do other processes depend upon the entities processed by it and its successful completion?

  - What is the volume of data?
  - Must the business process occur in real-time or batch?
  - Is the interface synchronous (where the initiating program requires a response) or asynchronous?
  - What is desired procedure for responding to errors?
- Describe the structure of the entities that are processed by the interface.
- Identify data transformations that must be performed between the source and destination application entity structures.
- Illustrate the business process flow with a flow diagram. The flow diagrams will help you analyze the functions that need to be performed, and later can be useful for comparisons with existing collaboration templates. This will help you determine which collaboration to use and the level of modification that may be needed. You may find it useful to use WebSphere Business Integration Workbench, Entry edition, for this purpose. This tool is available with the IBM WebSphere Business Integration Server Express Plus product package; it is not included in the Installer program for IBM WebSphere Business Integration Server Express Plus, but it is available to be installed separately.

Preparation of environments

Before you begin development activities, you should first install and prepare your environment. This includes these tasks:

1. Install IBM WebSphere Business Integration Server Express, v4.3, including InterChange Server Express, Toolset Express, and the available adapters that you intend to use, as described in the *IBM WebSphere Business Integration Server Express Installation Guide*.
2. Run the System Test sample to validate your installation, as described in the *Quick Start Guide.*
3. Set up InterChange Server Express, as described in the *IBM WebSphere Business Integration Server Express Installation Guide*.
4. Optionally, change the operation of InterChange Server Express from production mode to design mode, as described later in this guide, so that you can, if necessary, more easily add and customize integration components that are still under development.

Design

Evaluation and design are dependent upon the detailed information gathered during discovery.

When you have determined the detailed requirements of an interface and the integration components that it comprises, you are ready to evaluate available, existing integration components to see if any will meet your needs. You may find that for some requirements, components already exist and can be used as is, that for other requirements existing components need to be extended (revised according to your needs), and that for other requirements you will need to create new (custom) integration components.

To begin your evaluation, consult the documentation to learn about the characteristics of any pre-built components that may be available to you. Determine whether you can make use of pre-built technology adapters supplied with IBM WebSphere Business Integration Server Express and Express Plus. If you are using Express Plus, determine whether pre-built adapters available through the Adapter Capacity Pack, and collaborations and business objects available through the Collaboration Capacity Pack, will be useful in your interface. The capacity packs are separately available as optional additions to IBM WebSphere Business Integration Server Express Plus.

Evaluate each component both individually and in terms of how it relates to other components in the overall interface. You cannot complete the design of one component until you have also begun the design of the components with which it interacts in the interface.

For detailed information about designing components, see the following guides:
- *Business Object Development Guide*
- *Map Development Guide*
- *Collaboration Development Guide*

Development and configuration

In this stage you develop integration components according to the specifications that were derived in the "Design" on page 24 stage.

To develop the interface, you modify (if necessary) existing components that are available, such as collaboration templates and business objects, and create new components that are environment-specific (such as maps).

You define an integration component library to store your component definitions and a user project to represent your interface. You then create and store the integration components for your interface in the library and add shortcuts for them to the user project as you do so. These tasks are described in more detail in "Working with integration component libraries" on page 48 and "Working with user projects" on page 50.

The development and configuration stage is iterative and you may have to re-develop components and modify configurations as you progress.

It is recommended that you unit test components as you develop them to ensure that they satisfy their roles in the interface as you have designed it. After you have developed and unit-tested all of the components in the interface, you perform string tests to validate that the interface as a whole works as designed.

Development and configuration is typically performed in a prescribed sequence of steps, as described in Chapter 3, "Developing business process interfaces," on page 29.

While you are developing the integration components, they only exist in your local file system. When the interface is complete, you deploy it as a user project to an InterChange Server Express instance, where the component definitions are stored in the repository.

For more information on deploying user projects, see "Exporting components to a package using System Manager" on page 68.

For detailed information about developing components, see the following guides:
- Chapter 3, "Developing business process interfaces," on page 29
- The specific guide for any adapter you are using
- *Business Object Development Guide*
- *Map Development Guide*

- *Collaboration Development Guide*
- Chapter 7, “Configuring connectors,” on page 121
- Chapter 9, “Configuring collaboration objects,” on page 155

Validation

After you have deployed an interface to your local InterChange Server Express instance, you should test the interface to verify that it meets your requirements. For more information on testing, see Chapter 11, “Using Test Connector,” on page 189 and Chapter 12, “Using Integrated Test Environment,” on page 201.

In a typical implementation, two instances of InterChange Server Express are available to you. You use one instance for developing the interface, and another as the production server that will handle the live data in your completed business integration system. You can use either of these instances for testing, but the typical approach will be to test on your development instance, and then migrate the successfully tested business integration system to the production instance.

During this stage, you perform functional, performance, and regression testing as required:

- **Functional Testing**

  You perform functional testing to ensure that the interfaces in the business integration system satisfy the business process automation and integration goals of the project.
- **Performance Testing**

  You perform performance testing to ensure that the throughput, response time, and latency are within expectations.

  Depending on your specific circumstances, you may decide to migrate the business integration system into a different environment (other than your original production environment) for performance testing.

  If you find that performance is not satisfactory, then some components might need modification or re-configuration.
- **Regression Testing**

  If you modify any components as a result of the functional or performance testing (for instance, you modify a collaboration template that did not satisfy the business requirement), then you must perform regression testing to ensure that the changes made did not cause the component to violate requirements in some new way.

Overall, in completing the validation stage, you typically need to perform the following tasks:

- Create a requirements matrix that documents all functional and performance requirements.
- Prepare a system testing environment.
- Identify and develop a series of test cases that will address the functional and performance requirements.
- Depending on your specific capabilities and licensing, migrate the business integration system from the development integration environment to a test environment.
- Assign specific tests for each test matrix item.
- Execute the functional and performance tests described in the test matrices.
- Identify, document, and resolve defects or inadequacies revealed by the testing.

- Perform regression testing as required.

For more information about testing components and interfaces, see the following:
- Chapter 11, “Using Test Connector,” on page 189
- *Map Development Guide*

Deployment

The objective of the deployment stage is to migrate the business integration system into the production environment and start the production InterChange Server Express instance.

After you have validated that the business integration system satisfies the functional and performance requirements, you migrate the collection of interfaces into a production environment.

Migration

You may need to migrate your business integration system at any of several stages in development. For example, you might migrate from the original development machine to a different machine for testing and validation. Typically you migrate from one machine environment to another when you deploy to production. The decision to migrate a system will depend on your specific requirements and capabilities, and your licensing requirements and restrictions.

When you do migrate a system, the general procedure is as follows:
- Export a package that contains the components in the business integration system.
- Create an integration component library in the new environment.
- Import the package into the new library.
- Update the component definitions as necessary for environment-specific properties.
- Deploy the package of modified components into the new server repository.

For more information on migrating, and specifically for migrating to production, see the following:
- Chapter 4, “Using System Manager,” on page 37

Chapter 3. Developing business process interfaces

This chapter describes a suggested general approach for the development of the business process interfaces to be used in an IBM WebSphere Business Integration Server Express or Express plus system. The specific approach at your site may vary according to your requirements.

Although more iterations at any point in the sequence may be required in an actual implementation, these steps provide a framework for your overall development effort.

This chapter contains the following sections:

Sequence of development tasks

Many integration components in an interface depend upon and reference one another, so the tasks of developing each of the components are intertwined and iterative.

It is recommended that you develop your integration components in the following order:

1. "Obtain connectors"
2. "Develop or modify application-specific business objects" on page 30
3. "Configure connectors" on page 30
4. "Test connectivity and application-specific business objects" on page 31
5. "Develop or modify generic business objects" on page 32
6. "Configure database connection pools" on page 33
7. "Develop and test maps and relationships" on page 33
8. "Develop and test collaboration templates" on page 33
9. "Configure connectors for business object support and associated maps" on page 35
10. "Create and configure collaboration objects" on page 35
11. "Deploy the interface" on page 35

Obtain connectors

Identify and install the connectors you will use. Connectors communicate directly with the applications or technologies being integrated, and must meet their requirements. In turn, other integration components, such as application-specific business objects, depend, on the design of the connector. This makes connectors the logical starting point for your design.

Connectors are installed to your system when you install adapters. A connector is part of the adapter. (Many adapters also include an Object Discovery Agent, which assists you in generating business objects that are specific to the application with which the connector interacts.)

Adapters designed for certain technologies are provided with your installation of WebSphere Business Integration Server Express and Express Plus. If you are using

Express Plus, you may wish to consider obtaining additional, application-specific connectors that are included in an Adapter Capacity Pack. Capacity packs are separately available as optional additions to IBM WebSphere Business Integration Server Express Plus.

For more information about the connectors supplied with the product and the connectors contained in Adapter Capacity Packs, see the *IBM WebSphere Business Integration Server Express Installation Guide*.

You do not do any customization on connectors themselves, but you may need to make modifications to some of the integration components with which the connectors interact. And you will need to configure the connectors. These tasks are described in the topics that follow.

Develop or modify application-specific business objects

After you have selected and installed the files for your connector, you should develop the application-specific business objects for it. This topic provides a brief overview of that task. For details about working with application-specific business object definitions, see the *Business Object Development Guide*.

You should develop application-specific business objects after connectors, because you must understand the connector in order to develop business objects for it. You should develop application-specific business objects before you develop generic business objects, because generic business objects generally represent a superset of the application-specific business objects in an interface.

Many adapters include object discovery agents (ODAs) to assist you in generating application-specific business objects that work with the connector. Refer to the guide for the adapter to determine if the adapter features an Object Discovery Agent (ODA) that can be used to generate application-specific business objects. ODAs can greatly expedite this stage of development.

You may also need to design and create application-specific business objects individually. It is recommended that you create application-specific business objects in multiple iterations. Develop the application-specific business object at first as a relatively simple structure and then test it to make sure that the connector can use the business object structure to exchange data with the application successfully. Then add a layer of complexity and re-test the business object to make sure that the interface still works in spite of the modifications. Repeat this process until the application-specific business object is as large and complex as it has to be to satisfy the interface.

When you create the business object definition you should also create any source application triggers or other event detection mechanisms if necessary.

For details about designing and developing application-specific business object definitions, see the *Business Object Development Guide*.

Configure connectors

When you configure a connector for the purpose of unit testing an application-specific business object you may not yet have developed all of the business object definitions and maps that the connector requires to perform its role in the implementation. You can, however, add support for the business object definition you need to test and be able to test it successfully without those other components. When you have finished developing the other components you must then re-configure the connector definition to add support for the business object definitions and associate the required maps.

See the guide for your adapter for information about its application-specific properties. See Chapter 7, “Configuring connectors,” on page 121 for information about connector standard properties and information on how to use Connector Configurator.

Test connectivity and application-specific business objects

Once you have developed the application-specific business object and added support for it to a connector definition you should unit-test the business object to make sure that the connector can use it to exchange data with the application successfully. You do not need the generic objects, maps, or collaboration template that the interface will eventually use in order to perform this test. Do the following to unit-test an application-specific business object:

1. Create a pass-through collaboration template (see the *Collaboration Development Guide*) with the following design:
   - The collaboration template has two ports—one named `From` and one named `To`.
   - The `From` and `To` ports both support the application-specific business object definition.
   - The collaboration template has a single scenario named `Main`.
   - The `From` port is configured to be the triggering port for the `Main` scenario for all of the verbs belonging to the business object definition.
   - The `Main` scenario has the following elements:
     – A Start node that is connected to an Action node by a transition link.
     – A Service Call node that is attached to the Action node and that has the following characteristics:
       - The **Port** drop-down menu in the “Regular service call” pane is set to the `To` port.
       - The **Verb** drop-down menu in the “Regular service call” pane is set to the `Create` verb (or whatever the primary verb for the business object in question is).
       - The **BO Variable** field in the “Regular service call” pane is set to the value `triggeringBusObj`, which is the variable that can be used to reference the business object that triggers the collaboration.
     – An End Success node that is connected to the Action node by a transition link.

   For more information about creating collaboration templates, see the *Collaboration Development Guide*.
2. Configure the properties of the connector definition, add support for the business object definition, and enable the **Agent Support** checkbox for the business object definition.
3. Add support for the application-specific business object to another “dummy” connector definition. You do not have to configure this connector definition completely because you will only emulate it with testing tools. Ensure that the **Agent Support** checkbox is enabled for the business object definition.
4. If the connector is responsible for event notification, do the following:
   - Create a collaboration object based on the pass-through template with the connector to be tested bound to the `From` port and the “dummy” connector bound to the `To` port.

- Deploy the connectors, collaboration template, collaboration object, and business object definitions to the server.

  For more information about deploying components, see "Exporting components to a package using System Manager" on page 68.
- Start the connector agent of the connector you are testing.

  For more information about starting a connector, see your adapter guide and the *System Administration Guide*.
- Start Test Connector, open the connector definition for the other connector to which you added support for the business object definition, and connect Test Connector to the agent.

  For more information on Test Connector, see Chapter 11, "Using Test Connector," on page 189.
- Trigger an event in the source application.
- If the test is successful then the connector will send a business object representing the event to the collaboration, and it will appear as a request in Test Connector.

5. If the connector is responsible for request processing, do the following:
   - Create a collaboration object based on the pass-through template with the connector to be tested bound to the `To` port and the "dummy" connector bound to the `From` port.

     For more information on creating collaboration objects, see Chapter 9, "Configuring collaboration objects," on page 155.
   - Deploy the connectors, collaboration template, collaboration object, and business object definitions to the server.

     For more information about deploying components, see "Exporting components to a package using System Manager" on page 68.
   - Start the connector agent of the connector you are testing.

     For more information about starting a connector, see your adapter guide and the *System Administration Guide*.
   - Start Test Connector, open the connector definition for the other connector to which you added support for the business object definition, and connect Test Connector to the agent.

     For more information on Test Connector, see Chapter 11, "Using Test Connector," on page 189.
   - Send a business object request from Test Connector.
   - If the test is successful then the connector receives the business object request and processes it, creating a record for the entity in the application.

Develop or modify generic business objects

You should develop or customize the generic business object after you have developed all of the application-specific business objects required for the interface.

Determine if there are existing generic business objects available to you that might be appropriate for the business process. If so, examine them and customize them as needed, to ensure that they satisfy the interface in its current design, as reflected in characteristics of the application-specific business objects. You may need to create new generic business objects.

If you have obtained and installed one or more collaborations from the Collaboration Capacity Pack, you already have some generic business objects available to you; most of the collaborations in the Collaboration Capacity Pack come with a default set of generic business objects. You may be able to use some of these without customization; others you may need to customize. (Note: The Collaboration Capacity Pack is separately available as an optional addition to IBM WebSphere Business Integration Server Express Plus.)

If no existing generic business object can be used, you will need to either extend an existing generic business object or create a new one. For information on creating new business objects or customizing existing ones, see the *Business Object Development Guide*.

Configure database connection pools

This is an optional task. Configuring database connection pools beforehand allows you to code use of them into maps as you develop the maps. However, you may choose instead to begin the development of maps before configuring database connection pools.

To decide if and when you want to use database connection pools, see Chapter 8, "Configuring database connection pools," on page 145.

Develop and test maps and relationships

After you have identified or developed the business objects for an interface, you can develop the maps and relationships used in transforming application-specific objects into generic objects, and generic objects into application-specific objects.

You use Map Designer Express to create a map, define the transformation rules, and unit test the map with a sample input generic or application-specific business object.

As you develop maps, you may need to create relationship definitions that the maps will use for performing complex transformations. You may also find it useful to create database connection pools.

As you develop the maps, you should unit-test them by using the debugging facilities of Map Designer Express. Unit-test your maps at the following points:

- After you have defined all the simple transformation rules, such as moves, splits, and joins.
- After you have developed the more complex transformation rules that require custom Java code.
- After you have incorporated the use of relationships and database connection pools.

  When you test a map that uses relationships, you must make sure that you test the maps in the order in which they would execute within the context of the interface. If you do not, then the cross-referencing logic will not execute properly.

For more information on developing maps and relationships, see the *Map Development Guide*.

Develop and test collaboration templates

The collaboration template defines the business logic of the interface. The simplest collaborations merely route business objects between connectors. Other collaborations might also have complex interactions involving delegation of processing to other collaboration objects. Regardless, collaborations are centered around the generic business object that represents a superset of the application-specific business objects in an interface.

If you are using Express Plus, you may wish to consider obtaining collaboration templates that are included in a Collaboration Capacity Pack. Capacity packs are separately available as optional additions to IBM WebSphere Business Integration Server Express Plus. Consult the documentation, available at the InfoCenter for WebSphere Business Integration Server Express, for each specific collaboration that you might consider using. Examine the characteristics of the collaboration templates to determine whether they meet your needs.

Your specific needs may require that you customize a pre-developed collaboration template, or create additional ones, using Process Designer Express. For information about customizing or creating collaboration templates, and about the structure of collaborations generally, see the *Collaboration Development Guide*.

After you complete the collaboration template and all of the components with which it interfaces, you should create a collaboration object based on the template.

You should unit test the collaboration object without the connector agents running by using Integrated Test Environment or Test Connector. This allows you to test the integrity of the interface without introducing connectivity issues and thereby rule out errors related to the map and collaboration logic. Then you should test the interface as a whole with the connector agents running.

You can unit-test the collaboration template without using the maps, application-specific business objects, or connectors that you are developing for the interface. Do the following to unit-test a collaboration template:

1. Add support for the generic business object that is processed by the collaboration template to the `PortConnector` definition.
2. Create a collaboration object based on the collaboration template with the ports of the collaboration bound to the `PortConnector`.

   For more information on creating collaboration objects, see Chapter 9, “Configuring collaboration objects,” on page 155.
3. Deploy the `PortConnector`, collaboration template, collaboration object, and business object definitions to the server.

   For more information about deploying components, see “Exporting components to a package using System Manager” on page 68.
4. Start Test Connector, open the `PortConnector` definition, and connect Test Connector to the agent.

   For more information on Test Connector, see Chapter 11, “Using Test Connector,” on page 189.
5. Create an instance of the generic business object with data in it and send the event as a request from Test Connector.

   The collaboration receives and processes the event and each service call sends the generic business object to Test Connector, where you can edit it to examine the data for modification.
6. Examine the InterChange Server Express logging output to ensure that the collaboration logic executed as expected.

Configure connectors for business object support and associated maps

Before you can test all of the developed components together, you must reconfigure the connector definitions to add support for the application-specific and generic objects they require to participate in the interface. The system associates any maps that it can automatically, but in situations where there are multiple maps that could transform objects you must explicitly associate a map. For more information, see Chapter 7, “Configuring connectors,” on page 121.

Create and configure collaboration objects

Once you have added support for the required business object definitions to the connectors you can create a collaboration object based on the template and bind its ports to the proper components.

For more information on working with collaboration objects, see Chapter 9, “Configuring collaboration objects,” on page 155.

Deploy the interface

After you have created all of the required components for the interface you must deploy it to your local InterChange Server Express instance to test the interface. For more information on deployment, see “Exporting components to a package using System Manager” on page 68.

Test the interface

After you have deployed the interface to your local InterChange Server Express instance you should test the interface as a whole to ensure that the components satisfy the business requirements when working together. Do the following to test the interface:

1. Ensure that all of the required components are active.
2. Trigger an event in the source system.
3. Examine the logging output of InterChange Server Express to ensure that the components in the interface execute properly.
4. Examine the destination system to ensure that the entity was processed successfully and that the data appears as it should.

Chapter 4. Using System Manager

This chapter introduces you to System Manager and associated tools that you will use to develop business process integration interfaces. This chapter contains the following sections:

About System Manager, Integrated Test Environment, and Collaboration Debugger

System Manager, Integrated Test Environment, and Collaboration Debugger are plug-ins that run within the Eclipse-based tooling framework named WebSphere Studio Workbench (WSWB). This section provides an overview of the Eclipse framework, WSWB, and the plug-ins that comprise part of Toolset Express.

About the Eclipse Platform

The Eclipse Platform is an open-source integrated development environment (IDE) for the creation of tools. It provides tools developers with a development kit and runtime that enables the developer to write plug-ins that allow the user to work with a particular type of resource.

WebSphere Studio WorkBench (WSWB), which is supplied with IBM WebSphere Business Integration Server Express, is an IBM branded version of the Eclipse platform.

Plug-ins

Plug-ins are the modular extensions that software vendors develop to add functionality to an Eclipse-based workbench. Plug-ins encapsulate the perspectives, editors, and views that enable users of the workbench to work with particular types of resources. For instance, one plug-in might provide the features of a text editor. Another plug-in might provide the features of an HTML editor. The plug-in model provides a single tool with which you can work with many types of resources, rather than using dedicated tools for each type of resource.

The plug-ins available with IBM WebSphere Business Integration Server Express provide features for working with integration components. These plug-ins include System Manager, Integrated Test Environment, and Collaboration Debugger.

Workbench

The workbench is the collection of perspectives, editors, and views that are active in your Eclipse-based tooling framework, which are in turn affected by the collection of plug-ins you have installed and enabled. It is a general term used to refer to the Eclipse-based interface in which you are working, independent of the fact that the interface changes depending on how you use it.

Workspace

A workspace is a container for projects. The workspace is a directory in the file system where, by default, you are prompted to store your projects.

Projects

Projects are user-defined groups of resources, and are ultimately directories in the file system.

One of your first tasks when developing a business process interface is to define an integration component library, which is a project that contains the components you develop. When you create the integration component library you specify the location in the file system where it is stored (by default this is the workspace directory). A folder is created in that location with the name you specify for the integration component library and within the library folder a number of folders is created for each type of integration component (for instance, there are folders named `Maps`, `BusinessObjects`, and `Connectors`).

You also create projects named **user projects**. User projects are collections of shortcuts that reference integration components. You must add integration components to a user project from integration component libraries in order to deploy components to an InterChange Server Express instance. User projects are designed to allow you to functionally group components together. An integration component library is a collection of all components you might need to work with, but a user project is designed to let you group together the components that you will deploy for a specific interface.

Resources

Resources are projects, files, and folders that you work with in the workbench.

When you create an integration component, it is stored as a file in the appropriate folder within the integration component library project. The different types of integration components are stored with different extensions (for instance, maps are stored with an extension of `.cwm` whereas collaboration templates are stored with an extension of `.cwt`), but they are all stored in XML format.

Some components, such as maps and collaboration templates, also have Java source files in addition to their definition files.

Perspectives

A perspective is a grouping of editors and views designed to provide a particular user role with what it requires.

For instance, the System Manager perspective provides views for working with InterChange Server Express instances and integration component libraries and editors for collaboration object definitions and database connection pools. The Integrated Test Environment perspective has views for client emulation interfaces and business object test data, and editors for test units.

Editors

Editors allow you to open, save, and close resources in the workbench.

System Manager, for instance, has an editor in which you modify collaboration objects and another in which you configure InterChange Server Express.

Views

Views provide information about the resources with which you are working in the workbench.

System Manager, for example, has the WebSphere Business Integration System Manager view, which is the view to integration component libraries and user projects, and the InterChange Server Express view, where you work with the InterChange Server Express instances you have registered.

About WSWB

WebSphere Studio Workbench (WSWB) is an IBM-branded release of the Eclipse platform. IBM delivers WSWB with WebSphere Business Integration Server Express and it is installed when you choose to install Toolset Express. WSWB is capable of running all the plug-ins necessary to develop WebSphere Business Integration Server Express integration components.

About System Manager

System Manager is the perspective in which you work with the integration components and server instances in a WebSphere Business Integration Server Express business integration system. You use System Manager primarily for the following tasks:

- Launching other tools in the WebSphere Business Integration Server Express business integration toolset
- Developing and configuring some integration components
- Working with InterChange Server Express instances
- Deploying integration components to a repository

About Integrated Test Environment

Integrated Test Environment is the perspective in which you can test business integration interfaces you have developed. It provides graphical interfaces to emulate connectors, start the required components, and examine business object data. For more information on this perspective, see Chapter 12, “Using Integrated Test Environment,” on page 201.

About Collaboration Debugger

Collaboration Debugger is a perspective in which you can conveniently troubleshoot collaboration logic. For more information, see Chapter 13, “Using Collaboration Debugger,” on page 259.

Using System Manager

This section describes how to start and use the System Manager perspective.

Starting System Manager

To start the workbench and System Manager, do the following:

1. Select **Start > Programs > IBM WebSphere Business Integration Express > Toolset Express> Administrative > System Manager**.
2. Select **Windows > Open Perspective > Other** from the menu bar.
3. Select `System Manager` from the list of perspectives and then click **OK**.

   WebSphere Studio Workbench starts and appears. Figure 10 shows the System Manager perspective and "System Manager interface" describes the interface and its elements.

System Manager interface

The System Manager perspective has several views and editors in the default configuration it opens with. Figure 10 shows the System Manager perspective:

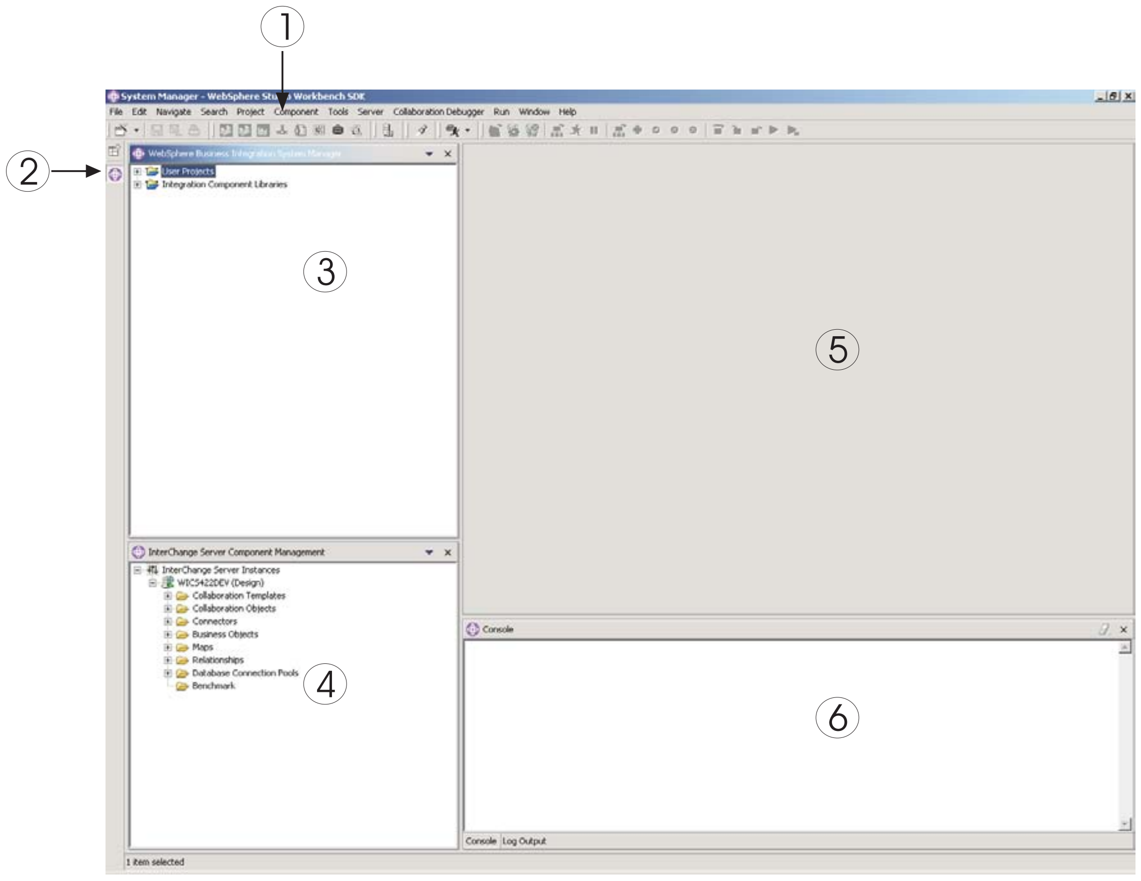

1
2
3
4
5
6
System Manager - WebSphere Studio Workbench SDK
File Edit Navigate Search Project Component Tools Server Collaboration Debugger Run Window Help
WebSphere Business Integration System Manager
User Projects
Integration Component Libraries
InterChange Server Component Management
InterChange Server Instances
WICS422DEV (Design)
Collaboration Templates
Collaboration Objects
Connectors
Business Objects
Maps
Relationships
Database Connection Pools
Benchmark
Console
Console Log Output
1 item selected

Figure 10. System Manager perspective

Table 2 on page 41 describes the interface elements of the System Manager perspective, identified by the numbers in Figure 10 on page 40:

Table 2. System Manager perspective interface elements

Interface element number	Interface element name
1	“Menu bar and toolbar”
2	“Perspective shortcut bar” on page 42
3	“WebSphere Business Integration System Manager view” on page 42
4	“InterChange Server Component Management view” on page 42
5	“Editor view” on page 43
6	“Console view” on page 43

The following sections describe the interface elements of System Manager in greater detail.

Menu bar and toolbar

You use the menu bar and toolbar to work with the Eclipse-based tooling framework and to work with WebSphere Business Integration Server Express components. Many of the menu bar items have toolbar equivalents, so the following sections only describe menu bars and their items.

File menu: This is an Eclipse-standard menu that is used to work with resources. You will use it primarily to create new integration component libraries and user projects.

For more information on creating integration component libraries and user projects, see “Working with integration component libraries” on page 48 and “Working with user projects” on page 50.

Edit menu: This is an Eclipse-standard menu that has many standard items such as Cut, Copy, and Paste.

For more information on cutting, copying, pasting, and deleting the components you create, see “Working with components in integration component libraries” on page 53.

Navigate menu: This is an Eclipse-standard menu that allows you to navigate among the resources in the workbench. For more information on this menu, see the workbench documentation.

Search menu: This is an Eclipse-standard menu that allows you to search for and search within resources. For more information on this menu, see the workbench documentation.

Project menu: This is an Eclipse-standard menu that has menu items for manipulating Project resources in the workbench; this menu item is not used when working with the System Manager perspective. For more information on this menu, see the workbench documentation.

Component menu: This menu is provided with the System Manager perspective and is useful for working with integration components you create. The items in this menu are documented throughout this guide and others in sections that describe specific tasks.

Tools menu: This menu is provided with the System Manager perspective and is used to launch tools used to create integration components.

For more information, see "Working with components in integration component libraries" on page 53.

Server menu: This menu has the **Register Servers** item, which you can use to register an InterChange Server Express instance.

For more information, see "Registering an InterChange Server Express instance" on page 43.

Run menu: This menu has an item to configure external tools to run external programs, batch files, and build scripts. For more information, see the workbench documentation.

Window menu: This menu has items for working with perspectives, views, editors, and preferences.

These items are covered in the workbench documentation and in various sections of this guide.

Help menu: This menu has items that launch the workbench documentation and that provide version information about the workbench and perspectives.

Perspective shortcut bar

Use the perspective shortcut bar to navigate conveniently between different perspectives. You can click on their workspace icons for different perspectives in the perspective shortcut toolbar to switch between them.

You can also navigate to other perspectives by using the **Window** menu:

- To navigate to a perspective represented by an icon that is higher up in the perspective shortcut bar than the icon for the perspective you are currently viewing use the keyboard shortcut **Alt+Up Arrow**.
- To navigate to a perspective represented by an icon that is lower down in the perspective shortcut bar than the icon for the perspective you are currently viewing, select **Perspective > Next** from the menu bar, or use the keyboard shortcut **Alt+Down Arrow**.

WebSphere Business Integration System Manager view

This view has the **User Projects** and **Integration Component Libraries** nodes, which are types of InterChange Server Express projects.

For more information on working with these types of projects, see "Working with integration component libraries" on page 48 and "Working with user projects" on page 50.

InterChange Server Component Management view

Use this view to work with InterChange Server Express instances and the components in them. When you register an InterChange Server Express instance, a row and entry are created underneath the **Server Instances** node for each server. You can expand the node to view the components in the repository of that instance, and can use right-click menu items to manipulate those components.

For more information about registering and working with InterChange Server Express instances, see "Working with InterChange Server Express instances."

Editor view

This is the view in which you work with different resources in the framework, such as files and integration component definitions. Different editors open to work with different types of resources. For instance, text files are opened in a text editor whereas collaboration object definitions are opened in a InterChange Server Express-specific editor.

Console view

This view has two tabs: **Console** and **Log Output**. When you compile maps or collaboration templates in System Manager, the **Console** tab displays messages to indicate whether compilation for each component completed successfully and the **Log Output** tab displays any errors or warnings encountered.

Working with InterChange Server Express instances

This section describes the tasks you perform in System Manager to work with InterChange Server Express.

Registering an InterChange Server Express instance

To work with an InterChange Server Express instance you must register it in System Manager. Do the following:

Important: To connect to an InterChange Server Express instance, it must be running. For information on starting InterChange Server Express see the *System Administration Guide*. You can still register a server that is not running, but System Manager will not be connected to it after doing so.

1. Do one of the following to display the "Register Server" dialog:
   - Select **Server** > **Register Servers** from the menu bar.
   - Right-click the **InterChange Server Instances** node in the InterChange Server Express Component Management view and choose **Register Server** from the context menu.

     Note: If you do not see the InterChange Server Express view, enable it by following the instructions in "Showing and closing views" on page 96.
2. Do one of the following to enter the name of the server in the **Server name** field:
   - Type the name of the InterChange Server Express instance in the **Server name** field.

     Important: The name of an InterChange Server Express instance is case-sensitive, so you must be accurate when specifying the name.
   - Do the following to browse for the InterChange Server Express instance on the network
     a. Click **Browse**.

        System Manager discovers active servers on the network and lists them at the "Find Server" dialog. Depending on the size, speed, and configuration of the network this may take some time.

Note: Although you can register a server that is not running by typing the name of the server, you will not be able to browse for the server if it is not running.

b. Select the server instance you want to register and click **OK**.

3. Type the user name to interact with the InterChange Server Express instance in the **User name** field.

   The default user name is `admin`.
4. Type the password for the user name specified in step 3 in the **Password** field.

   The default password for the default user name `admin` is `null`.
5. If you do not want to have to supply the user name and password each time you have to connect to the InterChange Server Express instance in System Manager then enable the **Save User ID and Password** checkbox.

   Important: Be sure to consider the security implications of caching the user name and password in this way. Some component definitions require sensitive information such as valid user names and passwords to log in to the applications being integrated. With such information, an individual could gain access to records stored in those applications and the critical information stored in those records, such as credit and payroll information. It is recommended that you only cache the user name and password in this way when the only information that could be accessed through System Manager is not production information.
6. To register the InterChange Server Express instance as a local test server, enable the **Test Server** checkbox and then either type the full path to the InterChange Server Express product directory in the **Test Server Installation Path** field, or use the **Browse** button to navigate to the product directory.

   You should only enable this checkbox when you need to test an interface, and should leave it disabled when developing components or when working with a production server.

   For more information on testing interfaces, see Chapter 12, “Using Integrated Test Environment,” on page 201 and Chapter 13, “Using Collaboration Debugger,” on page 259.
7. Click **OK**.

   System Manager registers the InterChange Server Express instance, connects to it (if the name of the InterChange Server Express instance, the user name, and the password supplied are all accurate and the server and the IBM Java Object Request Broker (ORB) are running), and displays an entry for it in the InterChange Server Express Component Management view.

   You can subsequently work with the InterChange Server Express instance by right-clicking it in the InterChange Server Express Component Management view.

Connecting to InterChange Server Express

When you register an InterChange Server Express instance in System Manager, System Manager automatically connects to the instance if the instance name, the user name, and the password are all accurate and the server and the IBM ORB are running.

If you have to shut down the instance, or exit from System Manager, then you need to reconnect System Manager to the instance. The task is slightly different depending on whether or not you chose to cache the user name and password, so follow the steps in the appropriate following section.

Connecting with a cached user name and password

Do the following if you chose to cache the user name and password when registering the server initially:

1. Right-click the entry for the InterChange Server Express instance in the InterChange Server Express Component Management view and choose **Connect** from the context menu.

   The “Login” dialog is displayed with the login information cached.
2. Click **OK**.

Note: If System Manager was unable to connect to the server, see “Troubleshooting problems connecting to InterChange Server Express in System Manager” on page 101.

Connecting when the user name and password have not been cached

If you did not choose to cache the user name and password when initially registering an InterChange Server Express instance, then you must do the following:

1. Right-click the entry for the InterChange Server Express instance in the InterChange Server Express Component Management view and choose **Connect** from the context menu.
2. Type the user name to interact with the InterChange Server Express instance in the **User name** field.

   The default user name is `admin`.
3. Type the password for the user name supplied in step 2 in the **Password** field.

   The default password for the default user name `admin` is `null`.
4. If you do not want to have to supply the user name and password each time you have to connect to the InterChange Server Express instance in System Manager then enable the **Save userid/password** checkbox.

   Important: Be sure to consider the security implications of caching the user name and password in this way. Some component definitions require sensitive information such as valid user names and passwords to log in to the applications being integrated. With such information, an individual could gain access to records stored in those applications and the critical information stored in those records, such as credit and payroll information. It is recommended that you only cache the user name and password in this way when the only information that could be accessed through System Manager is not production information.
5. Click **OK**.

InterChange Server modes

InterChange Server can run in different modes that best suit different stages of the implementation cycle.

Design mode

In design mode, InterChange Server Express permits the repository to be in an inconsistent state—you can add components to the repository without components they depend upon already existing. For instance, if you try to import a business object definition that has a child object into the repository, but the child object does not exist yet, then an InterChange Server Express in production mode would cause the import to fail to protect the integrity of the repository. An InterChange Server Express in design mode, however, would allow you to proceed so that you can assemble your integration components in a way that best suits your development approach.

Furthermore, compiling maps and collaboration templates when deploying a package to a design-mode server is optional. In production mode, the server automatically compiles all maps and collaboration templates.

Design mode is particularly useful when you are importing components from another environment. You may not be aware of all the dependencies yourself, so being able to incrementally import components without the import operations failing due to unresolved dependencies is very helpful.

To start InterChange Server Express in design mode, use the `-design` parameter when starting the server. To do so, edit the Windows shortcut for InterChange Server Express, adding the parameter`-design` at the end of the path shown in the **Target** field. Later, when you wish to use Production mode, edit the shortcut again, and remove the `-design`parameter from the path in the **Target** field.

Production mode

In production mode, InterChange Server Express is designed to guarantee the integrity of the repository. It will not allow you to deploy a package with unresolved dependencies to the repository, and it automatically compiles all maps and collaboration templates in the deployment package. These restrictions guarantee that the server environment is in a state in which its components can execute properly. If there were components with unresolved dependencies or uncompiled components in the server environment at runtime then any transactions that involved those components would fail. Although that is an acceptable situation in a development environment, where it is presumed that you are still creating the required components, it is not considered acceptable in a production environment, so these restrictions enforce safe deployment procedures.

Production mode is the default mode for InterChange Server Express, so you do not have to take any configuration steps to start it in production mode. If you want to start it in production mode, however, be sure that you have not taken steps to start it in design mode and confirm its mode in the InterChange Server Express Component Management view of System Manager.

Changing the InterChange Server Express password

You can change the password for the user account that is used to connect to InterChange Server Express. Do the following to change the password:

1. Right-click the entry for the InterChange Server Express instance in the InterChange Server Express view and choose **Change Password** from the context menu.

   The “Change InterChange Server Express Password” dialog displays.
2. Type the current password in the **Old Password** field.
3. Type the new password in the **New Password** field.
4. Type the new password again in the **Confirm Password** field.
5. Click **OK**.

Refreshing InterChange Server Express

After you have deployed components to an InterChange Server Express instance you must refresh the instance in System Manager for it to accurately display the components in the server. For instance, if you deploy components to a server and then try to create a new integration library and add components to the library from the server, System Manager does not list the recently deployed components unless you refresh the server.

To refresh a server instance, right-click it in the InterChange Server Express Component Management view and choose **Refresh** from the context menu.

Disconnecting from InterChange Server Express

To disconnect System Manager from an InterChange Server Express instance, right-click the InterChange Server Express instance from which you want to disconnect in the InterChange Server Express Component Management view, then select **Disconnect** from the context menu.

Shutting down InterChange Server Express

To shut down an InterChange Server Express instance, right-click the InterChange Server Express instance you want to shut down in the InterChange Server Express Component Management view, then select **Disconnect** from the context menu and then either select **Gracefully** or **Immediately** from the submenu depending on how you want the instance to shut down.

If you select **Immediately** as the type of shutdown, then InterChange Server Express shuts down immediately and any flows that it might be processing at the time fail. You can subsequently resolve any failed flows by using Flow Manager. Use this type of shutdown in development and test environments where you do not care about the flows in the system or in production environments where there are no complications presented by submitting failed flows.

If you select **Gracefully** as the type of shut down, then the InterChange Server Express integration components will finish processing their current flows before the server shuts down. Use this type of shutdown in production environments where there may be complications resulting from flow failures.

Note: You can also shut down InterChange Server Express by closing the console window in which it is running. This results in an immediate shutdown, and any flows currently being processed will fail. This may be acceptable in a development environment, if the integrity of the data does not matter, but should not be done in a production environment. A benefit of using System Manager to shut down InterChange Server Express even in a development environment is that the InterChange Server Express Component Management view displays the status of the server, but it cannot detect and report changes in the state of the server if it is shut down by some other means than the use of System Manager.

Deleting an InterChange Server Express instance from the view

You may want to remove an InterChange Server Express instance from the InterChange Server Express Component Management view in System Manager. Do the following to remove an InterChange Server Express instance:

1. Ensure that System Manager is not connected to the InterChange Server Express instance you want to remove. You can do this by either disconnecting System Manager from the InterChange Server Express instance or by shutting down the InterChange Server Express instance.

   For more information on disconnecting System Manager from an InterChange Server Express instance, see "Disconnecting from InterChange Server Express" on page 47.

   For more information on shutting down an InterChange Server Express instance, see "Shutting down InterChange Server Express" on page 47.
2. In the InterChange Server Express Component Management view, right-click the InterChange Server Express instance you want to remove, then select **Delete** from the context menu.
3. At the "Delete server confirmation" prompt click **OK**.

   The server instance is removed from the view.

Other System Manager commands for InterChange Server Express

The context menu displayed when you right-click an InterChange Server Express instance has several additional menu items.

Table 3. Other System Manager commands for InterChange Server Express

InterChange Server menu item	
Edit Configuration	For more information on this menu item, see Chapter 5, "Configuring InterChange Server Express," on page 103.
Statistics	For more information on this menu item, see the *System Administration Guide*.
System View	For more information on this menu item, see the *System Administration Guide*.
Server object delete	For more information on this menu item, see "Deleting components using the Server Object Delete Wizard" on page 91.
Delete Repository	"Deleting the entire repository" on page 90.
Monitor definition wizard	For more information on this menu item, see the *System Administration Guide*.

Working with integration component libraries

You use integration component libraries to store the components that you develop. This section describes how to create a new integration component library.

Once you have created an integration component library you will typically want to perform the following tasks as well:

- You will want to import components into the library. For information on several ways of doing this, see "Working with components in integration component libraries" on page 53.
- You will want to create shortcuts to the components in user projects. For more information on this, see "Working with user projects" on page 50.
- You will want to deploy components to an InterChange Server Express instance. For more information, see "Deploying components to a server" on page 70.

- You will want to export components to a package, either to import them into servers or other libraries or to back up your development. For more information, see "Exporting components to a package using System Manager" on page 68.

For conceptual information about integration component libraries, see "Integration component libraries" on page 22.

Creating integration component libraries

Do the following to create a new integration component library in System Manager by using a wizard:

1. Do one of the following to start the "New Integration Component Library" wizard:
   - Select **File > New > Integration Component Library** from the menu bar.
   - In the WebSphere Business Integration System Manager view, right-click the **Integration Component Libraries** folder and select **New Integration Component Library** from the context menu.
   - Click **Open The New Wizard button** in the toolbar and select **New Integration Component Library** from the menu.
2. Type a name for the integration component library in the **Project name** field.

   Project names can only contain alphanumeric characters and underscores, and must be specified in English.
3. To have the folder for the library created in the default location (your workspace) and with a name identical to the name specified for the library, leave the **Use default location** checkbox enabled.

   If you want to specify the name and location of the library folder, do the following:
   1. Clear the **Use default location** checkbox.
   2. Type the full path and name of the directory that you want to use for the library in the **Location** field, or click **Browse** to select an existing directory.

   Note: There is no way to create the folder for a library in the path of the workspace other than to let System Manager do it by use of the **Use default location** checkbox.
4. If you do not want to import components into the library from the repository of a server at this time, proceed to step 5 on page 50.

   If you do want to import components into the library from the repository of a running server, do the following:
   1. Do one of the following to specify the server from which you want to import the components:
      - Select the server in the **Import components from server** drop-down menu.

        The server must be running and System Manager must be connected to it for the server to be displayed in the drop-down menu.
      - Click **Register new server** and add the server to the list through the wizard. For more information, see "Registering an InterChange Server Express instance" on page 43.
   2. Click **Next**.
   3. At the next screen in the wizard, enable the checkbox next to a server instance to add all of the components in its repository or expand the server folder and enable the checkboxes next to component groups, or expand the folders for groups and enable the checkboxes for individual components.

   d. Enable the **Deep** checkbox to add all of the dependencies of the selected components. For more information on dependencies, see “Dependencies and references” on page 92.
5. Click **Finish** to complete the wizard.

   System Manager creates a folder with the name you specified under the **Integration Component Libraries** folder.

Working with user projects

You create shortcuts in user projects to the integration components you want to work with in one or more libraries. User projects provide a way for you to organize your view of components as an interface. You must add component shortcuts to a user project to deploy the components to a server from System Manager.

This section contains the following sections:
- “Creating user projects”
- “Adding shortcuts to a user project” on page 51
- “Exporting a solution” on page 67

For conceptual information about user projects, see “User projects” on page 22.

Creating user projects

Do the following to create a new user project in System Manager by using a wizard:
1. Do one of the following to start the “New User Project” wizard:
   - Select **File > New > User Project** from the menu bar.
   - In the WebSphere Business Integration System Manager view, right-click the **User Projects** folder, then select **New User Project**, then select **New ICS project** from the context menu.
   - In the WebSphere Business Integration System Manager view, expand the **User Projects** folder, then right-click the **InterChange Server Projects** folder and select **New ICS project** from the context menu.
   - Click **Open The New Wizard button** in the toolbar and select **New User Project** from the menu.
2. Type a name for the user project in the **Project name** field.

   Project names can only contain alphanumeric characters and underscores, and must be specified in English.
3. To have the folder for the user project created in the default location (your workspace) and with a name identical to the name specified for the user project, leave the **Use default** checkbox enabled in the “Project contents” pane.

   If you want to specify the name and location of the user project folder, do the following:
   a. Clear the **Use default** checkbox in the “Project contents” pane.
   b. Type the full path and name of the directory that you want to use for the user project in the **Directory** field, or click **Browse** to select an existing directory.

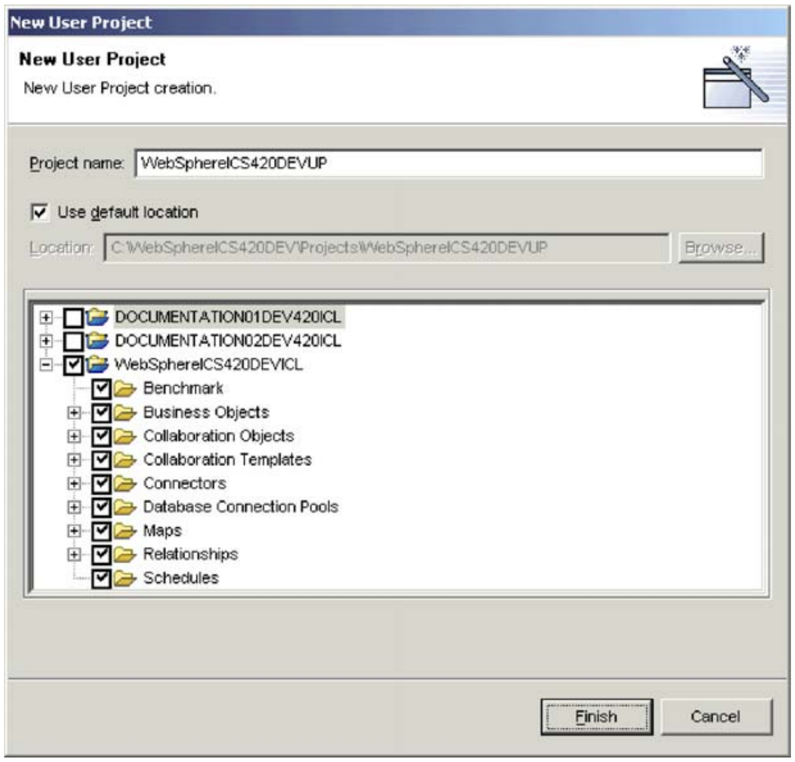
   Note: There is no way to create the folder for a user project in the path of the workspace other than to let System Manager do it by use of the **Use default location** checkbox.

4. If you do not want to create shortcuts to existing integration components at this time, proceed to step 5.

   If you do want to create shortcuts to existing integration components, enable the checkbox next to an integration component library to create shortcuts to all of the components within it or expand an integration component library folder and enable the checkboxes next to component groups, or expand the folders for groups and enable the checkboxes for individual components.

   Note: If you select components with the same names from multiple integration component libraries you do not receive a prompt to inform you that there are duplicate references in your selection. In the event that you do select duplicate components, shortcuts are created for the component in the integration component library that was furthest down in the list of libraries when you made your selections in the wizard.

   Figure 11 shows the “New User Project” wizard.



New User Project

New User Project

New User Project creation.

Project name: WebSphereICS420DEVUP

☑ Use default location

Location: C:\WebSphereICS420DEV\Projects\WebSphereICS420DEVUP

☐ DOCUMENTATION01DEV420ICL
☐ DOCUMENTATION02DEV420ICL
☑ WebSphereICS420DEVICL
  ☑ Benchmark
  ☑ Business Objects
  ☑ Collaboration Objects
  ☑ Collaboration Templates
  ☑ Connectors
  ☑ Database Connection Pools
  ☑ Maps
  ☑ Relationships
  ☑ Schedules

Figure 11. Creating a user project

5. Click **Finish** to complete the wizard.

   System Manager creates a folder with the name you specified in the **InterChange Server Projects** folder in **User Projects** folder.

Adding shortcuts to a user project

You add shortcuts to a user project to give yourself a view to an interface that you are working on. There are several ways to add shortcuts to a user project, described in the following sections:

- “Using the Dependency Tree” on page 52
- “Using the Update Project wizard” on page 53
- “Dragging-and-dropping components” on page 53

- “Importing a solution” on page 67

Using the Dependency Tree

The “Dependency Tree” wizard is the most convenient interface for adding component shortcuts to a user project. User projects are primarily designed to represent interfaces, and interfaces are generally centered around a collaboration object. User projects, then, are generally center around a collaboration object as well, and you can typically create the shortcuts you need for a user project by discovering the dependencies of a collaboration object.

For conceptual information about dependencies, see “Dependencies and references” on page 92.

Do the following to add shortcuts to a user project by using the “Dependency Tree” wizard:

1. Right-click an integration component, such as a collaboration object, in a library and choose **Show Dependencies** from the context menu.
2. Select the user project to which you would like shortcuts to be added from the **Add to the project** drop-down menu.
3. Select components for which you would like to create shortcuts in the left-hand pane of the wizard.

   You can use keyboard shortcuts to facilitate the process, such as holding down **Shift** to select ranges of objects and holding down **Ctrl** to select single non-contiguous objects.
4. Click the right-facing arrow to add the components to the right-hand pane of the wizard.

   Figure 12 shows the “Dependency Tree” wizard:

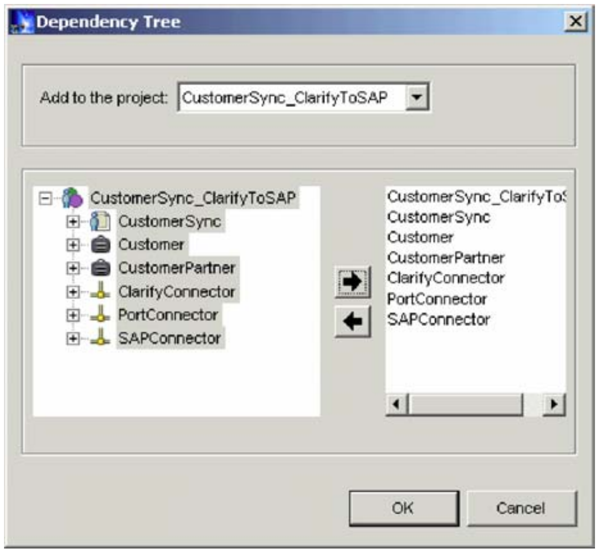


Figure 12. Adding shortcuts to a user project by using the Dependency Tree

5. Click **OK**.

   System Manager creates shortcuts to the selected components in the specified user project.

Using the Update Project wizard

The “Update Project” wizard provides an interface to add shortcuts to a user projects that is similar to the one displayed when you initially create a user project. Do the following to use the “Update Project” wizard:

1. Right-click any user project in the WebSphere Business Integration System Manager view and choose **Update project** from the context menu.
2. Enable the checkbox next to an integration component library to create shortcuts to all of the components within it or expand an integration component library folder and enable the checkboxes next to component groups, or expand the folders for groups and enable the checkboxes for individual components.
3. Click **Finish**.

   If there are shortcuts in the user project to components with the same name as those you selected in the wizard, then the wizard displays a prompt that allows you to do the following:

   - Overwrite the displayed component.
   - Overwrite all duplicate components.
   - Not overwrite the component.
   - Cancel the update operation.

   If you did not select any components with the same name as components for which shortcuts already exist in the user project, then shortcuts are added to the project and the wizard exits.

Dragging-and-dropping components

You can select components from integration component library folders and drag-and-drop them into a user project to add shortcuts to those components to the user project.

If there are no shortcuts in the folder of the user project currently then you must drag-and-drop the components onto the folder itself. Drag-and-drop the components onto the folder and release the mouse button when a square appears beneath the mouse pointer.

If there are already shortcuts in the folder of the user project then you cannot drag-and-drop components onto the folder. Drag-and-drop the components between existing shortcuts in the folder until a line appears and then release the mouse button.

Working with components in integration component libraries

The majority of your time is spent working with integration components when you implement a WebSphere Business Integration Server Express integration system. Although this guide does not go into detail about how to develop individual components, this section does cover how to launch the designer tools, how to start creating new components, how to start modifying existing components, and how to work with the few components that are developed in System Manager.

For more information on how to develop integration components, see the following guides:

- *Collaboration Development Guide*
- *Map Development Guide*
- *Business Object Development Guide*
- *Access Development Guide*

For information on working with components that you have developed and deployed to an InterChange Server Express instance, see "Working with components in an InterChange Server Express repository" on page 83.

Naming component guidelines

When you create components, keep in mind the following general guidelines and requirements

- In general, limit component names to combinations of alphanumeric characters and underscores because most punctuation and special characters are not used in an ICS system. Do not use spaces in component names.
- Begin all component names with a letter (not a digit or an underscore). See specific sections for exceptions.
- In property names, do not use apostrophes (′) or double quotation marks (″).
- Many component names have a maximum length limit:
  - Map names: Maximum 76 characters
  - Collaboration templates: Maximum 80 characters
  - Collaboration object: Maximum 80 characters
- The following components must be composed only of characters in the code set associated with the U.S. English locale, en_US: connectors and connector configuration properties, collaboration templates, collaboration objects, collaboration properties, collaboration ports, maps, business object definitions, attributes, attribute types, and verbs.

In the interest of backward compatibility, IBM enforces only a few naming requirements. However, because naming is subject to restrictions from underlying system components, follow the conventions in this guide. For example, Java imposes restrictions on Java class names, and your operating system and database likely have restrictions on the length of and characters used in file names.

Launching designer tools

This section describes the different ways you can launch each of the designer tools. You can use the designer tools to create new components or to open and modify existing components.

Note: If you attempt to launch one of the designer tools and experience an error about a class not being found, you must launch System Manager and then try to launch the designer tool again. System Manager does not have to remain running after the tool is initially launched, however.

Business Object Designer Express

To launch Business Object Designer Express, do one of the following:

- Right-click the **Business Objects** folder in the WebSphere Business Integration System view and choose **Create New Business Object** from the context menu
- Select any folder in the WebSphere Business Integration System Manager view and do one of the following:
  - Select **Tools > Business Object Designer Express** from the menu bar
  - Click the **Business Object Designer Express** toolbar button
  - Use the keyboard shortcut **Ctrl+4**
- Select **Start > Programs > WebSphere Business Integration Express > Toolset Express> Development > Business Object Designer Express**

For more information on Business Object Designer Express, see the *Business Object Development Guide*.

Connector Configurator

To launch Connector Configurator, do one of the following:

- Right-click the **Connectors** folder in the WebSphere Business Integration System view and choose **Create New Connector** from the context menu
- Select any folder in the WebSphere Business Integration System Manager view and do one of the following:
  - Select **Tools > Connector Configurator** from the menu bar
  - Click the **Connector Configurator** toolbar button
  - Use the keyboard shortcut **Ctrl+1**
- Select **Start > Programs > IBM WebSphere Business Integration Express >Toolset Express> Development > Connector Configurator**

For more information on Connector Configurator, see Chapter 7, "Configuring connectors," on page 121.

Map Designer Express

To launch Map Designer Express, do one of the following:

- Right-click the **Maps** folder in the WebSphere Business Integration System view and choose **Create New Map** from the context menu
- Select the folder for any type of integration component in the WebSphere Business Integration System Manager view and do one of the following:
  - Select **Tools > Map Designer Express** from the menu bar
  - Click the **Map Designer Express** toolbar button
  - Use the keyboard shortcut **Ctrl+3**
- Select **Start > Programs > IBM WebSphere Business Integration Express >Toolset Express> Development > Map Designer Express**

For more information on Map Designer Express, see the *Map Development Guide*.

Relationship Designer Express

To launch Relationship Designer Express, do one of the following:

- Right-click the **Relationships** folder in the WebSphere Business Integration System Manager view and choose **Relationship Designer Express** from the context menu
- Select the folder for any type of integration component in the WebSphere Business Integration System view and do one of the following:
  - Select **Tools > Relationship Designer Express** from the menu bar
  - Click the **Relationship Designer Express** toolbar button
  - Use the keyboard shortcut **Ctrl+5**
- Select **Start > Programs > IBM WebSphere Business Integration Express > Toolset Express> Development > Relationship Designer Express**

For more information on Relationship Designer Express, see the *Map Development Guide*.

Process Designer Express

To launch Process Designer Express, do one of the following:

- Right-click the **Collaboration Templates** folder in the WebSphere Business Integration System view and choose **Create New Collaboration Template** from the context menu
- Select the folder for any type of integration component in the WebSphere Business Integration System view and do one of the following:
  - Select **Tools > Process Designer Express** from the menu bar
  - Click the **Process Designer Express** toolbar button
  - Use the keyboard shortcut **Ctrl+2**
- Select **Start > Programs > IBM WebSphere Business Integration Express>Toolset Express> Development >Process Designer Express**

For more information on Process Designer Express, see the *Collaboration Development Guide*.

Creating new components

For the following components, launching their respective designer tools as described in "Launching designer tools" on page 54 allows you to create a new component of that type:
- Business objects
- Maps
- Relationships
- Collaboration templates

Some components do not have dedicated designer tools, so you create them in interfaces displayed by System Manager. To create new collaboration objects, database connection pools, and schedules see Table 4:

Table 4. Techniques for creating new integration components without designer tools

Component	**Technique**	**For more information, see...**
Collaboration object	Right-click the **Collaboration Objects** folder in the WebSphere Business Integration System view and select **Create New Collaboration Object** from the context menu	Chapter 9, "Configuring collaboration objects," on page 155
Database connection pool	Right-click the **Database Connection Pools** folder in the WebSphere Business Integration System view and select **Create New Database Connection** from the context menu	Chapter 8, "Configuring database connection pools," on page 145
Schedule	• Right-click the **Schedules** folder in the WebSphere Business Integration System view and select **Edit Components' schedule** from the context menu • Right-click any component that can be scheduled in a library or the shortcut to any component that can be scheduled in a user project and select **Edit Components' schedule** from the context menu	*System Administration Guide*

Table 4. Techniques for creating new integration components without designer tools (continued)

Component	Technique	For more information, see...
Web service	To designate a Web service as an integration component, right-click the**WebServices** folder in the WebSphere Business Integration System view and select **Register New Web Service** from the context menu.	"Designating web services as integration components"

Designating web services as integration components

This section describes how to register and invoke a web service as an integration component in System Manager and contains the following sections:

- "Overview"
- "Registering a web service"

Overview

Web services are self-contained, modular, distributed, dynamic applications that can be described, published, located, and invoked over the network to create products, processes, and supply chains. They can be local, distributed, or Web-based. Web services are built on top of open standards such as TCP/IP, HTTP, Java, HTML, and XML. Web services use new standard technologies such as SOAP (Simple Object Access Protocol) for messaging, and UDDI (Universal Description, Discovery and Integration) and WSDL (Web Services Description Language) for publishing and discovery.

Using System Manager, you can register a web service as a component of an integration component library (ICL). The registration process lists the web service as a component under Integration Component Libraries and automatically generates the business objects that InterChange Server Express will use to communicate with the web service. You can also use System Manager to test the registered web service before deploying it to the server.

Registering a web service

In System Manager, use the *Register a new web service* wizard to register a web service.

1. To start the wizard, expand**Integration Component Libraries** in the WebSphere Business Integration System view, right-click the **WebServices** folder, and select **Register New Web Service**. The *Choose a method to register web services* page appears.
2. Click one of the following options:

Option	Description
Enter a URL for WSDL file	Allows you to specify the exact web service you want to register
Search UDDI registry for a web service	Allows you to specify a UDDI registry that you can further search to locate the web service.

3. Click Next.
4. If you clicked **Enter a URL for WSDL file** in Step 2, the *Please enter WSDL URL* page appears. In the **WSDL URL** field, specify the URL of the service you wish to register, click Next, and proceed to Step 8 on page 58.

5. If you clicked **Search UDDI registry for a web service** in Step 2, the Searching for web services page appears. Do the following:
   a. In the **Search terms** field, type the search criteria for the web service you wish to find.
   b. Select one or more of the following supported UDDI registries:
      - IBM UDDI Registry
      - Microsoft UDDI Registry
      - XMethods UDDI Registry
   c. Click **Search**. The Search Results tab displays the names of those registries in which results have been found. If the registry you selected is not available, an error message appears on the page.
6. For each registry name on the list, do the following:
   - Select the registry name to view the number of search results for that registry.
   - Expand the registry list to view the names of the services found in that registry.
   - Select the name of a service to view the web service details.
7. Select the name of the web service you want to register and click next to proceed to the next page of the wizard. The *New Web service is found* page appears.
8. On the *New Web service is found* page, you can do the following:
   - Click the **Available Methods** tab to view the list of callable services for the web service. The services are essentially methods that can be called by a client.
   - Click the **WSDL** tab to view the XML code that describes the web service.
9. When you click on an **available method**, more information (if available) will be displayed on the right hand pane. Note that the `Fault(s)` parameter is used to store errors or exceptions that may occur when the service is called.
10. Click **Finish** to register the web service.

Viewing a registered web service

After you have registered a web service you can view it in System Manager.

1. Under **Integration Component Libraries > WebServices**, double-click the name of the web service: The **Overview** tab appears.
2. Click the **Overview** tab to display the Available Operations page. This page provides the list of services, or methods, for that web service. The Web Service Methods panel, which appears to the right of the Available Operations page, also displays the list of methods.

   All the methods of the web service, whether you intend to use them or not, are registered and appear on the Available Operations page, in the **Operation** column.

   The business objects that were created for each method when you registered the web service (see "Registering a web service" on page 57) are listed under the **Input BO** and **Output BO** columns.
3. To refresh the list, click the **Create Business Objects Definitions** button.
4. In the Web Service Methods panel, select a method to view its parameters in the Method Parameters panel. Parameters are listed under the **Input**, **Output**, and **Fault** categories.

Testing a registered web service

You can test a registered web service before deploying it to the integration server.

1. Under **Integration Component Libraries > WebServices**, double-click the name of the web service. The **Overview** tab appears, along with the Web Service Methods panel and the Method Parameters panel.
2. In the Web Service Methods panel, select a method.
3. In the Overview page, either double-click on the method name, or right-click it and select **Test** to test the method. A business object editing page appears with the name of the input and output business objects for the method. This page is used to test the input business object.
4. To test the business object, do the following:
   - Click the **Create New Request**. The request business object will be created in the left pane. Supply the input parameters. You can ignore the following attributes: SOAPConfigMO, ProtocolConfigMO, OBjectEventID.
   - Click the **Invoke Web Services** button to call the web service. The response from the web service displays in the **Response business object** panel.
5. Click the panel of each web service method with which you want to test the business objects created when you registered the service.

Modifying existing components

To modify business objects, maps, relationships, and collaboration templates you can do the following:

- Double-click the component in a library or the shortcut to a component in a user project
- Select the component in a library or the shortcut to a component in a user project and do one of the following:
  - Launch its designer tool as described in "Launching designer tools" on page 54
  - Press **Enter**
  - Press **Ctrl+E**
  - Select **Component > Edit Definitions** from the menu bar
- Right-click the component in a library or the shortcut to a component in a user project and choose **Edit definition** from the context menu
- Launch the component's designer tool as described in "Launching designer tools" on page 54 and then open the component after the tool has started

See Table 5 for information on how to modify collaboration objects, database connection pools, and schedules:

Table 5. Techniques for modifying integration components without designer tools

Component	**Technique**	**For more information, see...**
Collaboration object	Double-click the collaboration object in the **Collaboration Objects** folder in the WebSphere Business Integration System view	Chapter 9, "Configuring collaboration objects," on page 155
Database connection pool	You cannot modify a database connection pool. You may modify some properties of the pool component, but cannot change definition elements such as the database to which the pool connects.	Chapter 8, "Configuring database connection pools," on page 145

Table 5. Techniques for modifying integration components without designer tools (continued)

Component	Technique	For more information, see...
Schedule	• Right-click the **Schedules** folder in the WebSphere Business Integration System view and select **Edit Components' schedule** from the context menu • Right-click any component that can be scheduled in a library or the shortcut to any component that can be scheduled in a user project and select **Edit Components' schedule** from the context menu The "Schedule" interface appears and allows you to modify schedules that have been defined.	*System Administration Guide*

Importing components into a library from a server using the import wizard

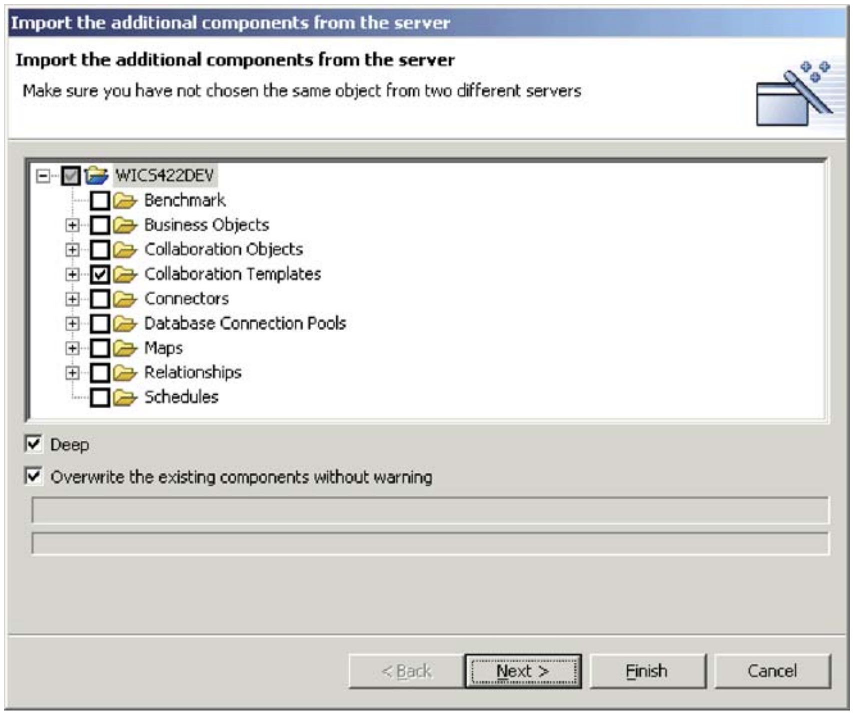
You can import integration components into a library from an InterChange Server Express repository.

Do the following to import components into an integration component library from an InterChange Server Express repository:

Note: To import components into a library from an InterChange Server Express instance, the server must be running. If System Manager is not connected to the server when you start the "Import components" wizard then no components will be displayed for importing.

1. Connect System Manager to InterChange Server Express as described in "Connecting to InterChange Server Express" on page 44.
2. In the WebSphere Business Integration System Manager view, right-click the library into which you want to import components, select **Import components from server** from the context menu, and then do one of the following to start the "Import components" wizard:
   - Select **Components** from the submenu to display all types of components in the server.
   - Select a component type from the submenu to only display that type of component.

   System Manager displays the "Import the additional components from the server" wizard, as shown in Figure 13 on page 61.


Import the additional components from the server

Import the additional components from the server

Make sure you have not chosen the same object from two different servers

- WICS422DEV
  - Benchmark
  - Business Objects
  - Collaboration Objects
  - [✓] Collaboration Templates
  - Connectors
  - Database Connection Pools
  - Maps
  - Relationships
  - Schedules

[✓] Deep

[✓] Overwrite the existing components without warning

Figure 13. Importing components from the server

3. Enable checkboxes next to the servers, component groups, or individual components that you want to import from the server.
4. Enable the **Deep** checkbox if you also want to import the dependencies of the component. For more information about dependencies, see "Dependencies and references" on page 92.
5. Enable the **Overwrite the existing components without warning** checkbox if you want to overwrite any components you currently have in the library that have the same name as those you selected to import from the server.
6. Click **Finish**.

Importing components into a library from the server using drag-and-drop

Do the following to import components into an integration component library by using drag-and-drop techniques among the System Manager views:

1. Do the following in the InterChange Server Express Component Management view to select the components you want to deploy:
   - Choose all components of one type by selecting its folder in a user project or integration component library.
   - Expand the component folders and select individual components to choose them. You can use standard Windows selection techniques to select multiple user projects for deployment at the same time, such as the following:
     - Hold down **Shift** to select contiguous items
     - Hold down **Ctrl** to select non-contiguous items
2. Drag-and-drop the selected resources onto the integration component library into which you want to import the components in the WebSphere Business Integration System Manager view.

System Manager attempts to import the components to the specified integration component library. Messages and errors are displayed in the Console view.

Importing components into a library from a package

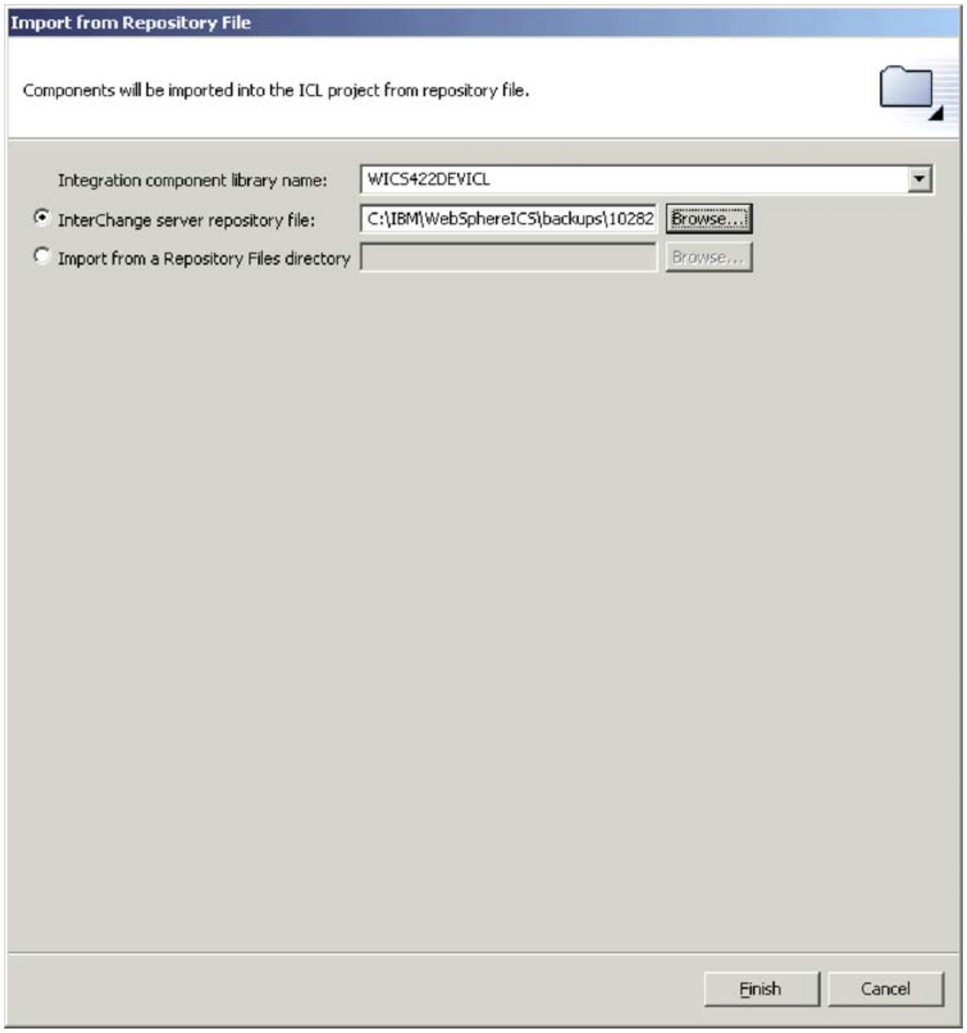
You can export integration components to a `.jar` file package, as described in "Exporting components to a package using System Manager" on page 68. This makes it easy to migrate components between environments, share them with other developers, and submit them to technical support.

Do the following to import components into an integration component library from a package:

Important: If there are components with the same name as those in the package you are importing, System Manager overwrites the existing components without a warning.

1. Right-click an integration component library and choose **Import from Repository File** from the context menu.

   System Manager displays the "Import Repository File" wizard, as shown in Figure 14.


Import from Repository File

Components will be imported into the ICL project from repository file.

Integration component library name: WICS422DEVICL

(●) InterChange server repository file: C:\IBM\WebSphereICS\backups\10282

( ) Import from a Repository Files directory

Figure 14. Importing a package

2. At the “Import Repository File” screen, ensure that the Integration Component Library Name drop-down menu contains the name of the library into which you want to import the components.

   If you launched the “Import from Repository File” wizard from a library other than the one into which you want to import the components, you can change the destination this way instead of closing the wizard and launching it again.
3. Do one of the following to specify the components you want imported:
   - To import a single package file, either type the full path and name of the `.jar` file that you want to import in the **InterChange server repository file** field or click **Browse** to select the file.
   - To import an entire directory of package files, either type the full path to the directory in the **Import from a Repository Files directory** field or click **Browse** to select the file.
4. Click **Finish**.

Note: Do not use the **File > Import** menu item in the workbench to import a package file. Although the “Zip file” wizard works with archives with the `.jar` extension, and InterChange Server Express package files have a `.jar` extension, the “Zip file” wizard does not work properly with package files.

Modifying map and collaboration object properties in integration component libraries

Maps and collaboration objects have properties that you can modify to change the behavior of those components. To modify the properties of a component, right-click either the component in an integration component library or a shortcut to it in a user project and select **Properties** from the context menu. System Manager displays a dialog to configure the properties of the component.

For more information about collaboration object properties, see Chapter 9, “Configuring collaboration objects,” on page 155.

For more information about map properties, see the *Map Development Guide*.

Creating *synonyms* for map automation

The Map Designer Express tool has a feature that allows you to create maps and reverse maps automatically between business objects that have source and destination attributes with similar attributes. To do this, Map Designer Express looks for attributes with matching names between the source and destination business objects.

System Manager enables you to enhance this basic matching process by creating multiple synonyms for attribute names of business objects.

This is done at the ICL project level in System Manager. The synonyms created are attached to the ICL, and are used by the map automation algorithm when it is invoked by Map Designer.The synonyms created are attached to the ICL project, and are used by the map automation algorithm when it is invoked by Map DesignerThe synonyms created are attached to the ICL project, and are used by the map automation algorithm when it is invoked by Map DesignerThe synonyms created are attached to the ICL project, and are used by the map automation algorithm when it is invoked by Map Designer."The synonyms created are attached to the ICL project, and are used by the map automation algorithm when it is invoked by Map Designer.""The synonyms created are attached to the ICL project, and are used by the map automation algorithm when it is invoked by Map Designer."

After they have been created, the synonyms from one ICL can be exported to a file (*.syn), and that file can be imported into another ICL. To export or import a synonyms file, right-click on the ICL, and click **Synonyms > Export** or **Synonyms > Import**.

Following are the steps for creating synonyms.

For a complete discussion of map automation, see the *Map Development Guide*.

Steps for creating synonyms

To add synonyms for attributes present in business objects, perform the following steps:

1. Right-click on an ICL in System Manager to open the Context menu. Click **Synonyms > Edit**.

   Figure 1 shows the Synonyms option in the Context menu.

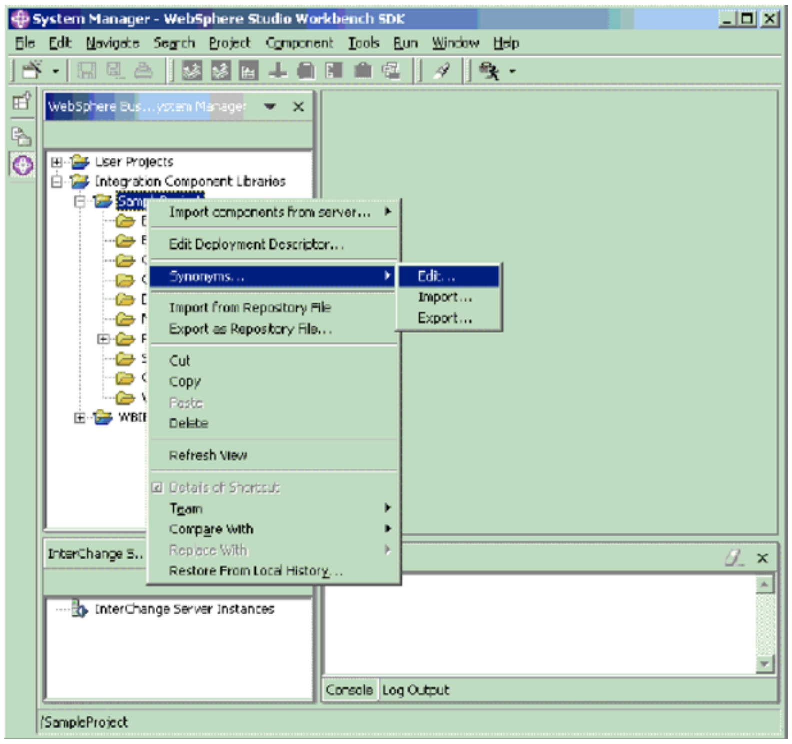
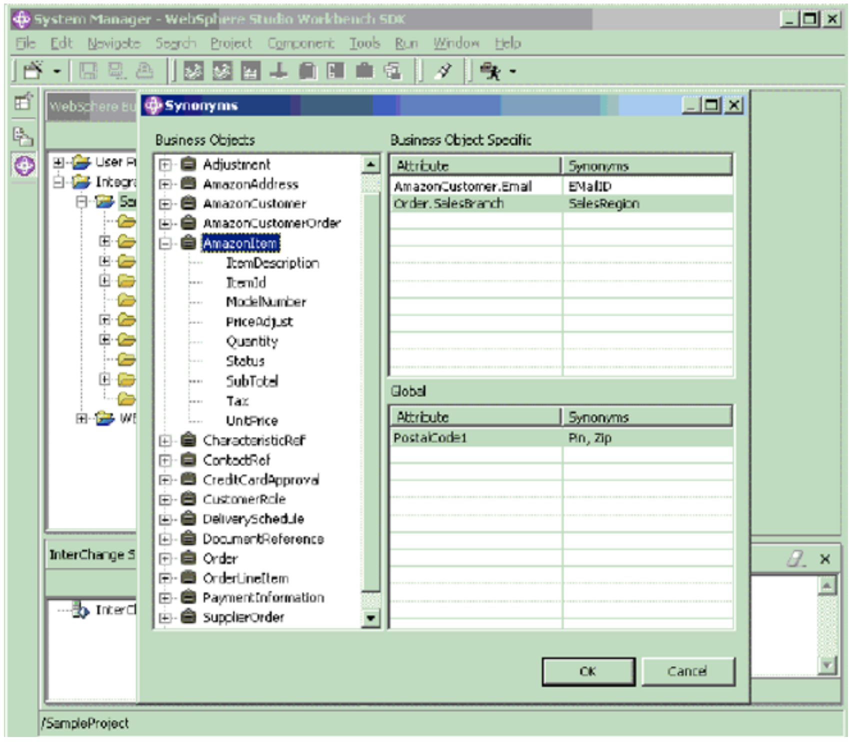

System Manager - WebSphere Studio Workbench SDK
File Edit Navigate Search Project Component Tools Run Window Help
WebSphere Bus...ystem Manager
User Projects
Integration Component Libraries
Import components from server...
Edit Deployment Descriptor...
Synonyms...
Edit...
Import...
Export...
Import from Repository File
Export as Repository File...
Cut
Copy
Paste
Delete
Refresh View
Details of Shortcut
Team
Compare With
Replace With
Restore From Local History...
InterChange S...
InterChange Server Instances
Console
Log Output
/SampleProject

Figure 15. Synonyms option in the Context menu

   Result: After you click Edit, the Synonyms window opens.

Figure 2 shows the Synonyms window.



Business Object Specific

Attribute	Synonyms
AmazonCustomer.Email	EMailID
Order.SalesBranch	SalesRegion

Global

Attribute	Synonyms
PostalCode1	Pin, Zip

Figure 16. Synonyms window

2. On the left-hand side are the business objects. On the right-hand side are two windows for configuring synonyms for attributes: Business Object Specific and Global.
   a. From the business objects on the left-hand side, select the attributes you want to use and drag and drop them onto the right-hand side under the Attribute column in the Business Object Specific window.
   b. From the business objects on the left-hand side, select the attributes you want to use as synonyms and drag and drop them onto the right-hand side under the Synonyms column of the Business Object Specific window.
   c. Repeat steps 2a and 2b to configure synonyms that apply to all the business objects in the project. Drag and drop the selected attributes under the Attribute column and the Synonyms column, respectively, in the Global window.

Result: Map Designer Express will search for all the synonyms for a given attribute and perform automatic mapping when it finds the matches.

Tip: Use the same procedure to edit attribute and synonym entries and to add comma strings under the Synonyms column.

Example: The synonym configuration shown in Figure 2 will have the following results:

- The business object specific attribute AmazonCustomer.Email will match up with EMailID.
- The business object specific attribute Order.SalesBranch will match up with SalesRegion.

- For all the business objects in the project, PostalCode1 will match up with Pin and Zip.

This topic described the procedure for setting up the synonyms in System Manager. For information about how to utilize the map automation feature when you work with Map Designer Express, see the *Map Development Guide*.

Validating database connection pools

There is a context menu item named **Validate Connection** for database connection pool components. For more information about this menu item, see "Validating database connection pools" on page 151..

Compiling maps and collaboration templates

You can compile maps and collaboration templates in System Manager, which is very convenient because components must be compiled to be deployed or exported to a package.

If the component you are compiling uses libraries that are not provided by IBM—for example, libraries in which you have created some components yourself—then you must configure System Manager to reference those libraries. For more information, see 4 on page 98 for information about the "Compiler" preferences interface.

Do one of the following to compile a map or collaboration template:
- Select the map or collaboration template in an integration component library or the shortcut to one in a user project and do one of the following:
  - Press **Ctrl+F7**
  - Select **Component > Compile** from the menu bar
- Right-click the map or collaboration template in an integration component library or the shortcut to one in a user project and select **Compile** from the context menu

Do one of the following to compile all the maps or collaboration templates in an integration component library:
- Select the **Maps** or **Collaboration Templates** folder in an integration component library and do one of the following:
  - Press **F7**
  - Select **Component > Compile all** from the menu bar
- Right-click the **Maps** or **Collaboration Templates** folder in an integration component library and select **Compile all** from the context menu

To compile a map with its submaps, right-click the map in an integration component library or the shortcut to a map in a user project and select **Compile with submap(s)** from the context menu.

Working with solutions

You can export a user project as a solution. This action copies the shortcuts from the user project as well as the component definitions that the shortcuts reference in the integration component libraries. This makes it easy to migrate an entire interface or business integration system from one environment to another.

Exporting a solution

Do the following to export a user project and the integration components it references as a solution:

1. In the WebSphere Business Integration System Manager view, expand the **User Projects** folder, then right-click the **InterChange Server Projects** folder and choose **Export Solution** from the context menu.

   System Manager displays the "Export Solution" wizard.
2. Use the following techniques to select the components you want to export:
   - Enable the checkbox next to user projects to select all of the components in the projects.
   - Enable the checkbox next to a component group to select all of the components in the group.
   - Highlight a component group and then enable checkboxes next to individual components in the right-hand pane to select those components.
3. Either type the full path and name of the directory into which the solution should be exported in the text field at the bottom of the wizard screen, or click **Browse** to navigate to the desired directory.
4. Click **Finish**.

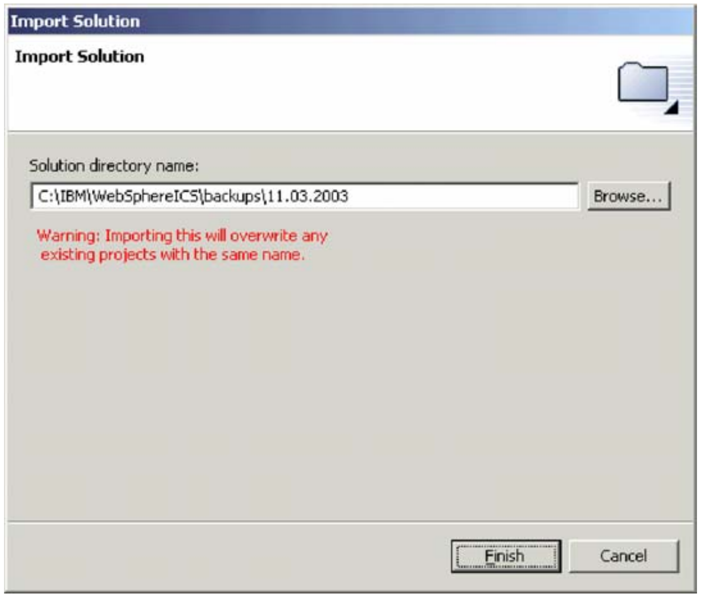
   System Manager does the following to export the solution in the directory specified in step 3:
   - Creates a `User` directory that contains the shortcuts in the user projects selected during the export of the solution.
   - Creates a `System` directory that contains the directories of the integration component library referenced by the shortcuts in the user projects selected during the export of the solution.
5. When prompted that the export operation completed successfully, click **OK**.

Importing a solution

Do the following to import a solution:

1. In the WebSphere Business Integration System Manager view, expand the **User Projects** folder, then right-click the **InterChange Server Projects** folder and choose **Import Solution** from the context menu.

   System Manager displays the "Import Solution" wizard, as shown in Figure 17 on page 68.



C:\IBM\WebSphereICS\backups\11.03.2003

Warning: Importing this will overwrite any existing projects with the same name.

Figure 17. Importing a solution

2. Either type the full path and name of the directory in which the exported solution exists into the **Solution directory name** field, or click **Browse** to navigate to the desired directory.
3. Click **Finish**.

   System Manager creates the integration component libraries and user projects defined in the exported solution in your environment.

Exporting components to a package using System Manager

You can export integration components to a package file. Integration components are resources, which are ultimately files stored in the file system as described in "Resources" on page 38. When System Manager exports components to a package, it compresses the following resources into a `.jar` (Java archive) file:

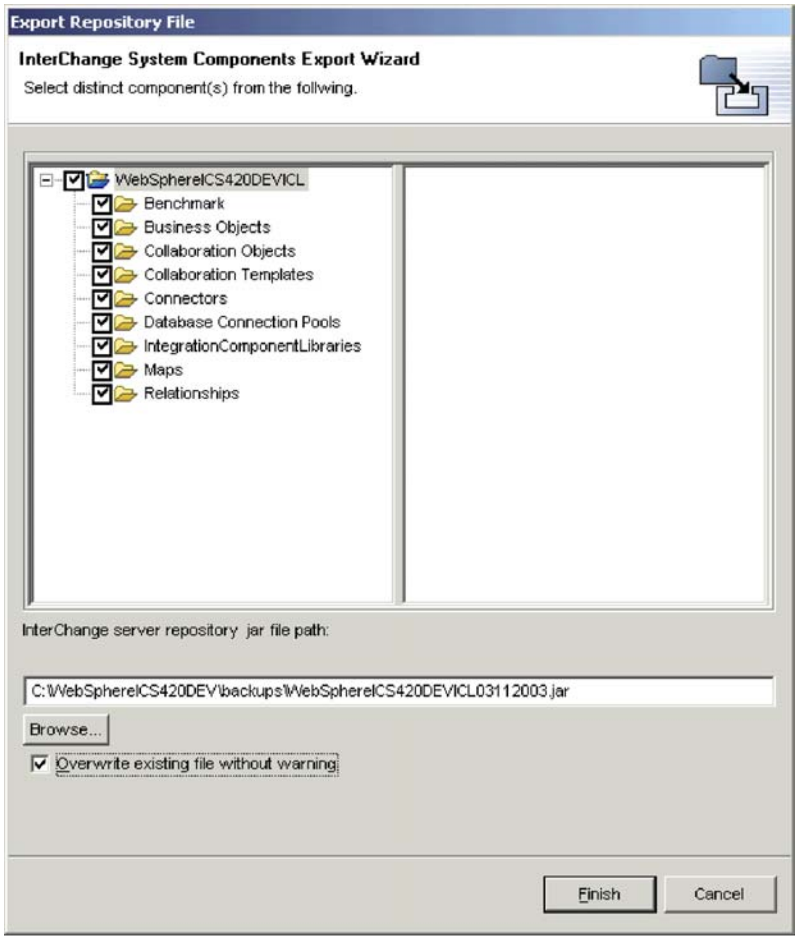
- Definition files (stored in XML format, with different extensions depending on the component type)
- Java source files for maps and collaboration templates
- Message files

Do the following to export components to a package:

Important: In order to export maps or collaboration templates they must be compiled. If you try to export uncompiled maps or collaboration templates System Manager prompts you to compile them first.

1. Right-click either an integration component library or a user project that contains the components you want to export and choose **Export as Repository File** from the context menu.

   System Manager displays the "Export Repository File" wizard, as shown in Figure 18 on page 69.


Export Repository File

InterChange System Components Export Wizard

Select distinct component(s) from the follwing.

- WebSphereICS420DEVICL
  - Benchmark
  - Business Objects
  - Collaboration Objects
  - Collaboration Templates
  - Connectors
  - Database Connection Pools
  - IntegrationComponentLibraries
  - Maps
  - Relationships

InterChange server repository jar file path:

C:\WebSphereICS420DEV\backups\WebSphereICS420DEVICL03112003.jar

☑ Overwrite existing file without warning

Figure 18. Exporting a package

2. Use the following techniques to select the components you want to export:
   - Enable the checkbox next to the integration component library or user project to select all of the components in the library or project.
   - Enable the checkbox next to a component group to select all of the components in the group.
   - Highlight a component group and then enable checkboxes next to individual components in the right-hand pane to select those components.
3. If you plan to specify an existing `.jar` file to export the components to and you want to overwrite it without receiving a prompt, enable the **Overwrite existing file without warning** checkbox.

   Note: To benefit from the **Overwrite existing file without warning** checkbox you must enable it before specifying the file to use as described in step 4. System Manager prompts you to overwrite a file as soon as it detects that an existing file has been specified and does not wait until you have finished the wizard, so you must enable this option beforehand to benefit from it.
4. Type the name and path of the `.jar` file to which the components should be exported into the **InterChange server repository jar file path** field, or click **Browse** to select a file to overwrite, or navigate to a directory and specify a file name.

If you specified the name and path to an existing file and did not enable the **Overwrite existing file without warning** checkbox and you want to overwrite the existing file then click **Yes** when prompted.

Note: If you type the name and path of the file into the field you must include the `.jar` extension in order for the **Finish** button to be enabled.

5. Click **Finish** to complete the wizard.

Deploying components to a server

You create integration components in a library in your local file system and deploy them to an InterChange Server Express instance to make them executable.

You can deploy a package of integration components using either the System Manager graphical interface or the repos_copy command-line interface. For information on using System Manager, see "Deploying components using the deployment wizard" on page 78. For information on using repos_copy, see Chapter 6, "Using repos_copy," on page 109. For information on the advantages and disadvantages of each interface, see "Deciding to use System Manager or repos_copy for deployment."

Deciding to use System Manager or repos_copy for deployment

Table 6 describes the advantages and disadvantages of using System Manager or repos_copy for deployment. Evaluate the two interfaces and use whichever one is best suited to your needs.

Table 6. Advantages and disadvantages of System Manager and repos_copy in deployment

Interface	Advantages	Disadvantages
System Manager	• System Manager uses a graphical interface. • You can selectively choose the components in the user project that you want to deploy. • You can choose to create relationship schemas in the server repository during the deployment, or can choose for them to not be created. • You can deploy multiple user projects at the same time. • With drag-and-drop deployment, you can deploy user projects, integration component libraries, or even individual components easily.	• System Manager runs only on Windows.

Table 6. Advantages and disadvantages of System Manager and repos_copy in deployment (continued)

Interface	Advantages	Disadvantages
repos_copy	• There is no requirement to have components grouped together, as they must be in a user project when deploying through System Manager. • Repos_copy is platform-independent.	• Repos_copy uses a command line interface. • You must export the components you want to deploy to a package file first, using either System Manager or repos_copy. • You cannot selectively deploy components in the package file. • You can only deploy a single package file at a time.

Setting deployment configuration properties

When you deploy a user project from System Manager to different servers, the configuration properties for some of the integration components may need to change for the new server environment. For example, an adapter in your solution may have properties that specify a database URL, a user name, and a password that are all valid for your development and testing server. But the production environment in which the solution will be deployed may require different values for these properties. Rather than manually reconfiguring the adapter (by manually using the connector configurator to change the connector configuration properties) --and other integration components that might require modification for the new environment--you can use the deployment configuration feature of System Manager.

The deployment configuration feature of System Manager enables you to define a set of values for certain integration component properties and make them applicable to a specific server (or servers) that you name. The values are stored in a deployment file (with the extension *.dfg*) associated with an Integration Component Library. When you deploy an existing user project to one of the servers that you listed in deployment configurator, System Manager uses the property values from the deployment file to modify the existing configuration files for the relevant components of the user project.

This feature can be used to modify the properties of connectors, maps, collaborations, and relationships.

To create and deploy properties using the deployment configuration feature, follow these steps:
- Open the deployment configuration dialog
- Add one or more server names
- Add integration components
- Add the properties that require transformation before deployment to a particular server
- Assign values to the properties, as appropriate for each server
- Close and save the deployment configuration
- Deploy the components

Opening the deployment configuration dialog

To open the deployment configuration dialog, do this:

- In System Manager, in the WebSphere Business Integration System Manager panel, right-click on an ICL and choose Edit Deployment Descriptor.
- The Deployment Configuration dialog appears. To create a configuration, you must first name at least one server, as described in "Adding a server name."

Adding a server name

To add a server to the deployment file, in the *Select the server* section you must designate a name for a server that will use the deployment time properties that you are creating.

- If you have previously designated a server name in the file, it will appear in the *Please select destination server* drop-down box, and you can select it again. The effect of this is to open the file that you created previously, allowing you to edit it.
- If this is the first time you have opened and used the Deployment Configuration dialog, no servers will yet be displayed in the selection box. You must provide a server name. The name that you enter for a new server does not have to be a server that currently exists or is registered. The name can be an arbitrary one--the server itself does not need to actually exist yet, and the act of naming it in this dialog does not create the server or register it. At this stage, the server name is used merely for grouping a set of specific integration component properties and values.
  1. To provide a name for a new configuration file, choose Add Server.
  2. The Add Server dialog appears. The Destination server drop-down box is prefilled with the name of your current local server instances. To create a new file with the name of a different server, overtype the prefilled selection with the new name, and choose OK.
  3. The Add Server dialog closes, and the new name that you have specified now appears in the *Please select destination server* dialog.

Adding integration components

After you have selected or named a server, you must add the integration components, properties, and values that you want to apply for use with that server.

If you have previously added elements for one server, when you later add a new server in the Deployment Configuration dialog, it will automatically be populated with the elements previously created. You can then delete or modify the elements.

If you have not previously added configuration elements in the dialog for any server, you will need to add elements to the dialog for the first time. You can do so either by importing an existing deployment file (described later in this section) or by manually adding elements in the dialog. To manually add elements, position the cursor in the Property Name column, and right-click. The Components Selection dialog appears, displaying all the available integration components in the ICL that you right-clicked to open the Deployment Configuration dialog. Expand the categories as necessary. Then, for each individual component for which you want to configure properties, click the checkbox for the component, and choose OK. Do this in turn for each individual component. When you choose OK each time, the Components Selection dialog closes, and the Deployment Configuration dialog displays, showing the new component that you have added.

Adding properties

For each component that you add, you will need to also add the properties whose values you want to set for that component. To do so:

1. Right-click in the value field next to the component for which you want to add properties. Choose Add Property.
2. The Properties Selection dialog appears, prefilled with all the properties that are available within the selected component definition.
3. Put checks in the boxes for one or more properties, and choose OK.
4. The Add Property dialog closes, and the Deployment Configuration dialog displays, showing the properties that you have added for the specific component.

Assigning values

Now you need to specify the values that you want to use for these properties. To do so:

1. Double-click in the Value field for any property.
2. Enter the value you want to use for that property.

Note: Consult the documentation for your component to determine the available values. The tool does not validate the values as you enter them.

When you are done, click OK. The Deployment Configurator dialog closes. When you open the Deployment Configurator dialog again, the last configuration that you created will be displayed by default. If you have created more than one configuration (that is, if you have named more than one server in the Deployment Configuration dialog) you can display any of the other configurations by using the *Please select destination server* drop-down box.

Deploying the property values

After you have created a deployment configuration, the integration component properties and their values can be deployed to another server. After deployment , the property values you specified during deployment configuration will be applied to the component that went into the server. You can deploy to a server in either of the following ways:

- Deploy to a registered instance of InterChange Server Express

  If the server that you named in Deployment Configurator when you created the deployment configuration properties is a registered instance of InterChange Server Express, you can use System Manager to connect to and deploy your user project to that server, as described in "Deploying components using System Manager" on page 78

  When you deploy the project, System Manager will use the deployment configuration property values to overwrite existing property values in the integration component's configuration, assuming that the System Manager preferences have been set for it to do so. Note that the original integration component definitions are not affected. The change occurs during deployment, when the values are modified in memory before being deployed to the server.

  Check the System Manager preferences setting before you deploy. To check the setting, from within System Manager go to the Windows menu and open **Preferences>System Manager Preferences>Deployment Preferences**. In the Deployment Settings display, check or uncheck the box for *Always apply Deployment configuration properties during deployment*, depending on whether you want to use the deployment configuration properties.

  When you use System Manager to deploy a user project, you can selectively choose the components in the user project that you want to deploy. The deployment configuration property values that you created will be applied only for the selected components.

- Export within a package

  Deployment configuration files are included within the package that is created when you export either user projects or individual ICLs to a repository package file, as described in "Exporting components to a package using System Manager" on page 68. When you do this for an ICL, any deployment configuration file (*.dfg file) that you created on that ICL will be included in the package. When you do this for a user project, the deployment configuration files that were created on any of the ICLs in the user project will be included in the package. If you selectively choose only certain components from an included ICL when you export the user project as a package, the deployment configuration file corresponding to that ICL will include properties and values for only those selected components.
- Export to a deployment configurator file

  You can export a file of deployment configuration properties, as described in "Export and import of deployment configuration properties." This is especially useful if you are exporting a repository file package from an InterChange Server Express repository, as described in Chapter 6, "Using repos_copy," and you want to include deployment configuration properties. When you create a deployment configuration file, the file resides within System Manager, not within the InterChange Server Express repository. Consequently, when you use repos_copy to create a package from an InterChange Server Express repository, the deployment configuration file is not included. To add it, you can export the deployment configurator file using the Export Property Descriptor dialog, and then copy the configuration content into the repos_copy package file, using this syntax:

  ```
  repos_copy -sserverName -uusername -ppassword -ireposcopyfile
  -ixdideploymentdescriptorfile
  ```

When you export an ICL or User Project from System Manager , the deployment descriptor.dfg file with the relevant entries will be included with the repository jar file. You can then use the repos_copy utility to repos copy the .jar file into the server. The values for the configuration file inside the .jar file will be automatically applied to the components before being deployed to the server.

If you do not want the transformations to happen, you can suppress them using the following option:

```
repos_copy -sserverName -uusername -ppassword -ireposcopyfile  -xdn
```

The `-xdn` option ignores the .dfg file inside the repository .jar file and sends the deployment package to the server without changing the property values.

Export and import of deployment configuration properties

After deployment configuration properties have been created in Deployment Configurator for one server, you can export those properties to a file, and then import that file as the starting point for the deployment configuration properties of another server. To do so, right-click in a field in the Property Name column, and choose Import. The Import Properties dialog appears, providing the following choices:

- Overwrite Property Descriptor

  This selection is the default. If you use this selection, all the server names, integration components, properties, and values established in the existing deployment configuration file will be removed, all the server names and properties and values established in the file being imported will be added. In effect, you are replacing the entire content of the original file with the entire content of the file being imported.

- Merge Property Descriptor

  The Merge option causes all the server names, integration components, properties, and values established in the imported file to be added to the existing file. The merge is completely hierarchical and granular. For example, if the original file contains only a Server1, and the import file contains a Server1 and Server2, the Server2 name and all the integration components, properties, and values associated with it will be added to the original file. The Server1 name already established in the original file is not removed. If an integration component exists in Server1 in the import file, but not in Server1 in the original file, that component will be added to Server1 in the original file. Likewise, if a property exists for a specific integration component in Server1 in the import file, but not in the original file, that property will be added for that specific integration component in Server1 in the import file. At the lowest level, value replacement occurs. That is, the value for a specific property of a specific integration component in a specific server name in the import file will replace the value (if one exists) for the same property of the same integration component associated with the same server name in the original file.

Copying and pasting deployment configuration properties

Deployment configuration properties created for one server can be copied and pasted into the configuration for another server. When you copy and paste properties, any values assigned to those properties are copied and pasted as well. When you paste properties, if the properties already exist in the configuration, they are overwritten by the paste operation.

Setting deployment properties without System Manager

A standalone utility, contained in the deploymentconfigutil.zip file in the product \bin directory, can allow a *.dfg file to be edited without the use of System Manager. This can be useful, for example, in a situation where a database administrator needs to have a connector password property set to a specific value for interacting with the database, but does not want to reveal that value to the implementer who is using System Manager. The implementer can use the Deployment Configurator feature of System Manager to create a *.dfg file with default values and then send that file to the database administrator. The database administrator then runs the standalone utility, which allows modification of the property values (not the addition or deletion of integration components or properties). The database administrator uses the utility to modify the password value, and then sends the file back to the implementer, who uses System Manager to import the *.dfg file into Deployment Configurator.

This utility is controlled by a pre-defined password and you must contact the product support center to get one. The password is intended to prevent users without appropriate privileges from changinge the passwords or encrypted data specified in the deployment descriptor files.

This standalone utility is contained in the deploymentconfigutil.zip file, which resides in the IBM\WebSphere\bin directory. The file can be extracted and used on a system that does not have either System Manager or InterChange Server Express installed.

Setting the initial states of components for deployment

For components with states (such as connectors, collaboration objects, maps, relationships, and database connection pools), you can set the state in which the component initializes when rebooting the server after deployment. Do the following to set the initial post-deployment state of components:

1. Select the shortcuts for the components whose initial states you want to set in a user project.

   You can use standard Windows selection techniques to choose multiple components, such as the following:

   - Hold down **Shift** to select contiguous items
   - Hold down **Ctrl** to select non-contiguous items
2. Right-click the selection made in step and do one of the following:
   - Choose **Start** or **Start All** to have the selected components initialize in a "Running" state.
   - Choose **Stop** or **Stop All** to have the selected components initialize in a "Stopped" state.

Validating a package using System Manager

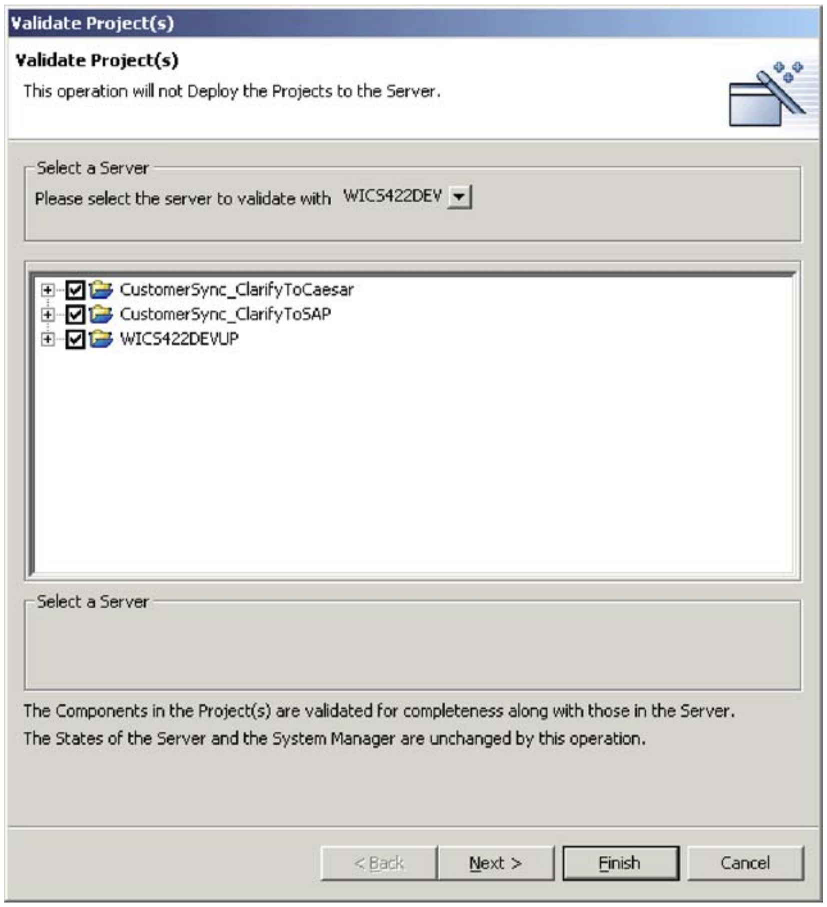
You can validate a package comprised of the components you want to deploy before performing the deployment to ensure that the deployment will succeed. Do the following to validate a package of components:

1. Add shortcuts for the components you want to validate to user projects.

   For information on creating a user project see "Creating user projects" on page 50, and for information on adding shortcuts to a user project see "Adding shortcuts to a user project" on page 51.

   Note: You can validate multiple user projects at once, so add shortcuts for all of the desired components to the appropriate user projects.
2. In the WebSphere Business Integration System Manager view, select the user projects you want to validate, right-click one of the selected user projects, and choose **Validate user project** from the context menu.

   System Manager displays the "Validate Project(s)" wizard, as shown in Figure 19 on page 77.


Validate Project(s)

Validate Project(s)

This operation will not Deploy the Projects to the Server.

Select a Server

Please select the server to validate with WICS422DEV

- [✓] CustomerSync_ClarifyToCaesar
- [✓] CustomerSync_ClarifyToSAP
- [✓] WICS422DEVUP

Select a Server

The Components in the Project(s) are validated for completeness along with those in the Server.

The States of the Server and the System Manager are unchanged by this operation.

Figure 19. Selecting components for validation

3. Select the server against which you want to validate the components from the **Please select the server to validate with** drop-down menu.
4. Use the following techniques to select the components you want to validate:
   - Enable the checkbox next to the integration component library or user project to select all of the components in the library or project.
   - Enable the checkbox next to a component group to select all of the components in the group.
   - Expand a component group and then enable checkboxes next to individual components to select those components.

   Note: If you select components with the same name from multiple user projects you will be required to select a single one from among the duplicates at the next screen of the wizard.

   If you have not selected any duplicate components, proceed to step 6 on page 78.

   If you have selected duplicate components, proceed to step 5.
5. Click **Next**.

   System Manager displays the “Local Duplicates” screen, as shown in Figure 20 on page 78.

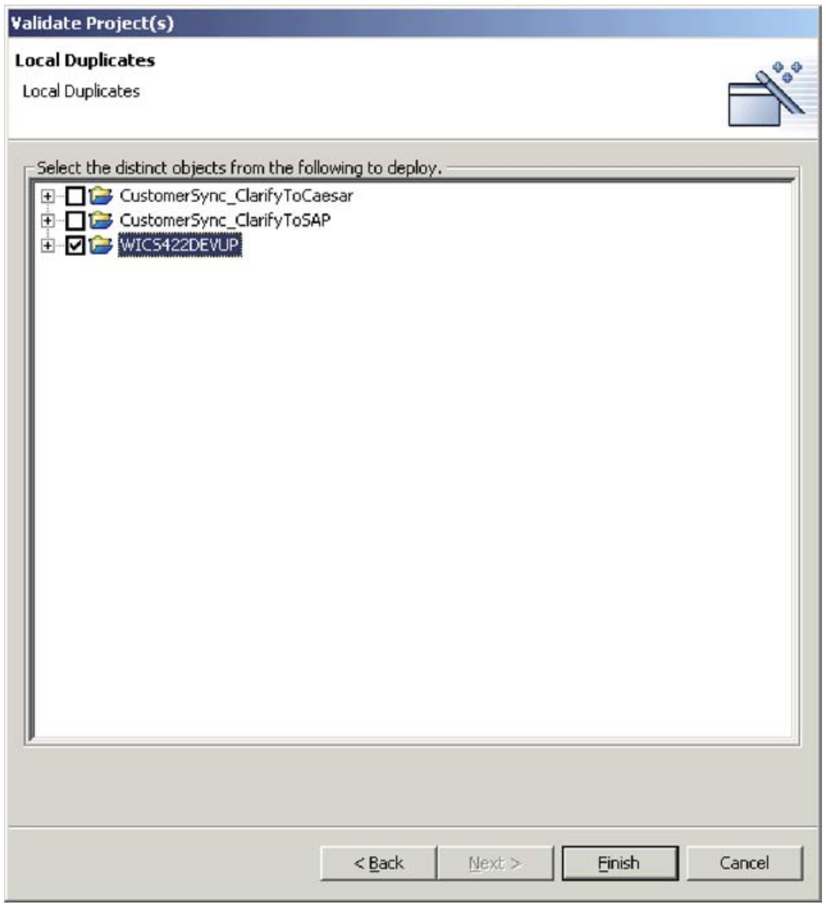

Validate Project(s)
Local Duplicates
Local Duplicates
Select the distinct objects from the following to deploy.
CustomerSync_ClarifyToCaesar
CustomerSync_ClarifyToSAP
WICS422DEVUP
< Back
Next >
Finish
Cancel

Figure 20. Selecting distinct components from user projects to validate

Select distinct components among the duplicates.

6. Click **Finish**.

   System Manager creates a package containing the selected components and validates it against the server repository. A message is displayed to indicate whether validation was successful or not.

Deploying components using System Manager

You can deploy components to an InterChange Server Express instance in one of two ways described in the following sections:

- "Deploying components using the deployment wizard"
- "Deploying components using drag-and-drop" on page 82

Deploying components using the deployment wizard

Do the following to use the deployment wizard to deploy a package of components to an InterChange Server Express instance:

1. If any of the components you want to deploy already exist in the server repository you must stop them prior to deploying them.

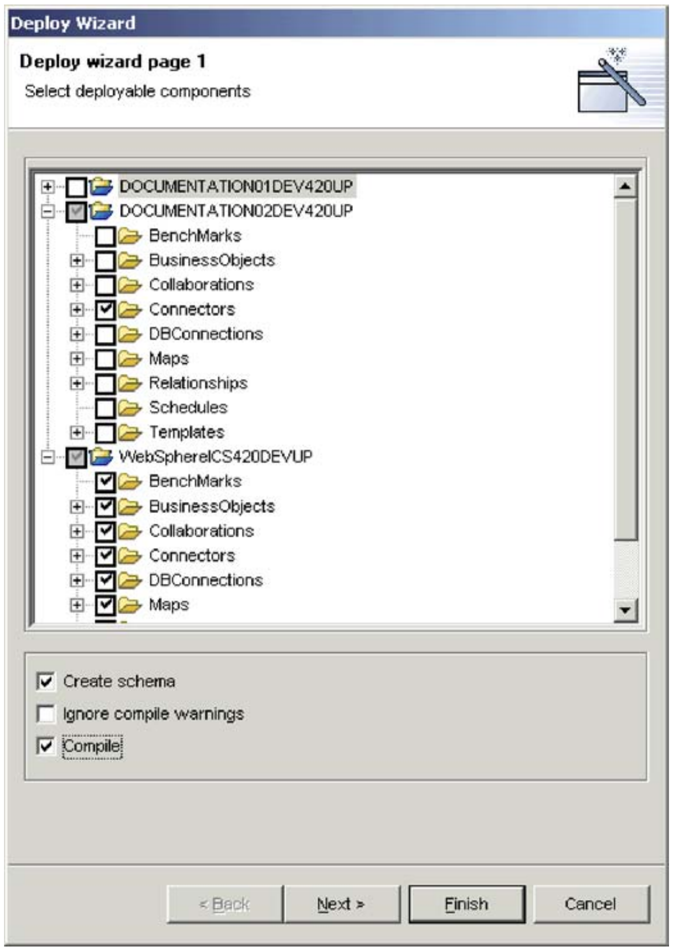
   Use the InterChange Server Express Component Management view or System Monitor to stop the duplicate components in the server before deploying a package. For more information, see "Managing component states in the repository" on page 84.

2. It is recommended that you set the initial state for the components you deploy. For more information, see “Setting the initial states of components for deployment” on page 75.
3. Add shortcuts for the components you want to deploy to a user project.

   For information on creating a user project see “Creating user projects” on page 50, and for information on adding shortcuts to a user project see “Adding shortcuts to a user project” on page 51.

   Note: You can deploy multiple user projects at once, so add shortcuts for all of the desired components to the appropriate user projects.
4. In the “WebSphere Business Integration System Manager” view, select each user project you want to deploy to the server instance. You can use standard Windows selection techniques to select multiple user projects for deployment at the same time, such as the following:
   - Hold down **Shift** to select contiguous items.
   - Hold down **Ctrl** to select non-contiguous items.
5. Right-click a selected user project and choose **Deploy user project** from the context menu.

   System Manager displays the “Deploy wizard page 1 screen”, as shown in Figure 21.


Deploy Wizard

Deploy wizard page 1

Select deployable components

- [ ] DOCUMENTATION01DEV420UP
- [✓] DOCUMENTATION02DEV420UP
  - [ ] BenchMarks
  - [ ] BusinessObjects
  - [ ] Collaborations
  - [✓] Connectors
  - [ ] DBConnections
  - [ ] Maps
  - [ ] Relationships
  - [ ] Schedules
  - [ ] Templates
- [✓] WebSphereICS420DEVUP
  - [✓] BenchMarks
  - [✓] BusinessObjects
  - [✓] Collaborations
  - [✓] Connectors
  - [✓] DBConnections
  - [✓] Maps

- [✓] Create schema
- [ ] Ignore compile warnings
- [✓] Compile

Figure 21. Selecting deployable components and deployment options

6. At the “Deployable Components” screen, do the following:

a. Select the server instance to which you want to deploy the components from the **Please select the destination server** drop-down menu.

b. Use the following techniques to select the components to deploy:
   - Enable the checkbox next to a user project to deploy all of the components in the project.
   - Expand the user project node and enable the checkbox next to entire component groups to deploy all of the components of that type.
   - Expand the user project node and a component group node and enable the checkbox next to individual components to deploy just those components.

   Note: If you select components with the same name in different user projects you will be prompted in the next screen of the wizard to choose which component among the duplicates should be deployed.

c. Enable the **Create schema** checkbox if you want to create the database connection pool and relationship schemas during the deployment process.

   The schemas for relationships and database connection pools must be created for interfaces that reference them to work. If you are deploying an interface for the purpose of running it then you must enable this option for any components the interface uses.

   Note: You should only enable this option if the settings for the database connection pools and relationships reference the appropriate connection information for the server into which you are deploying the components. If you are migrating components from one environment where the database settings were correct to another environment where they will no longer be valid, you may not want to create the schemas for these components during deployment.

d. Enable the **Ignore compile warnings** checkbox if you do not want to receive a prompt about warnings generated during compiling, such as about deprecated methods.

e. Enable the **Compile** checkbox if you want to compile the map and collaboration templates you selected for deployment.

   Maps and collaboration templates be compiled for interfaces that reference them to work.

   Note: Compiling can take a long time depending on the number of components and you may want to deploy them to the server first and compile them afterwards.

f. Do one of the following depending on your selections:
   - If you have not selected any components that are duplicates across the selected user projects and have not selected any components that already exist in the server repository then you may start the deployment at this time; proceed to step 9 on page 82.
   - If you have not selected any components that are duplicates across the selected user projects, but have selected components that already exist in the server repository, then the “Server Duplicate Objects” screen appears; proceed to step 8 on page 81.
   - If you have selected duplicate components within the user projects then proceed to step 7.

7. Click **Next**.

If you have selected duplicate components within the user projects then the "Deploy wizard page 2" screen appears, as shown in Figure 22:

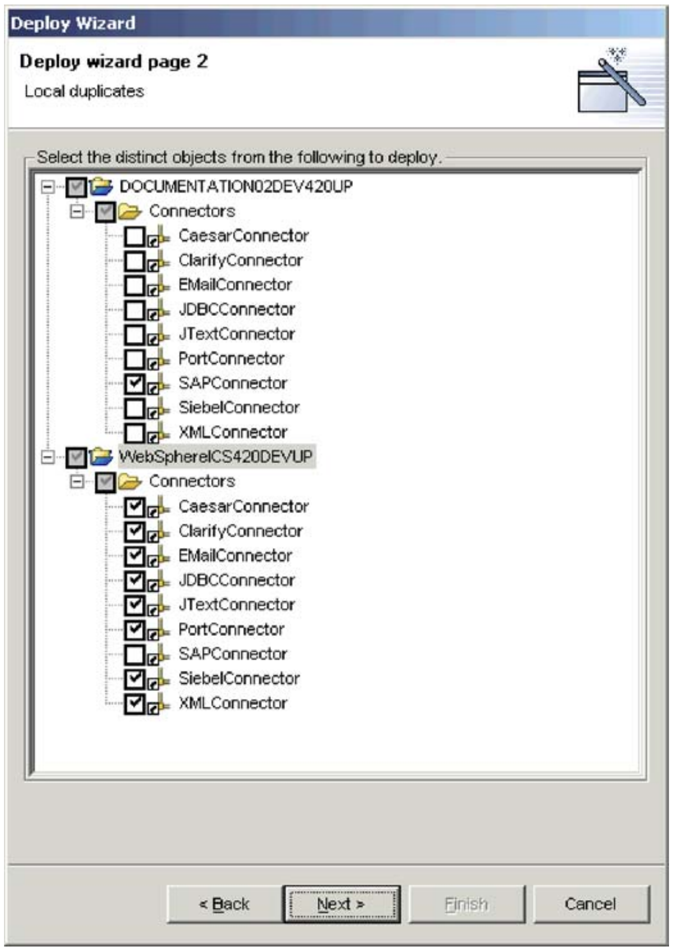


Figure 22. Selecting among local duplicates when deploying

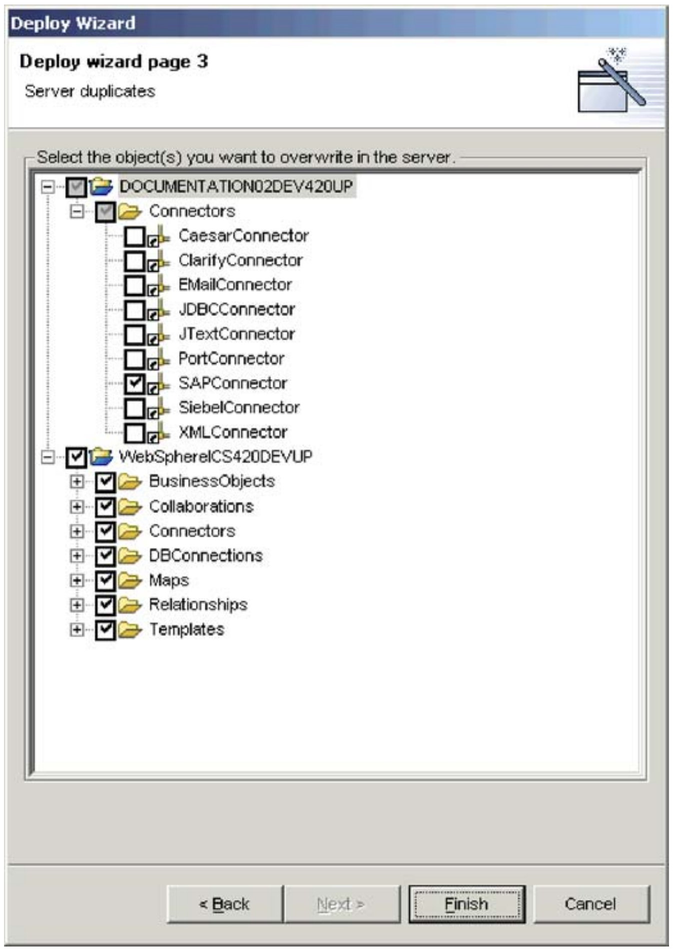
Expand the folders of the displayed user projects and enable checkboxes for the particular components you want deployed between the local duplicates.

Do one of the following depending on your selections:

- If you have not chosen to deploy any components that already exist on the server then you may start the deployment at this time; proceed to step 9 on page 82.
- If you have chosen to deploy components that already exist on the server then proceed to step 8.

8. Click **Next**.

   If you have chosen to deploy components that already exist on the server then the "Server Duplicate Objects" screen appears, as shown in Figure 23 on page 82:



Select the object(s) you want to overwrite in the server.

- ▨ DOCUMENTATION02DEV420UP
  - ▨ Connectors
    - ☐ CaesarConnector
    - ☐ ClarifyConnector
    - ☐ EMailConnector
    - ☐ JDBCConnector
    - ☐ JTextConnector
    - ☐ PortConnector
    - ✓ SAPConnector
    - ☐ SiebelConnector
    - ☐ XMLConnector
- ✓ WebSphereICS420DEVUP
  - ✓ BusinessObjects
  - ✓ Collaborations
  - ✓ Connectors
  - ✓ DBConnections
  - ✓ Maps
  - ✓ Relationships
  - ✓ Templates

Figure 23. Selecting duplicate objects to overwrite in the server

Expand the folders of the displayed user projects and enable checkboxes for the duplicate components you want to overwrite in the server repository.

9. Click **Finish**.

   System Manager attempts to deploy the selected components into the server repository. If you view the server logging output you will see messages logged as the deployment session is started and components are added to the repository.

   When the deployment session finishes System Manager either presents an informational prompt that the deployment succeeded or an error prompt that the deployment failed.

10. Take note of any significant information displayed in the prompt. For instance, the prompt might tell you to reboot the server to activate some deployed components, or might tell you that the deployment failed because active components cannot be overwritten. Click **OK** and take whatever actions were specified by the prompt.

Deploying components using drag-and-drop

Do the following to deploy components by using drag-and-drop techniques among the System Manager views:

1. If any of the components you want to deploy already exist in the server repository you must stop them prior to deploying them.

Use the InterChange Server Express Component Management view or System Monitor to stop the duplicate components in the server before deploying a package. For more information, see "Managing component states in the repository" on page 84.

2. Do the following in the WebSphere Business Integration System Manager view to select the components you want to deploy:
   - Select user projects or integration component libraries to choose all of the components in them.
   - Choose all components of one type by selecting its folder in a user project or integration component library.
   - Expand the component folders and select individual components to choose them. You can use standard Windows selection techniques to select multiple user projects for deployment at the same time, such as the following:
     - Hold down **Shift** to select contiguous items.
     - Hold down **Ctrl** to select non-contiguous items.
3. Drag-and-drop the selected resources onto the InterChange Server Express instance to which you want to deploy the components in the InterChange Server Express Component Management view.

   System Manager attempts to deploy the components to the specified InterChange Server Express instance. Messages and errors are displayed in the Console view.

   Note: If you selected duplicate components in the WebSphere Business Integration System Manager view then System Manager generates an error when you try to deploy the components. Refine your selection to include only unique components and then attempt the deployment again.

Working with components in an InterChange Server Express repository

The InterChange Server Express Component Management view allows you to manage the repositories of the InterChange Server Expresss you have registered, and to manage the components in those repositories.

Validating the components in the repository

As described in "InterChange Server modes" on page 45, InterChange Server Express has different restrictions regarding the integrity of the repository depending on the mode you start it in. Although a server running in design mode allows the repository to be inconsistent so that you can add components as they become available, a server running in production mode requires that all references and dependencies among the components be resolved. When you migrate from a design-mode server to a production-mode server, then, you should validate the repository beforehand to make sure that the repository will be in a consistent state.

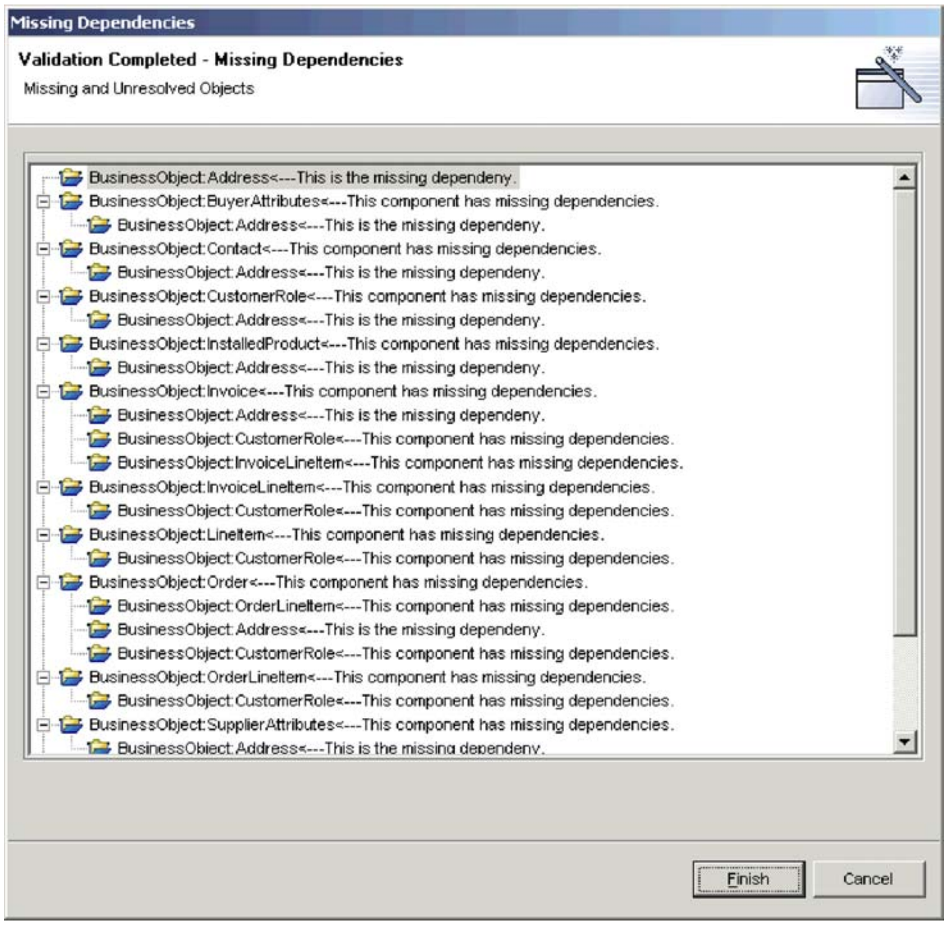
Do the following to validate the repository using System Manager:

1. In the InterChange Server Express Component Management view, right-click the server instance whose repository you want to validate and choose **Validate Repository** from the context menu.
2. If the repository is in a consistent state, an informational prompt is displayed.

   Click **OK** to close the "Validate Repository" prompt.

If there are unresolved dependencies in the repository then the “Missing Dependencies” dialog is displayed, as shown in Figure 24. Click **Finish** to close the “Missing Dependencies” dialog.

Note: You must resolve any missing dependencies before the server can start in production mode. For more information about production mode, see “Production mode” on page 46.



Validation Completed - Missing Dependencies

Missing and Unresolved Objects

```
BusinessObject:Address<---This is the missing dependeny.
BusinessObject:BuyerAttributes<---This component has missing dependencies.
    BusinessObject:Address<---This is the missing dependeny.
BusinessObject:Contact<---This component has missing dependencies.
    BusinessObject:Address<---This is the missing dependeny.
BusinessObject:CustomerRole<---This component has missing dependencies.
    BusinessObject:Address<---This is the missing dependeny.
BusinessObject:InstalledProduct<---This component has missing dependencies.
    BusinessObject:Address<---This is the missing dependeny.
BusinessObject:Invoice<---This component has missing dependencies.
    BusinessObject:Address<---This is the missing dependeny.
    BusinessObject:CustomerRole<---This component has missing dependencies.
    BusinessObject:InvoiceLineItem<---This component has missing dependencies.
BusinessObject:InvoiceLineItem<---This component has missing dependencies.
    BusinessObject:CustomerRole<---This component has missing dependencies.
BusinessObject:LineItem<---This component has missing dependencies.
    BusinessObject:CustomerRole<---This component has missing dependencies.
BusinessObject:Order<---This component has missing dependencies.
    BusinessObject:OrderLineItem<---This component has missing dependencies.
    BusinessObject:Address<---This is the missing dependeny.
    BusinessObject:CustomerRole<---This component has missing dependencies.
BusinessObject:OrderLineItem<---This component has missing dependencies.
    BusinessObject:CustomerRole<---This component has missing dependencies.
BusinessObject:SupplierAttributes<---This component has missing dependencies.
    BusinessObject:Address<---This is the missing dependeny.
```

Figure 24. Unresolved dependencies in the repository

Validating database connection pools

To validate a database connection pool, right-click the pool in the InterChange Server Express Component Management view and choose **Validate database connection** from the context menu.

For more information about database connection pools, see Chapter 8, “Configuring database connection pools,” on page 145.

Managing component states in the repository

You can use the InterChange Server Express Component Management view to manage the states of components in the repository of an InterChange Server Express instance.

This is very useful when you are developing an interface because you have to test the components you are developing. You will frequently test a component and find that you must change it. Then you will deploy the modified component to the server and test it again. To deploy a component to the server, however, it must be in an inactive state, so you must stop the component prior to deployment. Since you can develop, deploy, and test an interface all within the workbench, it is very convenient to also be able to manage the component states from it as well.

To change the state of a component in the InterChange Server Express Component Management view, right-click the component in the navigation tree and choose the desired state from the context menu. Table 7 on page 86 lists the state operations that you can apply for each of the component types that have runtime states. For more information on the behavior of the system when components are in different states, see the *System Administration Guide*.

Table 7. State operations available for components in System Manager

Component type	State operation
Collaboration object	• Start • Stop • Pause • Shut Down
Connector	• Start • Stop • Pause • Shut Down • Boot
Map	• Start • Stop
Relationship	• Start • Stop

You can also select multiple components, right-click one of the components, and apply state changes to all of them at the same time. This is very useful when you must deploy a number of components at the same time, because you do not have to perform a separate state-management operation for each one. You can use standard Windows selection techniques to select multiple user projects for deployment at the same time, such as the following:

- Hold down **Shift** to select contiguous items.
- Hold down **Ctrl** to select non-contiguous items.

Note: If you have any components that do not have states selected when you try to perform a state management operation on a group of selected components, then the state management options will not be displayed in the context-sensitive menu. For instance, if you want to stop several connectors, but accidentally select a business object definition as well, then the state management operations will not be exposed in the context-sensitive menu.

To modify the state all of the components of a particular type, right-click the folder for the component type in the InterChange Server Express Component Management view and select **Start All** or **Stop All** from the context menu.

It is recommended that you use System Monitor to manage components in a production installation. System Monitor offers greater flexibility and is a dedicated administration tool. For more information on using System Monitor, see the *System Administration Guide*.

Modifying component properties in the repository

Maps, connectors, and collaboration objects have runtime properties that you can modify to change the behavior of those components. Follow the instructions in the following sections to modify the properties for particular components:

- "Modifying collaboration object properties" on page 87
- "Modifying map properties" on page 87
- "Modifying connector properties" on page 88

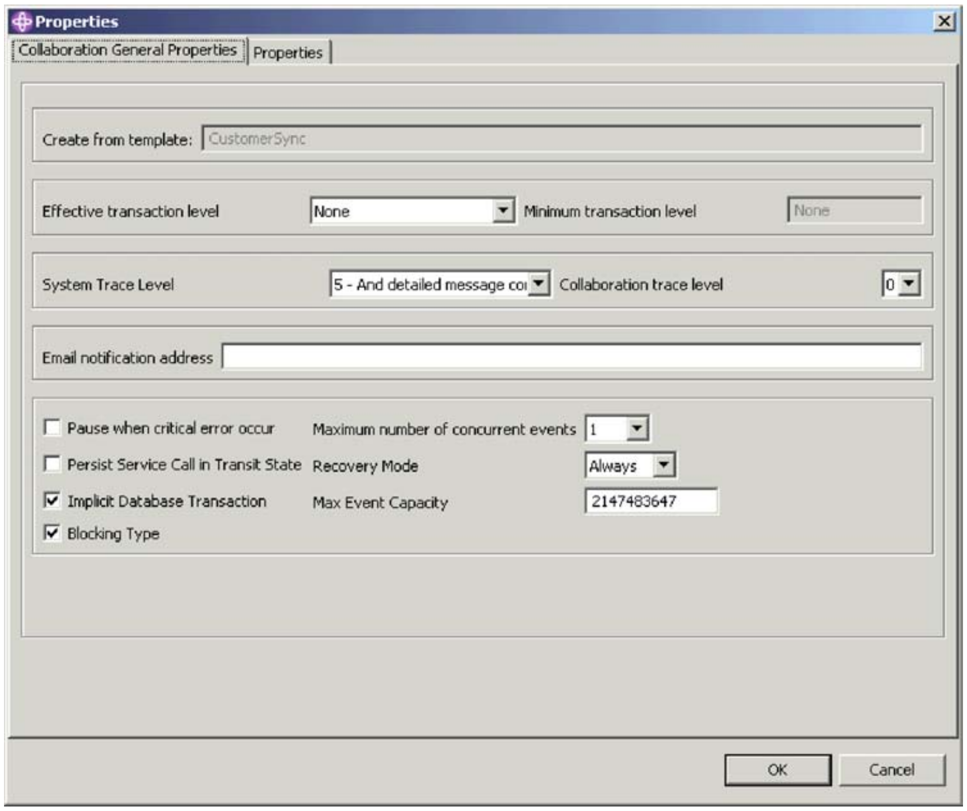
Modifying collaboration object properties

Do the following to modify the properties of a collaboration object:

1. Right-click the collaboration object whose properties you want to modify in the InterChange Server Express Component Management view and select **Properties** from the context menu.
2. Set the properties on the "Collaboration General Properties" and "Properties" tabs to the desired values.

   For more information about collaboration object properties, see Chapter 9, "Configuring collaboration objects," on page 155.

   Figure 25 shows the "Properties" dialog for collaboration objects:



Create from template: CustomerSync

Effective transaction level: None
Minimum transaction level: None

System Trace Level: 5 - And detailed message co
Collaboration trace level: 0

Email notification address:

- [ ] Pause when critical error occur
- [ ] Persist Service Call in Transit State
- [x] Implicit Database Transaction
- [x] Blocking Type

Maximum number of concurrent events: 1
Recovery Mode: Always
Max Event Capacity: 2147483647

Figure 25. Modifying collaboration object properties

3. Click **OK**.

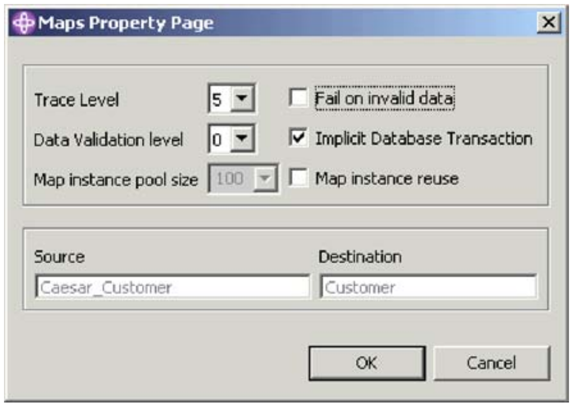
Modifying map properties

Do the following to modify the properties of a map:

1. Right-click the map whose properties you want to modify in the InterChange Server Express Component Management view and select **Properties** from the context menu.
2. Set the properties on the "Maps Property Page" dialog to the desired values.

   For more information about map properties, see the *Map Development Guide*.

   Figure 26 on page 88 shows the "Maps Property Page" dialog:


Maps Property Page

Trace Level: 5

Data Validation level: 0

Map instance pool size: 100

[ ] Fail on invalid data

[✓] Implicit Database Transaction

[ ] Map instance reuse

Source: Caesar_Customer

Destination: Customer

Figure 26. Modifying map properties

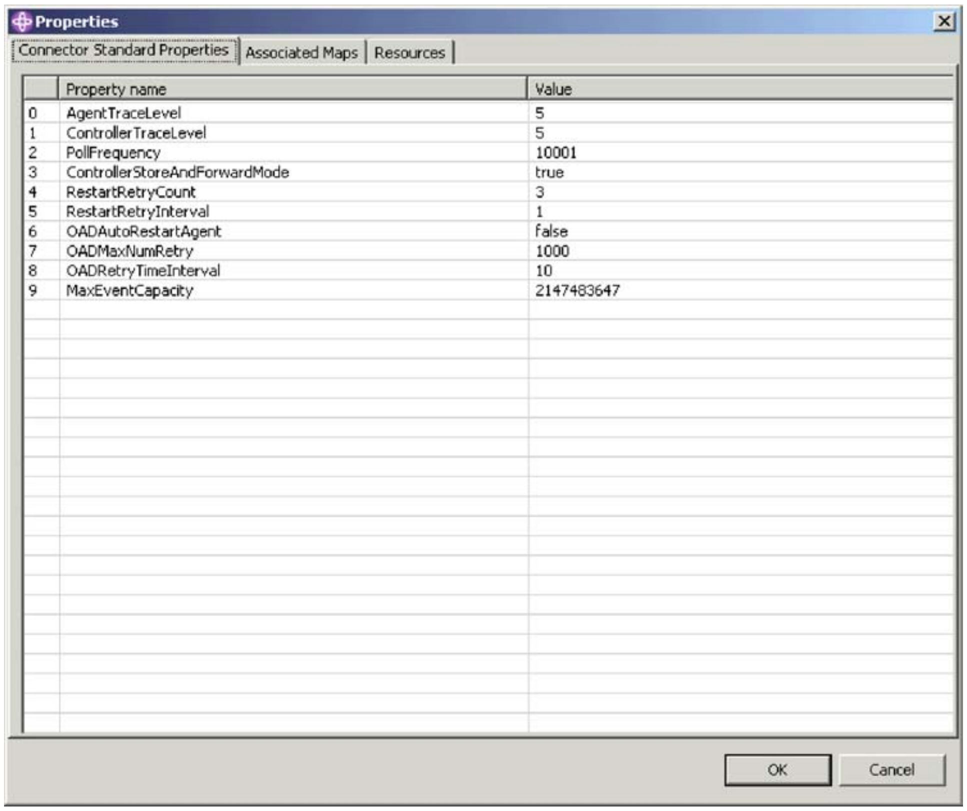
3. Click **OK**.

Modifying connector properties

Do the following to modify the properties of a connector:

1. Right-click the connector whose properties you want to modify in the InterChange Server Express Component Management view and select **Properties** from the context menu.
2. Set the properties on the “Connector Standard Properties”, “Associated Maps”, and “Resources” tabs to the desired values. For more information about connector properties, see Chapter 7, “Configuring connectors,” on page 121.

   Figure 27 on page 89 shows the “Properties” dialog for connectors:



	Property name	Value
0	AgentTraceLevel	5
1	ControllerTraceLevel	5
2	PollFrequency	10001
3	ControllerStoreAndForwardMode	true
4	RestartRetryCount	3
5	RestartRetryInterval	1
6	OADAutoRestartAgent	false
7	OADMaxNumRetry	1000
8	OADRetryTimeInterval	10
9	MaxEventCapacity	2147483647

Figure 27. Modifying connector properties

3. Click **OK**.

Compiling components in the repository

Collaboration templates must be compiled for the collaboration objects that are based on them to run, and maps must be compiled to run as well. You can deploy collaboration templates and maps to an InterChange Server Express running in design mode without compiling them, but must ensure that they are compiled to test them.

To compile a collaboration template in the server, right-click the template in the InterChange Server Express Component Management view and choose **Compile** from the context menu.

To compile all of the collaboration templates in the server, right-click the **Collaboration Templates** folder in the InterChange Server Express Component Management view and choose **Compile All** from the context menu.

To compile a map in the server, right-click the map in the InterChange Server Express Component Management view and choose **Compile** from the context menu.

To compile all of the maps in the server, right-click the **Maps** folder in the InterChange Server Express Component Management view and choose **Compile All** from the context menu.

To compile a map along with all of its submaps, right-click the map in the InterChange Server Express Component Management view and choose **Compile with submaps** from the context menu.

The "Console" view displays messages about the success or failure of compilation attempts.

Deleting components from the repository

As you develop a business integration system, the design of the system frequently changes. New requirements and environmental differences commonly result in the redesign of an interface. You will want to make sure that the repository contains only the components it needs to solve business problems according to the current design, and that will sometimes mean that you have to delete components that belong to designs that have changed.

Deleting the entire repository

To delete all of the components in a server repository using System Manager do the following:

1. In the InterChange Server Express Component Management view, right-click the server instance whose repository you want to delete and choose **Delete Repository** from the context menu.
2. At the "Delete repository confirmation" prompt click **OK**.
3. At the next "Delete repository confirmation" prompt click **OK**.

Deleting components using the component browser

Do the following to delete components from an InterChange Server Express repository by using the component browser:

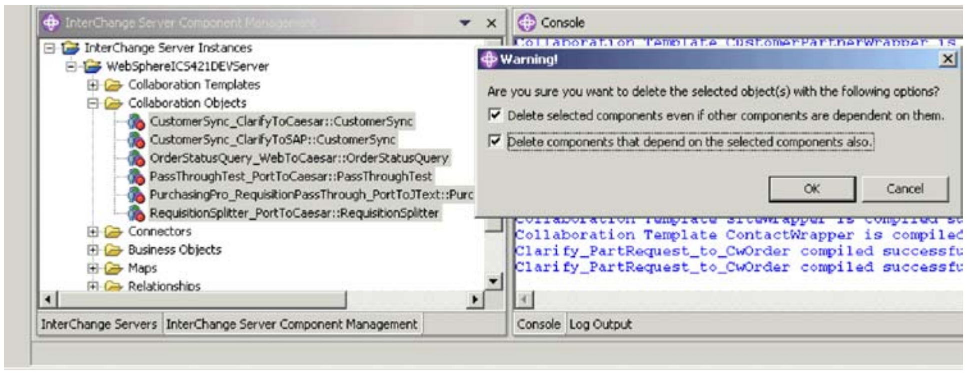
1. In the InterChange Server Express Component Management view, select the components you want to delete in the browser.

   You can use standard Windows techniques for selecting multiple items, such as holding **Shift** to select multiple contiguous items, and holding **Ctrl** to select multiple non-contiguous items.
2. To delete a component you must first make sure it is in an inactive state. Before attempting to delete any components, ensure that they are in an inactive state by stopping them.

   To stop a component using the InterChange Server Express Component Management view in System Manager, see "Managing component states in the repository" on page 84.

   To stop a component using System Monitor, see the *System Administration Guide*.
3. Do one of the following to delete the selected components:
   - Press **Delete**
   - Right-click one of the selected components and choose **Delete** from the context menu
   - Select **Edit > Delete** from the menu bar

   Figure 28 on page 91 shows System Manager when the component browser is used to delete components.


InterChange Server Component Management
InterChange Server Instances
WebSphereICS421DEVServer
Collaboration Templates
Collaboration Objects
CustomerSync_ClarifyToCaesar::CustomerSync
CustomerSync_ClarifyToSAP::CustomerSync
OrderStatusQuery_WebToCaesar::OrderStatusQuery
PassThroughTest_PortToCaesar::PassThroughTest
PurchasingPro_RequisitionPassThrough_PortToJText::Purc
RequisitionSplitter_PortToCaesar::RequisitionSplitter
Connectors
Business Objects
Maps
Relationships
InterChange Servers
InterChange Server Component Management
Console
Warning!
OK
Cancel
Console
Log Output

Are you sure you want to delete the selected object(s) with the following options?

[✓] Delete selected components even if other components are dependent on them.

[✓] Delete components that depend on the selected components also.

```
Collaboration Template CustomerPartnerWrapper is
Collaboration Template SiteWrapper is compiled s
Collaboration Template ContactWrapper is compiled
Clarify_PartRequest_to_CwOrder compiled successfu
Clarify_PartRequest_to_CwOrder compiled successfu
```

Figure 28. Deleting components using the component browser

4. Do the following when presented with the dialog:
   - Enable the **Delete selected components even if other components are dependent on them** checkbox to delete the component even if other components reference it.

     For instance, a business object cannot be deleted if it contains other business objects because they depend on it, even if those dependent components are also selected for deletion, unless this option is enabled.
   - Enable the **Delete components that depend on the selected components also** checkbox to also delete the dependencies of the selected components. For more information about dependencies, see "Dependencies and references" on page 92.
5. Click **Finish**.

   If the delete operation is successful then the wizard exits.

   If the delete operation failed, click **OK** to close the error prompt, then resolve the problem and try the operation again. The error information in the prompt can be very difficult to use in troubleshooting, so it is recommended that you use the server logging output to determine the cause of the failure.

Deleting components using the Server Object Delete Wizard

Do the following to use System Manager to delete components from an InterChange Server Express repository by using the Server Object Delete Wizard:

1. In the InterChange Server Express Component Management view, right-click the server instance from whose repository you want to delete components and choose **Server object delete** from the context menu.
2. To delete a component you must first make sure it is in an inactive state. Before attempting to delete any components, ensure that they are in an inactive state by stopping them.

   To stop a component using the InterChange Server Express Component Management view in System Manager, see "Managing component states in the repository" on page 84.

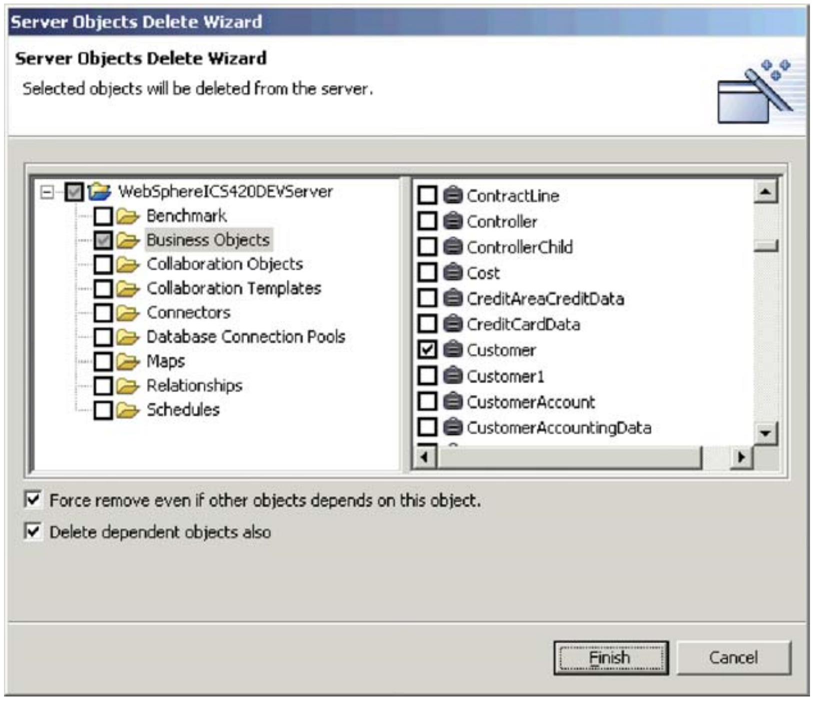
   To stop a component using System Monitor, see the *System Administration Guide*.
3. Do the following to select which components you want to delete:
   - Enable the checkbox for the server instance to delete all of the components in the repository.
   - Expand the folder for the server instance and enable the checkbox for a component type to delete all objects of that type.

- Select the folder for a component type to populate the right-hand pane with a list of the components of that type in the repository and then enable the checkbox for individual components.

4. Enable the **Force remove even if other objects depend on this object** checkbox to delete the component even if other components reference it.

   For instance, a business object cannot be deleted if it contains other business objects because they depend on it, even if those dependent components are also selected for deletion, unless this option is enabled.
5. Enable the **Delete dependent objects also** checkbox to also delete the dependencies of the selected components. For more information about dependencies, see "Dependencies and references."

   Figure 29 shows the "Server Objects Delete Wizard".


Server Objects Delete Wizard

Server Objects Delete Wizard

Selected objects will be deleted from the server.

- [✓] WebSphereICS420DEVServer
  - [ ] Benchmark
  - [✓] Business Objects
  - [ ] Collaboration Objects
  - [ ] Collaboration Templates
  - [ ] Connectors
  - [ ] Database Connection Pools
  - [ ] Maps
  - [ ] Relationships
  - [ ] Schedules

- [ ] ContractLine
- [ ] Controller
- [ ] ControllerChild
- [ ] Cost
- [ ] CreditAreaCreditData
- [ ] CreditCardData
- [✓] Customer
- [ ] Customer1
- [ ] CustomerAccount
- [ ] CustomerAccountingData

[✓] Force remove even if other objects depends on this object.

[✓] Delete dependent objects also

Figure 29. Deleting objects from the server

6. Click **Finish**.

   If the delete operation is successful then the wizard exits.

   If the delete operation failed, click **OK** to close the error prompt, then resolve the problem and try the operation again. The error information in the prompt can be very difficult to use in troubleshooting, so it is recommended that you use the server logging output to determine the cause of the failure.

Dependencies and references

Integration components depend upon one another to perform their roles in the business integration system. For instance, business object definitions can contain other business object definitions as children, collaboration templates have business object definitions associated with their ports, and connectors have maps associated with their supported business objects. These dependencies must be satisfied for the system to function properly, and when InterChange Server Express is started in production mode it checks to make sure that all dependencies and references are resolved. If it detects any unresolved dependencies or references then InterChange Server Express fails to start in production mode.

The terms **dependencies** and **references** are used to describe the relationship between components, depending on the context. For example, a connector definition requires the business object definitions it supports and the maps associated with those business object definitions to exist so that it can exchange data with InterChange Server Express. In this context, the business object definitions and maps are **dependencies** of the connector definition. If you view the same relationship between the business object definition and the connector definition, but in the context of the business object, the connector is one of the **references** of the business object definition—along with any other business objects that contain it as a child, any maps that transform it, any collaboration templates that support it for their port definitions, and so forth.

Table 8 specifies the components that can be dependencies and references for each component type.

Table 8. Integration component dependencies and references

Component	**Dependencies**	**References**
Business objects	• Business objects	• Business objects • Maps • Connectors • Collaboration templates • Collaboration objects
Relationships	• Business objects	None
Connectors	• Business objects • Maps	• Collaboration objects
Maps	• Business objects • Maps	• Connectors • Collaboration objects
Collaboration templates	• Business objects	None
Collaboration objects	• Business objects • Maps • Connectors • Collaboration templates • Collaboration objects	• Collaboration objects

Showing dependencies and references

You can use System Manager to show the dependencies and references for an integration component.

To show the dependencies for a component, right-click it in System Manager and select **Show Dependencies** from the context menu. The “Dependency Tree” wizard appears. For more information, see “Using the Dependency Tree” on page 52.

To show the references for a component, right-click it in System Manager and select **Show References** from the context menu. The “Object References” window appears.

Dependencies that cannot be detected by the system

There are some dependency and reference relationships that cannot be automatically detected and enforced by the system. Table 9 describes the components for which the system cannot determine dependencies and references.

Although the system is unable to detect these dependencies and references, they must still be satisfied for the system to operate properly. You must maintain accurate project documentation to make sure that when you deploy an interface from one environment to another you include any of these extra dependencies.

Table 9. Components for which the system cannot determine dependencies and references

Component	Explanation
Relationships	Maps and collaboration templates can use relationships, but they do so through APIs in Java code. The system cannot parse all the code in all maps and collaboration templates to detect the use of relationships.
Database connection pools	Maps and collaboration templates can use database connection pools, but they do so through APIs in Java code. The system cannot parse all the code in all maps and collaboration templates to detect the use of database connection pools.
Data handlers	Connectors, as well as other components, can use data handlers to format data. Data handlers are not managed by the system at all, and it therefore cannot detect them.
any component used exclusively in code	Although the system can determine that a map depends upon a business object when you have drag-and-dropped the business object definition into the map as a source or destination, it cannot detect any use of components that occurs exclusively through custom Java code. If you instantiate a new business object or manually invoke a map in a map or collaboration, the system is not able to detect that dependency.
Schedules	The system does not currently support the detection of dependencies and references for schedules.

Standard operations available for multiple workbench resources

Although many of the tasks you perform in the workbench are sensitive to the particular resource you are working with or the context in which you are working, there are many operations that affect all resources the same way. This section describes the tasks you can perform in the workbench that work similarly for all resources.

Cutting, copying, and pasting resources

You can cut, copy, and paste resources both in System Manager and in the file system.

To cut, copy, or paste a user project, integration component library, integration component, shortcut, or folder in System Manager, right-click the resource and choose the desired menu item. If you copy integration components, you cannot paste them into the same library (for instance, to copy a business object definition and paste it into the same library to specify a different name and use it as a template). You can, however, open the component definition in its designer tool and do a “Save as” operation to save it with a new name to the same library.

To cut, copy, and paste integration components or shortcuts in the file system, launch Windows Explorer and navigate to the appropriate subdirectory within the project directory, copy the file that shares the name of the component, and paste it into the appropriate subdirectory within the destination project directory.

If you use cut, copy, and paste operations in the file system, you must refresh the integration component library or user project in System Manager in order to see the newly added resources. For more information, see "Refreshing resources."

You cannot just cut, copy, and paste whole user projects or integration component libraries, because there are meta-data references maintained in the workbench that specify which user projects and libraries exist. Copying a folder into the workspace directory does not update those meta-data references. You can, however, create a new user project or library in System Manager to satisfy the meta-data entries, and then paste the folders of component definitions into the directory for the new library or user project.

Refreshing resources

If you add component definitions to a library or shortcuts to a user project by cutting, copying, and pasting files in the file system (as described in "Modifying map and collaboration object properties in integration component libraries" on page 63) then you must refresh the library in System Manager so that the changes are reflected.

To refresh an integration component library or user project, right-click it in System Manager and choose **Refresh View** from the context menu.

Deleting resources

Do the following to delete a workbench resource:

1. Either right-click the resource in System Manager and choose **Delete** from the context menu or select the resource in System Manager and press the **Delete** key.
2. When presented with the "Delete Component" dialog, click **OK**.

Note: If a component has dependencies then you will not be allowed to delete it.

Using Eclipse-based workbenches

This section describes some of the optional tasks you may want to perform in your tooling framework to make your experience with the tools more efficient.

Opening and closing perspectives

This section describes how to open and close perspectives.

Opening perspectives

Do the following to open a perspective in the workbench:

1. Select **Window > Open Perspective > Other** from the menu bar of your workbench.
2. At the "Select Perspective" dialog, choose the perspective you want to open and click **OK**.

Closing perspectives

You can do the following to close perspectives:

- Select **Window > Close Perspective** to close the currently active perspective.
- Select **Window > Close All Perspectives** to close all currently open perspectives
- Right-click the icon for a perspective in the perspective shortcut bar and choose **Close** from the context menu to close that perspective.
- Right-click the icon for a perspective in the perspective shortcut bar and choose **Close All** from the context menu to close all open perspectives.

Showing and closing views

You can control the panes that are displayed in the WebSphere WorkBench perspectives.

Showing views

Do the following to show a view:

1. Select **Window > Show View > Other**.
2. Expand the folder for the view group, such as **ICS Control View**.
3. Select the particular view, such as **InterChange Server Component Management**.
4. Click **OK**.

Closing views

To close a view, do one of the following:

- Right-click the title bar of the view and choose **Close** from the context menu.
- Click the close button in the title bar of the view.

Customizing perspectives

You can customize a perspective to include the perspectives, views, wizards, and plug-in interfaces that you want so that you can minimize the number of times you have to open elements you use frequently and close elements you do not need. Do the following to customize the currently active perspective:

1. Select **Window > Customize Perspective** from the menu bar.
2. Click on the node you want to customize to expand it.
3. Enable and disable checkboxes for the node elements. Table 10 lists the customizable perspective nodes and the effect that enabling them has.

Table 10. Customizable perspective nodes

Customize perspective node	Result
File > New	Adds or subtracts items from the **File > New** menu.
Window > Open Perspective	Adds or subtracts perspectives from the **Window > Open Perspective** menu.
Window > Show View	Adds or subtracts views from the **Perspective > Show View** menu.
Other	Adds or subtracts from the menu bar and toolbar. For instance, you must enable the **ClearCase** checkbox to display the **ClearCase** menu.

Saving perspectives

You can save a perspective configuration to preserve customizations you have made. Do the following to save a perspective:

1. Select **Window > Save Perspective As** from the tooling framework menu bar.
2. Type a name for the perspective in the **Name** field.
3. Click **OK**.

Setting the default perspective

By default your tooling framework opens to the `Resource` perspective. If you primarily use the tooling framework to work with integration components, you may want to make one of the IBM InterChange Server Express perspectives the default. Do the following to do so:

1. Select **Window > Preferences** from the menu bar of the tooling framework.
2. Expand the **Workbench** node.
3. Select the **Perspectives** node under the **Workbench** node.
4. Select the desired perspective (such as `System Manager`) from the **Available Perspectives** list.
5. Click **Make Default**.
6. Click **OK**.

Configuring System Manager preferences

To set your System Manager preferences, do the following:

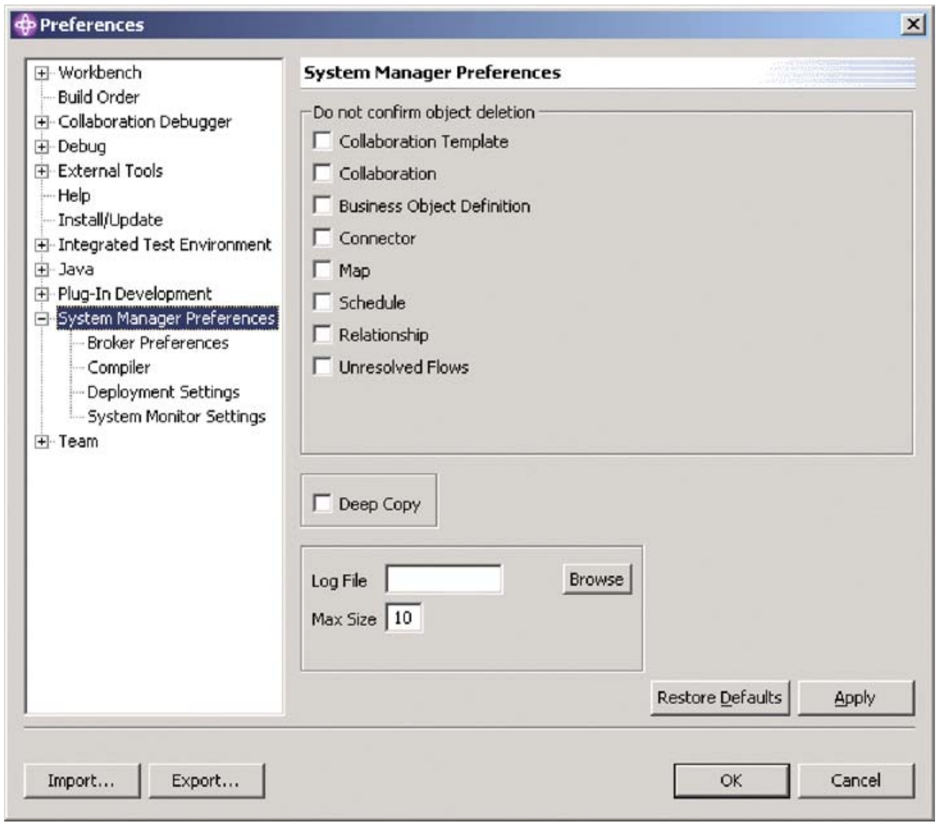
1. Select **Window > Preferences** from the menu bar of the workbench.
2. Select **System Manager Preferences** and do the following to configure the available preference options:
   - If you do not want to be prompted when you delete a component from an integration component library, enable the checkbox for the component type in the "Do not confirm object deletion" pane.

     Note: Enabling the Unresolved Flows checkbox does not have any effect on whether or not you receive a prompt when you delete a flow from Flow Manager. For information on Flow Manager and how to configure preferences for it, see the *System Administration Guide*.
   - Enable the **Deep Copy** checkbox if you want to copy a component's dependencies along with the component when you perform a copy operation on it.

     If **Deep Copy** is enabled and you copy a business object definition from one library to another, then all of the child business objects it contains are copied as well, for example. If **Deep Copy** is not enabled, however, and you copy a business object definition from one library to another, then only the business object itself is copied.

     For more information on dependencies, see "Dependencies and references" on page 92.
   - Type the name and path of a file in the **Log File** field, or click **Browse** to select a file. When errors occur in System Manager, the error information is written to the file specified. Type a number in the **Max Size** field to specify the maximum size of the log file in megabytes.
   - Click **Restore Defaults** to set the preferences elements to their default values.

   Figure 30 on page 98 shows the System Manager preferences interface.



System Manager Preferences

Do not confirm object deletion

- [ ] Collaboration Template
- [ ] Collaboration
- [ ] Business Object Definition
- [ ] Connector
- [ ] Map
- [ ] Schedule
- [ ] Relationship
- [ ] Unresolved Flows

- [ ] Deep Copy

Log File

Max Size 10

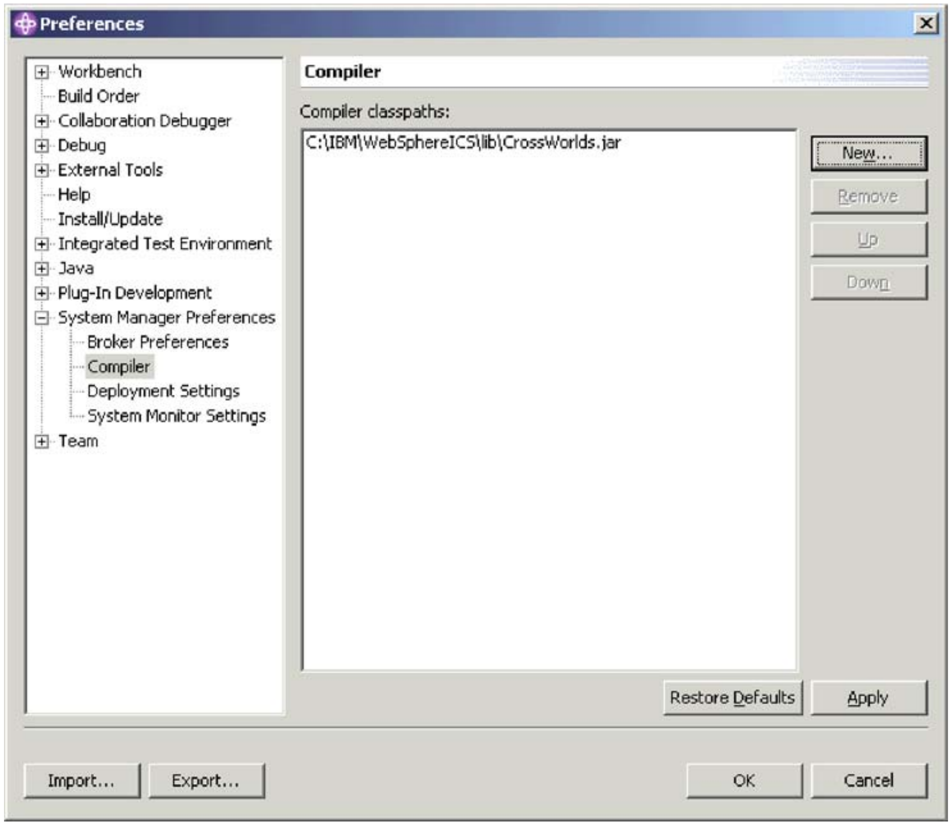
Figure 30. System Manager preferences

3. The “Broker Preferences” preference setting should not be used. Do not make any entires under this preference setting.
4. The “Compiler” preferences interface allows you to specify the location of libraries required by maps or collaborations that you develop so that the compiler can locate them.

   Do the following to set your compiler preferences:

   a. Expand **System Manager Preferences** and then select **Compiler**.
   b. Click **New**.
   c. At the “Add Classpath” dialog, navigate to a library that you want to add to the compiler classpath, select the library, and click **Open**.

   Figure 31 on page 99 shows the “Compiler” preferences interface:



Compiler classpaths:

C:\IBM\WebSphereICS\lib\CrossWorlds.jar

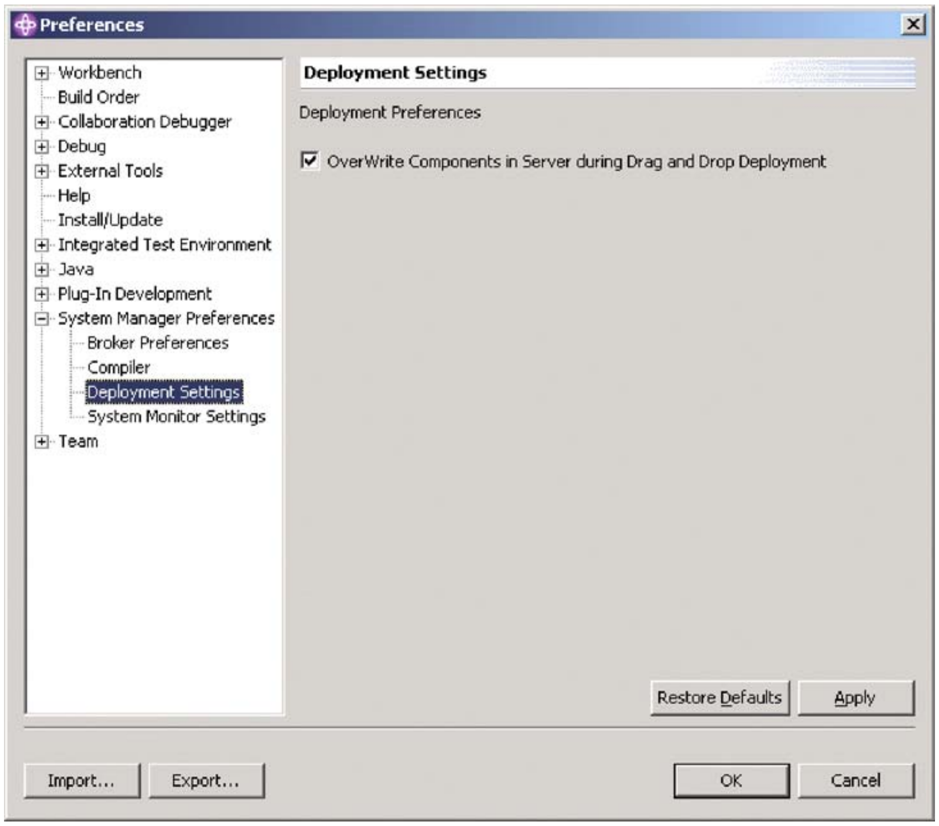
Figure 31. Compiler preferences

5. The “Deployment Settings” interface allows you to set options that are relevant when you deploy components to a server from System Manager.

   Do the following to set your deployment preferences:

   a. Expand **System Manager Preferences** and then select **Deployment Settings**.
   b. Enable the **Overwrite Components in Server during Drag and Drop deployment** checkbox.

   Figure 32 on page 100 shows the “Deployment Settings” interface:



Deployment Settings

Deployment Preferences

☑ OverWrite Components in Server during Drag and Drop Deployment

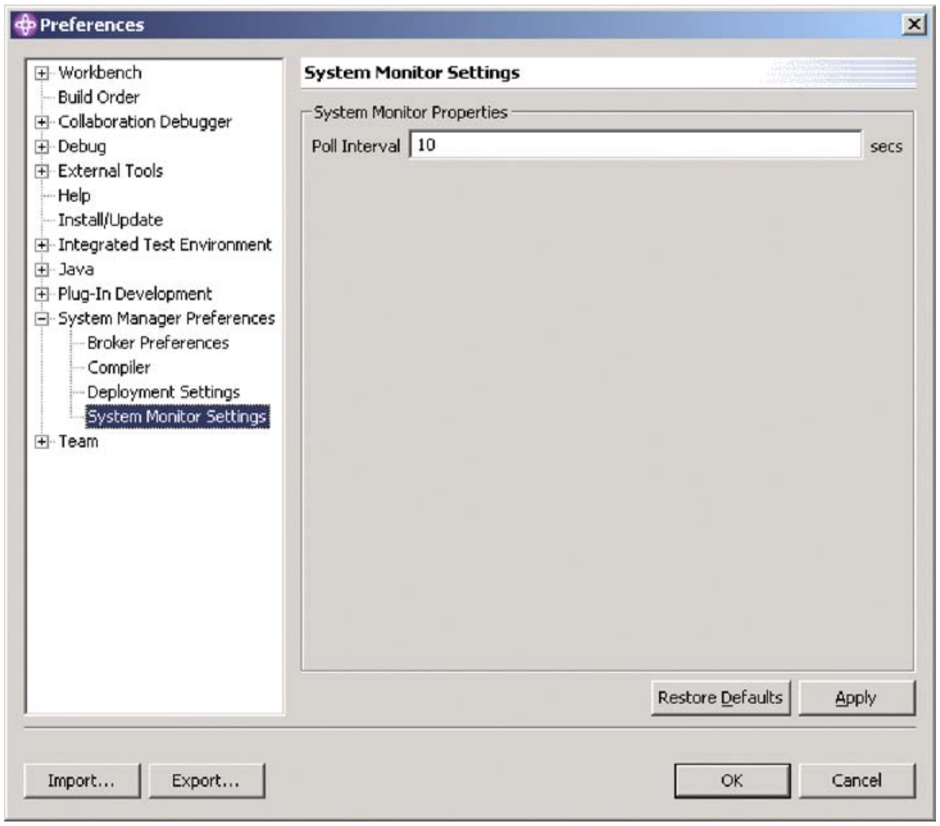
Figure 32. Deployment Settings interface

6. The "System Monitor Settings" interface allows you to configure the behavior of the monitoring features of the InterChange Server Express Component Management view.

   Do the following to set your System Monitor preferences:

   a. Expand **System Manager Preferences** and then select **System Monitor Settings**.
   b. Type the number of seconds in the **Poll Interval** field that you want System Manager to wait in between checking the system state to display information in the System View view.

   Figure 33 on page 101 shows the "System Monitor Settings" interface:



System Monitor Settings

System Monitor Properties

Poll Interval 10 secs

Figure 33. System Monitor Settings

For more information about the System View view, see the *System Administration Guide*.

7. Click **Apply** to save your preferences and continue working in the "Preferences" dialog, or click **OK** to save your preferences and exit the dialog.

Troubleshooting problems connecting to InterChange Server Express in System Manager

A failed connection between System Manager and InterChange Server Express is unlikely after you have validated your installed system as described in the *Quick Start Guide.* However, if at a later time System Manager is unable to connect to InterChange Server Express, make sure the following requirements are still being met in your current installation:

- InterChange Server Express must be running in order for System Manager to connect to it. Examine the logging output for InterChange Server Express to ensure that it has a logging statement which reads "<*server name*>" is ready", where <*server name*> is the name of your InterChange Server Express instance.
- The IBM Java Object Request Broker (ORB) must be running for clients such as the tools to communicate with it.
- When register, you must specify the name of InterChange Server Express exactly as it exists. If you use the wrong case or leave out a single character when trying to register a server instance, then System Manager will be unable to connect. You can be sure of the exact name of the server with the following techniques:
  – Examine the name exactly as it appears in whatever interface is used to start the server. On Windows, the server name is commonly supplied as a

parameter to the `start_server.bat` batch file. The value is in the **Target** field of the shortcut that invokes the batch file.

  – Examine the name as it is displayed in the InterChange Server Express logging output. The log entry "<*server name*>" is ready" indicates that the server has started and by what name it can be identified.

- You must specify the correct user name, which by default is `admin`.
- You must specify the correct password. The default password is `null`, but it can be changed, as described earlier in this guide. If you cannot connect when specifying the password `null`, make sure that no who works with the server instance has changed the password.

  If you have cached the user name and password, the password value sometimes gets corrupted. It will still appear as four asterisks, so you will not intuitively think that its value has changed. Delete the cached value in the **Password** field and type the password again.

Chapter 5. Configuring InterChange Server Express

InterChange Server Express must be configured with environment-specific information for it to run. The necessary configuration for a typical implementation is done for you, using default values, when you run the Installer, as described in the *WebSphere Business Integration Server Express Installation Guide*.

However, if you need to make a modification in the configuration of InterChanger Server Express for your specific implementation, you can do so, using System Manager.

Configuration information for InterChanger Server Express is stored in XML format in a file named `InterchangeSystem.cfg`, which is stored in the product directory. You should not modify this file directly. Instead, if you need to make modifications to the configuration of InterChange Server Express, use System Manager. This chapter provides instructions for that.

Using System Manager to configure InterChange Server Express

To use System Manager for configuration of InterChange Server Express, an instance of InterChange Server Express must be running and System Manager must be connected to it.

Do the following to open the InterChange Server Express configuration editor in System Manager:

1. Start System Manager as described in "Starting System Manager" on page 40.
2. Connect System Manager to the InterChange Server Express instance as described in "Connecting to InterChange Server Express" on page 44.
3. Right-click the InterChange Server Express instance and choose **Edit Configuration** from the context menu.
4. Make the desired changes in the configuration tabs as described in the following sections:
   - "Viewing general properties using System Manager"
   - "Configuring database properties using System Manager" on page 104
   - "Configuring tracing levels using System Manager" on page 105
   - "Configuring logging and tracing properties using System Manager" on page 105
   - "Configuring e-mail notification properties using System Manager" on page 106
   - "Configuring miscellaneous properties using System Manager" on page 106
   - "Configuring environment variables using System Manager" on page 107
   - "Configuring WebSphere MQ properties using System Manager" on page 107
5. When you are done use the keyboard shortcut **Ctrl+S** to save the file.

Viewing general properties using System Manager

The "General" tab of the System Manager configuration file editor has the following read-only fields:

- The **Server name** field, which identifies the name of the InterChange Server Express instance.

- The **Server version** field, which identifies the version of IBM WebSphere Integration Server Express software the server is running.
- The **Total memory** field, which identifies the amount of RAM on the computer running InterChange Server Express.
- The **Free memory** field, which identifies the amount of RAM that is available on the computer running InterChange Server Express.
- The **Startup time** field, which identifies the time at which the server instance started.
- The **Local Time Zone** field, which identifies the time zone in which the computer running System Manager resides.
- The **Server Time Zone** field, which identifies the time zone in which the computer running InterChange Server Express resides.
- The **Server locale** field, which identifies the locale of the computer on which InterChange Server Express is running.

Configuring database properties using System Manager

The "Database" tab of the configuration file editor allows you to change properties that specify the databases InterChange Server Express uses to store repository, event management, and transaction management data. Do the following to configure the database properties:

1. Click the **Database** tab.
2. Select the appropriate value in the **Database Driver** drop-down menu**—**either `IBM DB2 Server`or `MQ SQL Server`**—**depending on the database vendor.
3. Type the maximum number of connections you want InterChange Server Express to make with the database server in the **Max Connections** field, or enable the **Unlimited** checkbox to allow InterChange Server Express an unlimited number of connections.
4. Type the maximum number of pools InterChange Server Express should establish to contain the database connections it caches in the **Max pools** field.
5. Type the number of minutes you want a connection object to remain idle before being returned to the database connection pool for reuse in the **Idle Timeout** field.
6. Type the number of times that the server should attempt to execute a deadlocked database transaction in the **Max deadlock retry** field.

   This field is related to the **Deadlock retry interval** field documented in step 7.

   For more information on the behavior of the server in the case of a database deadlock, see the *System Administration Guide*.
7. Type the number of seconds that the server should wait in between attempts to execute a deadlocked database transaction in the **Deadlock retry interval** field.

   This field is related to the **Max deadlock retry** field documented in step 6.

   For more information on the behavior of the server in the case of a database deadlock, see the *System Administration Guide*.
8. Do the following in the "Event Management" pane to configure the database connectivity for the event management service:
   a. If you chose `MQ SQL Server(Type 4)`in the **Database Driver** drop-down menu, then type the name of the computer on which the database server resides in the **Host name** field.
   b. Type the name of the database in the **Database** field.

c. Type the maximum number of connections you want InterChange Server Express to make with the specific database server in the **Max Connections** field, or enable the **Unlimited** checkbox to allow InterChange Server Express an unlimited number of connections.

d. Type the user name that should be used by InterChange Server Express to log in to the specified database in the **Login** field.

e. Type the password for the user name in the **password** field.

f. If you chose `MQ SQL Server(Type 4)`, or `Oracle(Type 4)` in the **Database Driver** drop-down menu, then you must type the port number through which clients communicate with the database server in the **Port Number** field.

9. Repeat step 8 in the “Transactions” pane to configure the database connectivity for the transaction management service.
10. Repeat step 8 in the “Repository” pane to configure the database connectivity for the repository service.

Changing database passwords

Do the following to change the password for the user name that InterChange Server Express uses to access one of the databases:

1. Click **Change** next to the **Password** field in the “Event Management”, “Transactions”, or “Repository” panes.
2. Type the new password in the **New Password** field.
3. Type the new password again in the **Confirm Password** field.
4. Click **OK**.

Configuring tracing levels using System Manager

Do the following to set properties for flow tracing and subsystem tracing at the “Tracing Levels” tab:

1. Click the **Tracing Levels** tab.
2. To set the flow trace level for a business object, select the desired value in the **Level** column for the particular business object in the “Flow Tracing Levels” pane.

   For more information on flow tracing, see the *System Administration Guide*.
3. To set the trace level for an InterChange Server Express subsystem, select the desired trace level from the drop-down menu associated with the subsystem.

Configuring logging and tracing properties using System Manager

Do the following to configure the locations to which InterChange Server Express logs and traces runtime information:

1. Click the **Trace/Log Files** tab.
2. Do the following in the Logging pane to configure how InterChange Server Express logs runtime information:
   - Enable the **To Console** checkbox if you want InterChange Server Express to output logging information to the console.
   - Enable the **To File** checkbox if you want InterChange Server Express to output logging information to a file. Do the following steps as well if you enable this checkbox:
     a. Type the name and path of the file to which output should be logged in the unlabeled text field, or click the browse button to select a file.

b. Do one of the following to specify how the log files should be managed:
   - Click the **Unlimited** checkbox to have InterChange Server Express write information to the file specified in step 2a on page 105 without a limit on the size of the file.

     Note: Be sure to manage the file well if you make this selection, because InterChange Server Express could potentially fill up the disk and cause a system failure.
   - Do the following to configure a log file archiving approach:
     1) Select **Byte**, **KB**, **MB**, or **GB** from the unlabeled drop-down menu to indicate whether you want to store the log and archive files in increments of bytes, kilobytes, megabytes, or gigabytes.
     2) Select a number from the **Log file** drop-down menu to indicate the number of bytes, kilobytes, megabytes, or gigabytes you want the log file to grow to before it is archived.
     3) Select the number of archive files that should be kept before they are deleted in the **Number of archives** drop-down menu.

3. Repeat step 2 on page 105 for the "Tracing" pane.
4. To enable tracing for WebSphere MQ, type the name and path of the file to which output should be logged in the unlabeled text field in the "WebSphere MQ Tracing" pane, or click the browse button to select a file.
5. To use stack tracing, select **Enable Stack Tracing** and then select the desired severity level from the **Tracing Severity Level** drop-down menu.

   For more information about stack tracing, see *System Administration Guide*.

Configuring e-mail notification properties using System Manager

InterChange Server Express can send e-mail notifications when errors occur in the business integration system. By default the server uses the Java e-mail APIs, and can send e-mail notifications for a number of different subsystems. To enable the server to send e-mail notifications for errors experienced by individual collaborations, you must configure the system to use the Adapter for e-Mail. For detailed information on configuring the system for e-mail notification, see the *System Administration Guide*. To make the necessary modifications to the configuration file (which is just one step in the total number of tasks described in the *System Administration Guide*), do the following:

1. Click the **E-mail** tab.
2. Select `Java mail` or `Connector mail` from the **E-mail send type** drop-down menu depending on which e-mail notification mechanism you want to use.

   Note: Keep in mind that collaborations will only be able to send e-mail notifications if you choose `Connector mail`.
3. If you chose `Java mail` in step 2, type the name of the computer on which the SMTP server is installed in the **SMTP mail host** field.
4. Type a valid e-mail address or comma-separated series of e-mail addresses in the text fields for each subsystem as desired.

Configuring miscellaneous properties using System Manager

The "Misc" tab in the System Manager configuration file editor has panes for configuring features such as persistent monitoring, flow control, and long-lived business processes. Do the following to configure settings for these features:

1. Click the **Misc** tab.
2. Do the following in the "Persistent Monitoring" pane:
   a. Select `Continue` in the **Action on error** drop-down menu if you want InterChange Server Express to continue running in the event of errors experienced by the persistent monitoring subsystem.

      Select `Shutdown` in the **Action on error** drop-down menu if you want InterChange Server Express to shut down in response to errors with the subsystem.
   b. Select the desired tracing level in the **Persistent monitoring service** drop-down menu to specify the tracing level for the subsystem.
3. For information on the interface elements in the "Flow Control" pane, see the *System Administration Guide*.
4. For information on the interface elements in the "Flow Monitoring" pane, see the *System Administration Guide*.
5. In the "Workflow Management" pane, type a number in the **Business object consumer processor** to indicate how many threads you want spawned to manage business objects participating in a long-lived business process.

   These threads deliver business objects from the collaboration runtime to the appropriate connector controllers when the business objects are retrieved from persistent storage to resume processing in a long-lived business process. For more information on long-lived business processes, see the *Collaboration Development Guide*.
6. For information on the "Server Memory" pane, see "Using the memory checker thread" on page 281.

Configuring environment variables using System Manager

At the "Environment Properties" tab, you can specify any Java user environment properties that might be required in the business integration system. Some environments may require such a variable to be specified; instead of adding them to the script that starts InterChange Server Express, you can add them to the configuration file by using the "Environment Properties" tab. Do the following to add a new environment property:

1. Click the **Environment Properties** tab.
2. Type the name for the property in the **Property name** field.
3. Type the value for the property in the **Property value** field.
4. Click **Add Property**.

To delete a property, select the property and click **Delete Property**.

To edit a property's value, select the property and edit the text in the **Property value** column for the property.

Configuring WebSphere MQ properties using System Manager

Do the following to configure the connectivity between InterChange Server Express and the WebSphere MQ persistent messaging software:

1. Click the **WebSphere MQ** tab.
2. Type the name of the computer on which the WebSphere MQ server is installed in the **Host name** field.
3. Type in the **Port number** field the port over which WebSphere MQ allows clients to communicate with the server.

   The default port is `1414`.

4. Type in the **Queue manager name** field the name of the queue manager that has been created to contain the queues on which InterChange Server Express will persistently store messages for the flows it processes.
5. Type in the **Channel** field the name of the channel over which clients of WebSphere MQ communicate with the WebSphere MQ server. The default value is `CHANNEL1`. Consult with the WebSphere MQ administrator at the site to determine if `CHANNEL1` can be used and if not, which channel can be used.

Chapter 6. Using repos_copy

`Repos_copy` is a command line interface for working with integration components and InterChange Server Express repositories. It allows you to deploy a package—a collection of integration components—to a server repository, or to export components from the repository to a package.

To run `repos_copy`, enter the command in an MS-DOS command prompt window. The *ProductDir*`/bin` directory, where the utility resides, should be in your path as a result of installation.

Note: The `repos_copy` output file contains encrypted passwords for relationships and connector applications. If you try to edit the output file and change these passwords, `repos_copy` will not work.

Important: When `repos_copy` deploys components to the repository, it deploys them to the repository *only*. It does *not* deploy them to any in-memory tables of business object definitions. For instance, connectors load business object definitions from the repository into their memory space when they start. If you deploy a business object definition to the repository to update it, you must restart the connector agent so that it loads the modified business object definition into memory. You must therefore stop and restart InterChange Server Express and components that load definitions into memory for them to load recently deployed components.

This chapter has the following sections:
- "Repos_copy syntax" on page 109
- "Repos_copy usage scenarios" on page 113
- "Locale for repos_copy files" on page 119

For more information about backing up the system, see the *System Administration Guide*.

Repos_copy syntax

Table 11 on page 110 describes the options of repos_copy and their arguments, and shows the correct case usage for the options and the lack of spacing between the option and its argument. The syntax shows that the options between curly braces ({}) represent a set of options that are required. If you do not specify the `-u`, `-p`, `-i`, `-o`, or `-s` options at the command line, then `repos_copy` prompts you for them. If you do not specify them when prompted, repos_copy does not execute. Options enclosed in brackets ([]) are optional.

```
repos_copy [-sserverName][-uusername][-ppassword]
{-i[filename1][-rrelationshipName[relationshipName2]][[-k][-ai|-ar|-arp]
[-xcompilePackage][-vp|-vr]}
{-o[outfilename[[-fEntityFile][-eEntityType:Entity1[+EntityType:Entity2][+...]]
[-deep][-summary]}
{[-d]|[-doEntityType:Entity[+EntityType:Entity2][+...]|
[-dfoEntityType:Entity[+EntityType:Entity2][+...]}
{-v}
{-vr}
```

```
{[-xCompileAll]|[-xCompileAllCollabs]|[-xCompileAllMaps]|
[-xCompileCollab:collabTemplateName[+collabTemplateName][+...]]|
[-xCompileMap:nativeMapName[+nativeMapName][+...]]}
```

Table 11. Repos_copy command options

Option	**Description**
`-ai`	Ignore and do not load any duplicate objects (business objects, maps, relationships, collaboration templates and objects, and connectors) that are found when deploying a package.
`-ar`	Replace any duplicate objects (business objects, maps, relationships, collaboration templates and objects, and connectors) that are found when deploying a package.
`-arp`	This is an interactive version of the `-ar` option. If the components in the package being deployed already exist in the repository then repos_copy displays a prompt asking if you want to ignore or replace the component.
`-d`	Deletes the components in the repository, except the state data. Use this option to delete all of the components from the repository.
`-deep`	Used with the `-e` option when you want to include all the dependent components. If you omit the `-deep` option, only the component that is specified with the `-e` option will be included.
`-dfo`*`EntityType:Entity[+EntityType:Entity2]`*	This option is the same as the `-do` option except that it will forcefully delete the component even if the component has referents that depend on it. This option only works with the repository of a server that is running in design mode. A server that is running in production mode does not permit unresolved dependencies and references.
`-do`*`EntityType:Entity[+EntityType:Entity2]`*	Specifies the entities to be deleted from the repository. See Table 12 on page 113 for the list of entity types and keywords. If the object has no referents—other components that depend on it—then the deletion takes place. If the object has referents, then the deletion fails and a message is displayed. The behavior is the same in both design mode and production mode. For more information about starting the server in design mode or production mode, see the *System Implementation Guide*.

Table 11. Repos_copy command options (continued)

Option	Description
`-e`*`EntityType:Entity1[+EntityType:Entity2...]`*	Exports one or more referenced first-class entities. A first-class entity is a business object, collaboration object, collaboration template, connector, database connection pool, map, or relationship. You identify the entity to load or unload by specifying one of the keywords in Table 12 on page 113. Follow the *`EntityType`* keyword with a colon (:) and the name of the entity. Use the "+" to specify more than one entity. When combined with the `-o` option, the `-e` option unloads the data to an output file.
`-f`*`entityFile`*	This option is similar to the `-e` option except that the names of the entities to be imported are stored in a file. The file should contain references to the entities, with the following conditions: • The entity names must follow after the proper entity type keyword. The entity types and their keywords are listed in Table 12 on page 113. • A colon must separate the entity type from the entity name. • There must be a new line separating each entity reference. When combined with the `-o` option, this option exports the components to a package.
`-i`*`filename`*	Deploys the specified package file to the repository. If you omit the input file name value, the command interactively prompts you to enter the name of the input file. The file can be a `.jar` file containing objects in XML format. The `.jar` files created by repos_copy or System Manager have a particular structure which must be maintained for any subsequent imports of such a file to be successful. You should not, therefore, ever modify an input file manually.
`-k`	Overrides the default behavior of repos_copy when it finds a Mercator map in the package file it is loading. By default, repos_copy exits if it encounters a Mercator map. If you use the `-k` option, repos_copy skips over any Mercator maps in the package file and proceeds with the deployment process.
`-mode`	Returns the mode of the server. For more information about InterChange Server Express modes, see the *Implementation Guide for WebSphere InterChange Server*.
`-nc`*`encoding`*	For a list of valid character encodings, see the Java documentation about the String class.

Table 11. Repos_copy command options (continued)

Option	Description
-o*outfilename*	Exports the components in the repository to the specified package file. You must specify the name of the package file. If the file already exists then repos_copy prompts you to overwrite it or not. The output file is in `.jar` format, and contains the component definitions in XML format, as well as `.java` source files for components that have them. This option cannot be combined with the `-i` or `-d` options, nor can it export components in text formats. Repos_copy does not append the `.jar` extension, so you must specify it when specifying the name of the output file.
-p*password*	Specifies the password for the user name supplied with the -u option. The password case-sensitive. If you do not specify this option then repos_copy prompts you for the password.
-r*	This option is similar to the `-r` option; it allows you to import relationship definitions and not create the runtime schemas for any of them.
-r*relationshipName1[:relationshipName2]*	Loads the named relationship definition(s) into the repository without creating its runtime schema.
-s*serverName*	Specifies the name of the InterChange Server Express instance with which repos_copy should interface. The name is case-sensitive. If the server name is not specified, the tool prompts for a server name.
-summary	This option prints a list of components in the server repository (they are identified as "artifacts" rather than as components in the output). The output is in XML format. You can combine this option with the `-o` option to print the output to a file rather than the console.
-u*username*	Specifies the user name to log in to InterChange Server Express. If no user name is specified, repos_copy prompts for a user name.
-v	Prints the version number of the program that the repos_copy utility executes.
-vp	This option validates a package file. The server validates packages against the repository and makes sure that the dependencies among the components in the package are resolved. If the validation is not successful, repos_copy prints a list of the missing dependencies. This option does not make any changes to the repository; it just validates the package file. When using the `-vp` option you must also use the `-i` option to specify the package file to be validated.
-vr	This option validates the repository. The output message indicates whether the validation is successful or not. If the validation is not successful, repos_copy prints a list of the missing dependencies.

Table 11. Repos_copy command options (continued)

Option	Description
`-wi`	When this option is specified, repos_copy does not display any warnings that occur during the compilation of collaboration templates or maps. Only errors that occur during compilation are displayed. This allows the user to ignore warnings about deprecated methods, for instance.
`-xCompileAll`	Compiles all collaboration templates and maps in the repository.
`-xCompileAllCollabs`	Compiles all collaboration templates in the repository.
`-xCompileAllMaps`	Compiles all maps in the repository.
`-xCompileCollab:`*`collabTemplateName[+collabTemplateName]`*	Compiles the specified collaboration templates in the repository.
`-xCompileMap:`*`nativeMapName[+nativeMapName]`*	Compiles the specified maps in the repository.
`-xCompilePackage`	This option automatically compiles the package being deployed to the server. Since the production-mode server automatically compiles all packages, this option applies only to design-mode servers.

Table 12. Keywords for different entity types

Entity type	Keyword
Business object	`BusObj`
Collaboration object	`Collaboration`
Collaboration template	`CollabTemplate`
Database connection pool	`ConnectionPool`
Connector	`Connector`
Map	`Map`
Relationship	`Relationship`

Repos_copy usage scenarios

This section describes many of the common situations in which you will use repos_copy. It contains the following sections:

- “Printing the repos_copy command” on page 114
- “Validating a package” on page 114
- “Validating a package” on page 114
- “Deploying a package to the repository” on page 114
- “Validating the repository” on page 116
- “Deleting components from the repository” on page 117
- “Exporting components to a package” on page 118
- “Printing a list of components in the repository” on page 119

Printing the repos_copy command

You can run repos_copy without any arguments to have the command and its arguments printed out. The example below shows repos_copy when executed without any arguments, and the resulting output:

```
C:\>repos_copy
No Command line arguments to ReposCopy were specified
Usage: repos_copy {-o[outputFile] | -i[inputFile]}
    [-sserverName] [-uuserName] [-ppassword]
    [-ai] [-ar] [-arp] [-d] [-k] [-v]
    [-eentityType:entityName1[+entityType:entityName2] -deep]
    [-fentityFileName]
    [-rrelationshipName1[:relationshipName2] ]
    [-xCompileAll] [-xCompileAllCollabs] [-xCompileAllMaps]
    [-xCompileCollab:collabTemplateName[+collabTemplateName]]
    [-xCompileMap:nativeMapName[+nativeMapName]]
    [-xcompilepackage]
    [-mode]
    [-doentityType:entityName1[+entityType:entityName2] -deep]
    [-dfoentityType:entityName1[+entityType:entityName2] -deep]
    [-summary]
    [-vp]
    [-vr]
```

Validating a package

You can validate a package of components before deploying the package to a server. This is very useful because if you deploy a package to a production-mode server all the dependencies must be resolved or the deployment will fail. You cannot validate a user project or integration component library in System Manager to make sure that the dependencies are satisfied, so the only way to find out if a package is valid when deploying with System Manager is to attempt the deployment and use the error information when it fails to resolve the dependencies. If there are many components in the package, this can be a very time-consuming process.

Although you cannot validate an integration component library, you can export it to a package file and then validate the package file using repos_copy.

To validate a package file using repos_copy, use the `-i` option to specify the name of the package file to be validated and the `-vp` argument to validate it rather than deploy it.

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-iWebSphereICSServer.jar -vp
```

Repos_copy validates the contents of the package and displays a message to indicate whether or not the dependencies are resolved.

Deploying a package to the repository

The `-i` option allows you to deploy a package of components to the repository. If you do not specify the name of the package file then you are prompted to enter it.

The following example shows a a file named `WebSphereICSServer.jar` being deployed to a repository:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-iWebSphereICSServer.jar
```

Working with duplicate components during deployment

Commonly there will be components with the same name in the package file as there are in the repository. In this case you must decide whether or not you want to replace the components in the repository with those in the package file. The `-ai` option specifies that duplicate components should not be loaded into the repository:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-iCustomer.jar -ai
```

If you want to replace all the duplicate components in the repository, use the `-ar` option as in the following example:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-iCustomerSyncInterface.jar -ar
```

You can use the `-arp` option to interactively replace duplicate components in the repository. This lets you decide for each individual duplicate component whether it should be replaced or not.

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-iCustomerSyncInterface.jar -arp
```

Compiling and creating schemas

For maps and collaboration to execute at runtime, the maps and collaboration templates defined in the repository must be compiled. For relationships to function properly at runtime, their schemas must be created.

When you deploy components to a server running in production mode, all templates are automatically compiled and all relationship schemas are created. For the deployment to succeed, then, the code of the map and collaboration templates must be valid and InterChange Server Express must be able to communicate with the databases specified in the settings of the relationship definitions.

When you deploy components to a server running in design mode, the templates are not automatically compiled; relationship schemas are automatically created. There are options you can use to compile the templates, however, and there are options to not create relationship schemas.

The following example uses the `-xCompilePackage` option and does not use any form of the `-r` option. The result is that when the package specified by the `-i` option is deployed, the maps and collaboration templates are compiled and schemas are created for the relationships:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-iWebSphereICSServer.jar -xCompilePackage
```

You may not want relationship schemas created when you do a deployment. For instance, if you are deploying a package from one environment to another and did not change the properties of the relationships to use the database resources in the new environment then you will not want the schemas created until after you have changed the relevant properties. The following example uses the `-r*` option to not create schemas for all of the relationships in the package being deployed:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-iWebSphereICSServer.jar -xCompilePackage -r*
```

Note: You can use the `-r` option without the asterisk to specify the names of individual relationships whose schemas should not be created. For instance,

`-rCustomer:Order` would not create schemas for the `Customer` and `Order` relationships, but would still create schemas for any other relationships in the package being deployed.

Important: Although there are options to compile maps and collaboration templates after deployment, there is no way to either through repos_copy or System Manager to create the schema for a relationship other than during deployment. So, if you chose not to create the schema for a relationship during deployment because you needed to change the database settings, then you need to re-deploy the relationship afterwards and allow repos_copy to create the schema for the relationship.

Validating the repository

The repository must be in a valid state for a server instance to start in production mode. The reason for this is that ultimately the repository must be valid for the server to process flows successfully. Use the `-vr` option to validate a server repository, as in the example below:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull -vr
```

If the server is valid then repos_copy writes the following output to the console:

```
Validation Succeeded.All Dependencies Resolved.
```

If the repository is not valid then repos_copy prints a list of the dependencies that must be resolved.

Compiling components in the repository

If you deployed maps or collaboration templates to the repository and did not compile them during deployment, you can use repos_copy to compile them afterwards. This can be useful in situations where there are many components to deploy because deployment can take a long time and compiling can make the operation take even longer. Waiting until after the deployment has succeeded to do the compilation task can reduce the risk of spending an even greater amount of time migrating the environment if an error occurs.

The following example shows the use of the `-xCompileAll` option to compile all maps and collaboration templates in the the repository:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-xCompileAll
```

There are options to compile all of either type of component as well. Use `-xCompileAllCollabs` to compile all the collaboration templates, and `-xCompileAllMaps` to compile all the maps. The example below shows the use of `-xCompileAllMaps`:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-xCompileAllMaps
```

Just as you can compile all of one type of component, you can also compile an individual component. Use the `-xCompileCollab` or `-xCompileMap` option followed by a colon and the name of the collaboration template or map to compile a single component. The example below would compile a collaboration template named `CustomerSync`:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-xCompileCollab:CustomerSync
```

Deleting components from the repository

There are several options provided by repos_copy for deleting components in the repository. You can delete the entire repository, individual components, and individual components as well as any components that reference them.

Note: Components must be inactive for you to delete them. If you delete a single component then you must deactivate it first or the delete operation will fail. If you want to delete a component and all the components that reference it, you must deactivate not only the single component, but all those that reference it as well. You can delete the entire repository while the components are in an active state. Use System Monitor or web-based System Monitor to manage the states of components. System Monitor and web-based System Monitor are described in the *System Administration Guide*.

Deleting the entire repository

Use the `-d` option to delete all of the components in the repository. The following example shows the syntax:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin
-pnull -d
```

Repos_copy presents a prompt asking if you want to delete the entire repository or not.

Deleting components without referents

If a component does not have any referents—other components that reference it and require it to exist in order to perform their function in the system—then you can delete the individual component.

Use the `-do` option followed by the entity type, a colon, and the name of the component. The entity types are listed in Table 12 on page 113. The following example deletes the relationship named **Customer**:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin
-pnull -doRelationship:Customer
```

Deleting components with referents

If a component does have referents—other components that reference it and require it to exist in order to perform their function in the system—then you can only delete the component if the server is running in design-mode, and by using certain options.

Forcing a delete in spite of references: If a component has referents, repos_copy will not let you delete it with the `-do` option. You must use the `-dfo` option to force deletion of a component with referents. Forcing deletion of a component that has referents will leave the repository in an inconsistent state, and a server running in production mode does not permit that, so this option only works with a design-mode server. The following example shows the use of the `-dfo` option to delete the `Order` business object in spite of the fact that other components in the system (such as maps and relationships) have references to it:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-dfoBusObj:Order
```

Deleting the referents as well: Another way you can delete a component that has referents is to use the `-deep` option to delete the referents as well. This deletes the component and all of the components that have references to it. The following example shows the use of the `-deep` option when using the `-do` option to delete the `Customer` business object:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-doBusObj:Customer -deep
```

This option, unlike the `-dfo` option, is supported with servers running in production mode because the deletion of the referents along with the component guarantees that the repository remains valid. Keep in mind, however, that it can result in many components being deleted; you should be aware of the implications of this action prior to taking it.

Exporting components to a package

The `-o` option allows you to export components from the repository to a package. You must specify the name of the package file. When the `-o` option is used alone the entire repository is exported to a file, as in the following example:

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-oWebSphereICSServer.jar
```

You can specify individual components to be exported by using the `-e` option. You must use the -e option with the appropriate `EntityType` keyword listed in Table 12 on page 113, and must follow the keyword with the name of the component. You can specify multiple components by concatenating them with the plus (+) sign. In the following example, the `Customer` business object and `CustomerSync` collaboration template are exported to a package named `CustomerSyncInterface.jar`.

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-eBusObj:Customer+CollabTemplate:CustomerSync -oCustomerSyncInterface.jar
```

You can use the `-deep` option to export the dependencies of a component as well. In the previous example, the `Customer` business object was exported, but none of its child business objects were. The following example uses the `-deep` option to export the `CustomerSync_ClarifyToSAP` collaboration object and all of its dependencies.

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-eCollaboration:CustomerSync_ClarifyToSAP -oCustomerSyncInterface.jar -deep
```

If you want to export specific components, but do not want to have to enter the entity type keyword and component names, you can store them in a text file and use the `-f` option. This is very convenient when you want to frequently export the same components. The following example uses the `-f` option to load the components listed in a text file named `Components.txt` :

```
C:\WebSphereICS>repos_copy -sWebSphereICSServer -uadmin -pnull
-fComponents.txt -oCustomerSyncInterface.jar -deep
```

The contents of the file `Components.txt` are shown below; a paragraph return follows each entity type keyword and name combination:

```
BusObj:Customer
Relationship:Customer
CollabTemplate:CustomerSync
```

Note: Repos_copy and System Manager are unfortunately inconsistent with respect to what they identify as “dependencies”. If you attempt to delete a component using repos_copy but there are components that depend upon it then repos_copy lists those referring components as dependencies. However, if you right-click the component in System Manager and select **Show Dependencies** from the context menu the tool lists the components that the selected component depends on.

Printing a list of components in the repository

You can use the `-summary` argument when executing repos_copy to print a list of the components in the repository. The output is presented in XML format. Although it is not particularly useful to view at the command line, you can combine the `-summary` argument with the the `-o` argument to redirect the output to a file and then view the file in a browser or XML editor. The command usage in this case would be the following:

```
C:\>repos_copy -sWebSphereICSServer -uadmin -pnull -summary -oRepository.xml
```

Locale for repos_copy files

The repos_copy utility reads metadata from the repository and writes the data out to files in Unicode (UTF-8 format). It also reads such files and loads them into the repository in Unicode (UTF-8 or UCS-2, as the underlying repository database dictates).

Chapter 7. Configuring connectors

This chapter describes the procedures for configuring adapters and contains the following sections:
- "Using Connector Configurator"
- "Connector standard properties" on page 133

Using Connector Configurator

This appendix describes how to use Connector Configurator Express to set configuration property values for your adapter.

The topics covered in this appendix are:
- "Overview of Connector Configurator Express" on page 121
- "Starting Connector Configurator Express" on page 122
- "Creating a connector-specific property template" on page 122
- "Creating a new configuration file" on page 125
- "Setting the configuration file properties" on page 127
- "Using Connector Configurator Express in a globalized environment" on page 132

Overview of Connector Configurator Express

Connector Configurator Express allows you to configure the connector component of your adapter for use with InterChange Server Express.

You use Connector Configurator Express to:
- Create a **connector-specific property template** for configuring your connector.
- Create a **connector configuration file**; you must create one configuration file for each connector you install.
- Set properties in a configuration file.
  You may need to modify the default values that are set for properties in the connector templates. You must also designate supported business object definitions and maps for use with collaborations as well as specify messaging, logging and tracing, and data handler parameters, as required.

Connector configuration properties include both standard configuration properties (the properties that all connectors have) and connector-specific properties (properties that are needed by the connector for a specific application or technology).

Because **standard properties** are used by all connectors, you do not need to define those properties from scratch; Connector Configurator Express incorporates them into your configuration file as soon as you create the file. However, you do need to set the value of each standard property in Connector Configurator Express.

The range of standard properties may not be the same for all brokers and all configurations. Some properties are available only if other properties are given a specific value. The Standard Properties window in Connector Configurator Express will show the properties available for your particular configuration.

For **connector-specific properties**, however, you need first to define the properties and then set their values. You do this by creating a connector-specific property template for your particular adapter. There may already be a template set up in your system, in which case, you simply use that. If not, follow the steps in "Creating a new template" on page 123 to set up a new one.

Starting Connector Configurator Express

You can start and run Connector Configurator Express in either of two modes:

- Independently, in stand-alone mode
- From System Manager

Running Configurator Express in stand-alone mode

You can run Connector Configurator Express independently and work with connector configuration files, irrespective of your broker.

To do so:

- From **Start>Programs**, click **IBM WebSphere Business Integration Server Express> Toolset Express>Development>Connector Configurator Express**.
- Select **File>New>Configuration File**.

You may choose to run Connector Configurator Express independently to generate the file, and then connect to System Manager to save it in a System Manager project (see "Completing a configuration file" on page 126.)

Running Configurator Express from System Manager

You can run Connector Configurator Express from System Manager.

To run Connector Configurator Express:

1. Open the System Manager.
2. In the System Manager window, expand the **Integration Component Libraries** icon and highlight **Connectors**.
3. From the System Manager menu bar, click **Tools>Connector Configurator Express**. The Connector Configurator Express window opens and displays a **New Connector** dialog box.

To edit an existing configuration file:

1. In the System Manager window, select any of the configuration files listed in the Connector folder and right-click on it.
2. Click the Standard Properties tab to see which properties are included in this configuration file.

Creating a connector-specific property template

To create a configuration file for your connector, you need a connector-specific property template as well as the system-supplied standard properties.

You can create a brand-new template for the connector-specific properties of your connector, or you can use an existing file as the template.

- To create a new template, see "Creating a new template" on page 123.
- To use an existing file, simply modify an existing template and save it under the new name.

Creating a new template

This section describes how you create properties in the template, define general characteristics and values for those properties, and specify any dependencies between the properties. Then you save the template and use it as the base for creating a new connector configuration file.

To create a template:

1. Click **File>New>Connector-Specific Property Template**.
2. The **Connector-Specific Property Template** dialog box appears, with the following fields:
   - **Template**, and **Name**

     Enter a unique name that identifies the connector, or type of connector, for which this template will be used. You will see this name again when you open the dialog box for creating a new configuration file from a template.
   - **Old Template**, and **Select the Existing Template to Modify**

     The names of all currently available templates are displayed in the Template Name display.
   - To see the connector-specific property definitions in any template, select that template's name in the **Template Name** display. A list of the property definitions contained in that template will appear in the **Template Preview** display. You can use an existing template whose property definitions are similar to those required by your connector as a starting point for your template.
3. Select a template from the **Template Name** display, enter that template name in the **Find Name** field (or highlight your selection in **Template Name**), and click **Next**.

If you do not see any template that displays the connector-specific properties used by your connector, you will need to create one.

Specifying general characteristics: When you click **Next** to select a template, the **Properties - Connector-Specific Property Template** dialog box appears. The dialog box has tabs for General characteristics of the defined properties and for Value restrictions. The General display has the following fields:

- **General:**
  Property Type
  Updated Method
  Description
- **Flags**
  Standard flags
- **Custom Flag**
  Flag

After you have made selections for the general characteristics of the property, click the **Value** tab.

Specifying values: The **Value** tab enables you to set the maximum length, the maximum multiple values, a default value, or a value range for the property. It also allows editable values. To do so:

1. Click the **Value** tab. The display panel for Value replaces the display panel for General.
2. Select the name of the property in the **Edit properties** display.

3. In the fields for **Max Length** and **Max Multiple Values**, make any changes. The changes will not be accepted unless you also open the **Property Value** dialog box for the property, described in the next step.
4. Right-click the box in the top left-hand corner of the value table and click **Add**. A **Property Value** dialog box appears. Depending on the property type, the dialog box allows you to enter either a value, or both a value and range. Enter the appropriate value or range, and click **OK**.
5. The **Value** panel refreshes to display any changes you made in **Max Length** and **Max Multiple Values**. It displays a table with three columns:

   The **Value** column shows the value that you entered in the **Property Value** dialog box, and any previous values that you created.

   The **Default Value** column allows you to designate any of the values as the default.

   The **Value Range** shows the range that you entered in the **Property Value** dialog box.

   After a value has been created and appears in the grid, it can be edited from within the table display. To make a change in an existing value in the table, select an entire row by clicking on the row number. Then right-click in the **Value** field and click **Edit Value**.

Setting dependencies: When you have made your changes to the **General** and **Value** tabs, click **Next**. The **Dependencies - Connector-Specific Property Template** dialog box appears.

A dependent property is a property that is included in the template and used in the configuration file *only if* the value of another property meets a specific condition. For example, `PollQuantity` appears in the template only if JMS is the transport mechanism and `DuplicateEventElimination` is set to `True`.
To designate a property as dependent and to set the condition upon which it depends, do this:

1. In the **Available Properties** display, select the property that will be made dependent.
2. In the **Select Property** field, use the drop-down menu to select the property that will hold the conditional value.
3. In the **Condition Operator** field, select one of the following:

   == (equal to)

   != (not equal to)

   > (greater than)

   < (less than)

   >= (greater than or equal to)

   <=(less than or equal to)
4. In the **Conditional Value** field, enter the value that is required in order for the dependent property to be included in the template.
5. With the dependent property highlighted in the **Available Properties** display, click an arrow to move it to the **Dependent Property** display.
6. Click **Finish**. Connector Configurator Express stores the information you have entered as an XML document, under `\data\app` in the`\bin` directory where you have installed Connector Configurator Express.

Creating a new configuration file

You create a connector configuration file from a connector-specific template or by modifying an existing configuration file.

Creating a configuration file from a connector-specific template

Once a connector-specific template has been created, you can use it to create a configuration file:

1. Click **File>New>Connector Configuration**.
2. The **New Connector** dialog box appears, with the following fields:
   - **Name**

     Enter the name of the connector. Names are case-sensitive. The name you enter must be unique, and must be consistent with the file name for a connector that is installed on the system.

     Important: Connector Configurator Express does not check the spelling of the name that you enter. You must ensure that the name is correct.
   - **System Connectivity**

     The default broker is ICS. You cannot change this value.
   - **Select Connector-Specific Property Template**

     Type the name of the template that has been designed for your connector. The available templates are shown in the **Template Name** display. When you select a name in the Template Name display, the **Property Template Preview** display shows the connector-specific properties that have been defined in that template.

     Select the template you want to use and click **OK**.
3. A configuration screen appears for the connector that you are configuring. The title bar shows the integration broker and connector names. You can fill in all the field values to complete the definition now, or you can save the file and complete the fields later.
4. To save the file, click **File>Save>To File** or **File>Save>To Project**. To save to a project, System Manager must be running.
   If you save as a file, the **Save File Connector** dialog box appears. Choose `*.cfg` as the file type, verify in the File Name field that the name is spelled correctly and has the correct case, navigate to the directory where you want to locate the file, and click **Save**. The status display in the message panel of Connector Configurator Express indicates that the configuration file was successfully created.

   Important: The directory path and name that you establish here must match the connector configuration file path and name that you supply in the startup file for the connector.
5. To complete the connector definition, enter values in the fields for each of the tabs of the Connector Configurator Express window, as described later in this chapter.

Using an existing file

To use an existing file to configure a connector, you must open the file in Connector Configurator Express, revise the configuration, and then save the file as a configuration file (`*.cfg`).

You may have an existing file available in one or more of the following formats:

- A connector definition file.
  This is a text file that lists properties and applicable default values for a specific connector. Some connectors include such a file in a `\repository` directory in their delivery package (the file typically has the extension `.txt`; for example, CN_XML.txt for the XML connector).
- An InterChange Server Express repository file.
  Definitions used in a previous InterChange Server Express implementation of the connector may be available to you in a repository file that was used in the configuration of that connector. Such a file typically has the extension `.in` or `.out`.
- A previous configuration file for the connector.
  Such a file typically has the extension `*.cfg.`

Although any of these file sources may contain most or all of the connector-specific properties for your connector, the connector configuration file will not be complete until you have opened the file and set properties, as described later in this chapter.

To use an existing file to configure a connector, you must open the file in Connector Configurator Express, revise the configuration, and then resave the file.

Follow these steps to open a `*.txt,` `*.cfg` or `*.in` file from a directory:

1. In Connector Configurator Express, click **File>Open>From File**.
2. In the **Open File Connector** dialog box, select one of the following file types to see the available files:
   - Configuration (`*.cfg`)
   - InterChange Server Express Repository (`*.in`, `*.out`)

     Choose this option if a repository file was used to configure the connector. A repository file may include multiple connector definitions, all of which will appear when you open the file.
   - All files (*.*)

     Choose this option if a `*.txt` file was delivered in the adapter package for the connector, or if a definition file is available under another extension.
3. In the directory display, navigate to the appropriate connector definition file, select it, and click **Open**.

Follow these steps to open a connector configuration from a System Manager project:

1. Start System Manager. A configuration can be opened from or saved to System Manager only if System Manager has been started.
2. Start Connector Configurator Express.
3. Click **File>Open>From Project**.

Completing a configuration file

When you open a configuration file or a connector from a project, the Connector Configurator Express window displays the configuration screen, with the current attributes and values.

Connector Configurator Express requires values for properties described in the following sections:

- “Setting standard connector properties” on page 127
- “Setting application-specific configuration properties” on page 128

- "Specifying supported business object definitions" on page 128
- "Associated maps" on page 130
- "Setting trace/log file values" on page 131

Note: For connectors that use JMS messaging, an additional category may display, for special configuration of data handlers that convert the data to business objects. For further information, see "Data handlers" on page 131.

Setting the configuration file properties

When you create and name a new connector configuration file, or when you open an existing connector configuration file, Connector Configurator Express displays a configuration screen with tabs for the categories of required configuration values.

Standard properties differ from connector-specific properties as follows:

- Standard properties of a connector are shared by both the application-specific component of a connector and its broker component. All connectors have the same set of standard properties. These properties are described in Appendix A of each adapter guide. You can change some but not all of these values.
- Application-specific properties apply only to the application-specific component of a connector, that is, the component that interacts directly with the application. Each connector has application-specific properties that are unique to its application. Some of these properties provide default values and some do not; you can modify some of the default values. The installation and configuration chapters of each adapter guide describe the application-specific properties and the recommended values.

The fields for **Standard Properties** and **Connector-Specific Properties** are color-coded to show which are configurable:

- A field with a grey background indicates a standard property. You can change the value but cannot change the name or remove the property.
- A field with a white background indicates an application-specific property. These properties vary according to the specific needs of the application or connector. You can change the value and delete these properties.
- You can configure Value fields.
- The **Update Method** displayed for each property indicates whether a component or agent restart is necessary to activate changed values.

Setting standard connector properties

To change the value of a standard property:

1. Click in the field whose value you want to set.
2. Either enter a value, or select one from the drop-down menu if it appears.
3. After entering all the values for the standard properties, you can do one of the following:
   - To discard the changes, preserve the original values, and exit Connector Configurator Express, click **File>Exit** (or close the window), and click **No** when prompted to save changes.
   - To enter values for other categories in Connector Configurator Express, select the tab for the category. The values you enter for **Standard Properties** (or any other category) are retained when you move to the next category. When you close the window, you are prompted to either save or discard the values that you entered in all the categories as a whole.

- To save the revised values, click **File>Exit** (or close the window) and click **Yes** when prompted to save changes. Alternatively, click **Save>To File** from either the File menu or the toolbar.

Setting application-specific configuration properties

For application-specific configuration properties, you can add or change property names, configure values, delete a property, and encrypt a property. The default property length is 255 characters.

1. Right-click in the top left portion of the grid. A pop-up menu bar will appear. Click **Add** to add a property. To add a child property, right-click on the parent row number and click **Add child**.
2. Enter a value for the property or child property.
3. To encrypt a property, select the **Encrypt** box.
4. Choose to save or discard changes, as described for "Setting standard connector properties" on page 127.

The Update Method displayed for each property indicates whether a component or agent restart is necessary to activate changed values.

Important: Changing a preset application-specific connector property name may cause a connector to fail. Certain property names may be needed by the connector to connect to an application or to run properly.

Encryption for connector properties: Application-specific properties can be encrypted by selecting the **Encrypt** check box in the **Edit Property** window. To decrypt a value, click to clear the **Encrypt** check box, enter the correct value in the **Verification** dialog box, and click **OK**. If the entered value is correct, the value is decrypted and displays.

The adapter user guide for each connector contains a list and description of each property and its default value.

If a property has multiple values, the **Encrypt** check box will appear for the first value of the property. When you select **Encrypt**, all values of the property will be encrypted. To decrypt multiple values of a property, click to clear the **Encrypt** check box for the first value of the property, and then enter the new value in the **Verification** dialog box. If the input value is a match, all multiple values will decrypt.

Update method: Refer to the descriptions of update methods found in the *Standard configuration properties for connectors* appendix, under "Setting and updating property values" on page 133.

Connector properties are almost all static and the **Update Method** is `Component restart`. For changes to take effect, you must restart the connector after saving the revised connector configuration file.

Specifying supported business object definitions

Use the **Supported Business Objects** tab in Connector Configurator Express to specify the business objects that the connector will use. You must specify both generic business objects and application-specific business objects, and you must specify associations for the maps between the business objects.

For you to specify a supported business object, the business objects and their maps must exist in the system. Business object definitions, including those for data handler meta-objects, and map definitions should be saved into Integration Component Library (ICL) projects. For more information on ICL projects, see the *User Guide for WebSphere Business Integration Server Express*.

Note: Some connectors require that certain business objects be specified as supported in order to perform event notification or additional configuration (using meta-objects) with their applications. For more information, see the chapter on business objects in this guide as well as the *Business Object Development Guide*.

To specify that a business object definition is supported by the connector, or to change the support settings for an existing business object definition, click the **Supported Business Objects** tab and use the following fields.

Business object name

To designate that a business object definition is supported by the connector, with System Manager running:

1. Click an empty field in the **Business Object Name** list. A drop-down list displays, showing all the business object definitions that exist in the System Manager project.
2. Click on a business object to add it.
3. Set the **Agent Support** (described below) for the business object.
4. In the File menu of the Connector Configurator Express window, click **Save to Project**. The revised connector definition, including designated support for the added business object definition, is saved to the project in System Manager.

To delete a business object from the supported list:

1. To select a business object field, click the number to the left of the business object.
2. From the **Edit** menu of the Connector Configurator Express window, click **Delete Row**. The business object is removed from the list display.
3. From the **File** menu, click **Save to Project**.

Deleting a business object from the supported list changes the connector definition and makes the deleted business object unavailable for use in this implementation of this connector. It does not affect the connector code, nor does it remove the business object definition itself from System Manager.

Agent support

If a business object has Agent Support, the system will attempt to use that business object for delivering data to an application via the connector agent.

Typically, application-specific business objects for a connector are supported by that connector's agent, but generic business objects are not.

To indicate that the business object is supported by the connector agent, check the **Agent Support** box. The Connector Configurator Express window does not validate your Agent Support selections.

Maximum transaction level

The maximum transaction level for a connector is the highest transaction level that the connector supports.

For most connectors, `Best Effort` is the only possible choice.

You must restart the server for changes in transaction level to take effect.

Associated maps

Each connector supports a list of business object definitions and their associated maps that are currently active in InterChange Server Express. This list appears when you select the **Associated Maps** tab.

The list of business objects contains the application-specific business object which the agent supports and the corresponding generic object that the controller sends to the subscribing collaboration. The association of a map determines which map will be used to transform the application-specific business object to the generic business object or the generic business object to the application-specific business object.

If you are using maps that are uniquely defined for specific source and destination business objects, the maps will already be associated with their appropriate business objects when you open the display, and you will not need (or be able) to change them.

If more than one map is available for use by a supported business object, you will need to explicitly bind the business object with the map that it should use.

The **Associated Maps** tab displays the following fields:

- **Business Object Name**

  These are the business objects supported by this connector, as designated in the **Supported Business Objects** tab. If you designate additional business objects under the Supported Business Objects tab, they will be reflected in this list after you save the changes by choosing **Save to Project** from the **File** menu of the Connector Configurator Express window.
- **Associated Maps**

  The display shows all the maps that have been installed to the system for use with the supported business objects of the connector. The source business object for each map is shown to the left of the map name, in the **Business Object Name** display.
- **Explicit**

  In some cases, you may need to explicitly bind an associated map.

  Explicit binding is required only when more than one map exists for a particular supported business object. When InterChange Server Express boots, it tries to automatically bind a map to each supported business object for each connector. If more than one map takes as its input the same business object, the server attempts to locate and bind one map that is the superset of the others.

  If there is no map that is the superset of the others, the server will not be able to bind the business object to a single map, and you will need to set the binding explicitly.

  To explicitly bind a map:

  1. In the **Explicit** column, place a check in the check box for the map you want to bind.
  2. Select the map that you intend to associate with the business object.
  3. In the **File** menu of the Connector Configurator Express window, click **Save to Project**.
  4. Deploy the project to InterChange Server Express.
  5. Reboot the server for the changes to take effect.

Resources

The **Resource** tab allows you to set a value that determines whether and to what extent the connector agent will handle multiple processes concurrently, using connector agent parallelism.

Not all connectors support this feature. If you are running a connector agent that was designed in Java to be multi-threaded, you are advised not to use this feature, since it is usually more efficient to use multiple threads than multiple processes.

Setting trace/log file values

When you open a connector configuration file or a connector definition file, Connector Configurator Express uses the logging and tracing values of that file as default values. You can change those values in Connector Configurator Express.

To change the logging and tracing values:

1. Click the **Trace/Log Files** tab.
2. For either logging or tracing, you can choose to write messages to one or both of the following:
   - To console (STDOUT):
     Writes logging or tracing messages to the STDOUT display.

     Note: You can only use the STDOUT option from the **Trace/Log Files** tab for connectors running on the Windows platform.
   - To File:
     Writes logging or tracing messages to a file that you specify. To specify the file, click the directory button (ellipsis), navigate to the preferred location, provide a file name, and click **Save**. Logging or tracing message are written to the file and location that you specify.

     Note: Both logging and tracing files are simple text files. You can use the file extension that you prefer when you set their file names. For tracing files, however, it is advisable to use the extension `.trace` rather than `.trc`, to avoid confusion with other files that might reside on the system. For logging files, `.log` and `.txt` are typical file extensions.

Data handlers

The data handlers section is available for configuration only if you have designated a value of JMS for DeliveryTransport and a value of JMS for ContainerManagedEvents. Adapters that make use of the guaranteed event delivery enable this tab.

See the descriptions under `ContainerManagedEvents` in the Standard Properties appendix for values to use for these properties.

Saving your configuration file

After you have created the configuration file and set its properties, you need to deploy it to the correct location for your connector. Save the configuration in an ICL project, and use System Manager to load the file into InterChange Server Express.

The file is saved as an XML document. You can save the XML document in three ways:

- From System Manager, as a file with a `*.con` extension in an Integration Component Library, or

- In a directory that you specify.
- In stand-alone mode, as a file with a `*.cfg` extension in a directory folder.

For details about using projects in System Manager, and for further information about deployment, see the *User Guide for IBM WebSphere Business Integration Server Express.*

Completing the configuration

After you have created a configuration file for a connector and modified it, make sure that the connector can locate the configuration file when the connector starts up.

To do so, open the startup file used for the connector, and verify that the location and file name used for the connector configuration file match exactly the name you have given the file and the directory or path where you have placed it.

Using Connector Configurator Express in a globalized environment

Connector Configurator Express is globalized and can handle character conversion between the configuration file and the integration broker. Connector Configurator Express uses native encoding. When it writes to the configuration file, it uses UTF-8 encoding.

Connector Configurator Express supports non-English characters in:

- All value fields
- Log file and trace file path (specified in the **Trace/Log files** tab)

The drop list for the `CharacterEncoding` and `Locale` standard configuration properties displays only a subset of supported values. To add other values to the drop list, you must manually modify the `\Data\Std\stdConnProps.xml` file in the product directory.

For example, to add the locale `en_GB` to the list of values for the `Locale` property, open the `stdConnProps.xml` file and add the line in boldface type below:

```
<Property name="Locale"
isRequired="true"
updateMethod="component restart">
                <ValidType>String</ValidType>
            <ValidValues>
                                <Value>ja_JP</Value>
                                <Value>ko_KR</Value>
                                <Value>zh_CN</Value>
                                <Value>zh_TW</Value>
                                <Value>fr_FR</Value>
                                <Value>de_DE</Value>
                                <Value>it_IT</Value>
                                <Value>es_ES</Value>
                                <Value>pt_BR</Value>
                                <Value>en_US</Value>
                                <Value>en_GB</Value>

                    <DefaultValue>en_US</DefaultValue>
            </ValidValues>
    </Property>
```

Connector standard properties

This appendix describes the standard configuration properties for the connector component of the adapters in WebSphere Business Integration Server Express.

Not every connector makes use of all these standard properties. When you select an integration broker from Connector Configurator Express, you will see a list of the standard properties that you need to configure for your adapter.

For information about properties specific to the connector, see the relevant adapter user guide.

Configuring standard connector properties

Adapter connectors have two types of configuration properties:

- Standard configuration properties
- Connector-specific configuration properties

This section describes the standard configuration properties. For information on configuration properties specific to a connector, see its adapter user guide.

Using Connector Configurator Express

You configure connector properties from Connector Configurator Express, which you access from System Manager. For more information on using Connector Configurator Express, refer to the Connector Configurator Express appendix.

Setting and updating property values

The default length of a property field is 255 characters.

The connector uses the following order to determine a property's value (where the highest number overrides other values):

1. Default
2. Repository
3. Local configuration file
4. Command line

A connector obtains its configuration values at startup. If you change the value of one or more connector properties during a run-time session, the property's **Update Method** determines how the change takes effect. There are four different update methods for standard connector properties:

- **Dynamic**
  The change takes effect immediately after it is saved in System Manager.
- **Component restart**
  The change takes effect only after the connector is stopped and then restarted in System Manager. You do not need to stop and restart the application-specific component or the integration broker.
- **Server restart**
  The change takes effect only after you stop and restart the application-specific component and the integration broker.
- **Agent restart**
  The change takes effect only after you stop and restart the application-specific component.

To determine how a specific property is updated, refer to the **Update Method** column in the Connector Configurator Express window, or see the Update Method column in the Property Summary table below.

Summary of standard properties

Table 13 provides a quick reference to the standard connector configuration properties. Not all the connectors make use of all these properties, and property settings may differ from integration broker to integration broker, as standard property dependencies are based on `RepositoryDirectory`.

You must set the values of some of these properties before running the connector. See the following section for an explanation of each property.

Table 13. Summary of standard configuration properties

Property name	Possible values	Default value	Update method	Notes
AdminInQueue	Valid JMS queue name	*CONNECTORNAME* /ADMININQUEUE	Component restart	Delivery Transport is JMS
AdminOutQueue	Valid JMS queue name	*CONNECTORNAME*/ADMINOUTQUEUE	Component restart	Delivery Transport is JMS
AgentConnections	1-4	1	Component restart	Delivery Transport is IDL
AgentTraceLevel	0-5	0	Dynamic	
ApplicationName	Application name	Value specified for the connector application name	Component restart	
BrokerType	ICS	ICS		
CharacterEncoding	ascii7, ascii8, SJIS, Cp949, GBK, Big5, Cp297, Cp273, Cp280, Cp284, Cp037, Cp437 **Note:** This is a subset of supported values.	ascii7	Component restart	
ConcurrentEventTriggeredFlows	1 to 32,767	1	Component restart	Repository directory is <REMOTE>
ContainerManagedEvents	No value or JMS	No value	Component restart	Delivery Transport is JMS
ControllerStoreAndForwardMode	true or false	truetrue	Dynamic	Repository directory is <REMOTE>
ControllerTraceLevel	0-5	0	Dynamic	Repository directory is <REMOTE>
DeliveryQueue		*CONNECTORNAME*/DELIVERYQUEUE	Component restart	JMS transport only
DeliveryTransport	IDL or JMS	IDL	Component restart	

Table 13. Summary of standard configuration properties (continued)

Property name	Possible values	Default value	Update method	Notes
DuplicateEventElimination	`true or false`	`false`	Component restart	JMS transport only: Container Managed Events must be <NONE>
EnableOidForFlowMonitoring	`true or false`	`false`	Component restart	
FaultQueue		*`CONNECTORNAME`*`/FAULTQUEUE`	Component restart	JMS transport only
jms.FactoryClassName	`CxCommon.Messaging.jms .IBMMQSeriesFactory` or any Java class name	`CxCommon.Messaging. jms.IBMMQSeriesFactory`	Component restart	JMS transport only
jms.MessageBrokerName	`crossworlds.queue. manager`	`crossworlds.queue.manager`	Component restart	JMS transport only
jms.NumConcurrentRequests	Positive integer	`10`	Component restart	JMS transport only
jms.Password	Any valid password		Component restart	JMS transport only
jms.UserName	Any valid name		Component restart	JMS transport only
JvmMaxHeapSize	Heap size in megabytes	`128m`	Component restart	Repository directory is <REMOTE>
JvmMaxNativeStackSize	Size of stack in kilobytes	`128k`	Component restart	Repository directory is <REMOTE>
JvmMinHeapSize	Heap size in megabytes	`1m`	Component restart	Repository directory is <REMOTE>
Locale	`en_US`, `ja_JP`, `ko_KR`, `zh_CN`, `zh_TW`, `fr_FR`, `de_DE`, `it_IT`, `es_ES`, `pt_BR` **Note:** This is a subset of the supported locales.	`en_US`	Component restart	
LogAtInterchangeEnd	`true` or `false`	`false`	Component restart	
MaxEventCapacity	`1-2147483647`	`2147483647`	Dynamic	Repository Directory is <REMOTE>
MessageFileName	Path or filename	`InterchangeSystem.txt`	Component restart	
MonitorQueue	Any valid queue name	*`CONNECTORNAME`*`/MONITORQUEUE`	Component restart	JMS transport only: DuplicateEvent Elimination must be true
OADAutoRestartAgent	`true` or `false`	`false`	Dynamic	Repository Directory is <REMOTE>

Table 13. Summary of standard configuration properties (continued)

Property name	Possible values	Default value	Update method	Notes
OADMaxNumRetry	A positive number	1000	Dynamic	Repository Directory is <REMOTE>
OADRetryTimeInterval	A positive number in minutes	10	Dynamic	Repository Directory is <REMOTE>
PollEndTime	HH:MM (HH is 0-23, MM is 0-59)	HH:MM	Component restart	
PollFrequency	A positive integer in milliseconds no (to disable polling) key (to poll only when the letter p is entered in the connector's Command Prompt window)	10000	Dynamic	
PollQuantity	1-500	1	Agent restart	JMS transport only: Container Managed Events is specified
PollStartTime	HH:MM(HH is 0-23, MM is 0-59)	HH:MM	Component restart	
RepositoryDirectory	Location of metadata repository		Agent restart	Set to <REMOTE>
RequestQueue	Valid JMS queue name	*CONNECTORNAME*/REQUESTQUEUE	Component restart	Delivery Transport is JMS
ResponseQueue	Valid JMS queue name	*CONNECTORNAME*/RESPONSEQUEUE	Component restart	Delivery Transport is JMS:
RestartRetryCount	0-99	3	Dynamic	
RestartRetryInterval	A sensible positive value in minutes: 1 - 2147483547	1	Dynamic	
SourceQueue	Valid JMS queue name	*CONNECTORNAME*/SOURCEQUEUE	Agent restart	Only if Delivery Transport is JMS and Container Managed Events is specified
SynchronousRequestQueue	Valid JMS queue name	*CONNECTORNAME*/ SYNCHRONOUSREQUESTQUEUE	Component restart	Delivery Transport is JMS
SynchronousRequestTimeout	0 - any number (millisecs)	0	Component restart	Delivery Transport is JMS
SynchronousResponseQueue	Valid JMS queue name	*CONNECTORNAME*/ SYNCHRONOUSRESPONSEQUEUE	Component restart	Delivery Transport is JMS
WireFormat	CwBO	CwBO	Agent restart	

Standard configuration properties

This section lists and defines each of the standard connector configuration properties.

AdminInQueue

The queue that is used by the integration broker to send administrative messages to the connector.

The default value is `CONNECTORNAME/ADMININQUEUE`.

AdminOutQueue

The queue that is used by the connector to send administrative messages to the integration broker.

The default value is `CONNECTORNAME/ADMINOUTQUEUE`.

AgentConnections

The AgentConnections property controls the number of ORB connections opened by `orb.init[]`.

By default, the value of this property is set to 1. There is no need to change this default.

AgentTraceLevel

Level of trace messages for the application-specific component. The default is `0`. The connector delivers all trace messages applicable at the tracing level set or lower.

ApplicationName

Name that uniquely identifies the connector's application. This name is used by the system administrator to monitor the WebSphere business integration system environment. This property must have a value before you can run the connector.

BrokerType

Identifies the integration broker that you are using, which is ICS.

CharacterEncoding

Specifies the character code set used to map from a character (such as a letter of the alphabet, a numeric representation, or a punctuation mark) to a numeric value.

Note: Java-based connectors do not use this property. A C++ connector currently uses the value `ascii7` for this property.

By default, a subset of supported character encodings only is displayed in the drop list. To add other supported values to the drop list, you must manually modify the `\Data\Std\stdConnProps.xml` file in the product directory. For more information, see the appendix on using Connector Configurator Express in this guide.

ConcurrentEventTriggeredFlows

Determines how many business objects can be concurrently processed by the connector for event delivery. Set the value of this attribute to the number of business objects you want concurrently mapped and delivered. For example, set the value of this property to `5` to cause five business objects to be concurrently processed. The default value is 1.

Setting this property to a value greater than 1 allows a connector for a source application to map multiple event business objects at the same time and deliver them to multiple collaboration instances simultaneously. This speeds delivery of business objects to the integration broker, particularly if the business objects use complex maps. Increasing the arrival rate of business objects to collaborations can improve overall performance in the system.

To implement concurrent processing for an entire flow (from a source application to a destination application), you must:

- Configure the collaboration to use multiple threads by setting its `Maximum number of concurrent events` property high enough to use multiple threads.
- Ensure that the destination application's application-specific component can process requests concurrently. That is, it must be multi-threaded, or be able to use connector agent parallelism and be configured for multiple processes. Set the `Parallel Process Degree` configuration property to a value greater than `1`.

The `ConcurrentEventTriggeredFlows` property has no effect on connector polling, which is single-threaded and performed serially.

ContainerManagedEvents

This property allows a JMS-enabled connector with a JMS event store to provide guaranteed event delivery, in which an event is removed from the source queue and placed on the destination queue as a single JMS transaction.

This property only appears if the `DeliveryTransport` property is set to the value `JMS`.

The default value is `No value`.

When ContainerManagedEvents is set to `JMS`, you must configure the following properties to enable guaranteed event delivery:

- PollQuantity = `1` to `500`
- SourceQueue = `CONNECTORNAME/SOURCEQUEUE`

You must also configure a data handler with the MimeType, DHClass, and DataHandlerConfigMOName (optional) properties. To set those values, use the **Data Handler** tab in Connector Configurator Express. The fields for the values under the Data Handler tab will be displayed only if you have set `ContainerManagedEvents` to `JMS`.

Note: When ContainerManagedEvents is set to `JMS`, the connector does *not* call its `pollForEvents()` method, thereby disabling that method's functionality.

ControllerStoreAndForwardMode

Sets the behavior of the connector controller after it detects that the destination application-specific component is unavailable.

If this property is set to `true` and the destination application-specific component is unavailable when an event reaches InterChange Server Express, the connector controller blocks the request to the application-specific component. When the application-specific component becomes operational, the controller forwards the request to it.

However, if the destination application's application-specific component becomes unavailable **after** the connector controller forwards a service call request to it, the connector controller fails the request.

If this property is set to `false`, the connector controller begins failing all service call requests as soon as it detects that the destination application-specific component is unavailable.

The default is `true`.

ControllerTraceLevel

Level of trace messages for the connector controller. The default is `0`.

DeliveryQueue

Applicable only if `DeliveryTransport` is JMS.

The queue that is used by the connector to send business objects to InterChange Server Express.

The default value is `CONNECTORNAME/DELIVERYQUEUE`.

DeliveryTransport

Specifies the transport mechanism for the delivery of events. Possible values are `IDL` for CORBA IIOP or `JMS` for Java Messaging Service. The default is IDL.

The connector sends service call requests and administrative messages over CORBA IIOP if the value configured for the `DeliveryTransport` property is `IDL`.

JMS: Enables communication between the connector and client connector framework using Java Messaging Service (JMS).

If you select JMS as the delivery transport, additional JMS properties such as `jms.MessageBrokerName`, `jms.FactoryClassName`, `jms.Password`, and `jms.UserName`, appear in Connector Configurator Express. The first two of these properties are required for this transport.

Important: There may be a memory limitation if you use the JMS transport mechanism for a connector running on InterChange Server Express.

In this environment, you may experience difficulty starting both the connector controller (on the server side) and the connector (on the client side) due to memory use within the WebSphere MQ client.

DuplicateEventElimination

When you set this property to `true`, a JMS-enabled connector can ensure that duplicate events are not delivered to the delivery queue. To use this feature, the connector must have a unique event identifier set as the business object's **ObjectEventId** attribute in the application-specific code. This is done during connector development.

This property can also be set to `false`.

Note: When `DuplicateEventElimination` is set to `true`, you must also configure the `MonitorQueue` property to enable guaranteed event delivery.

EnableOidForFlowMonitoring

When you set this property to `true`, the adapter framework will mark the incoming **ObjectEventId** as a foreign key for the purpose of flow monitoring.

The default is `false`.

FaultQueue

If the connector experiences an error while processing a message then the connector moves the message to the queue specified in this property, along with a status indicator and a description of the problem.

The default value is `CONNECTORNAME/FAULTQUEUE`.

JvmMaxHeapSize

The maximum heap size for the agent (in megabytes).

The default value is 128m.

JvmMaxNativeStackSize

The maximum native stack size for the agent (in kilobytes).

The default value is 128k.

JvmMinHeapSize

The minimum heap size for the agent (in megabytes).

The default value is 1m.

jms.FactoryClassName

Specifies the class name to instantiate for a JMS provider. You *must* set this connector property when you choose `JMS` as your delivery transport mechanism (DeliveryTransport).

The default is `CxCommon.Messaging.jms.IBMMQSeriesFactory`.

jms.MessageBrokerName

Specifies the broker name to use for the JMS provider. You *must* set this connector property when you choose `JMS` as your delivery transport mechanism (see DeliveryTransport).

The default is `crossworlds.queue.manager`.

jms.NumConcurrentRequests

Specifies the maximum number of concurrent service call requests that can be sent to a connector at the same time. Once that maximum is reached, new service calls block and wait for another request to complete before proceeding.

The default value is 10.

jms.Password

Specifies the password for the JMS provider. A value for this property is optional.

There is no default.

jms.UserName

Specifies the user name for the JMS provider. A value for this property is optional.

There is no default.

Locale

Specifies the language code, country or territory, and, optionally, the associated character code set. The value of this property determines such cultural conventions as collation and sort order of data, date and time formats, and the symbols used in monetary specifications.

A locale name has the following format:

```
ll_TT.codeset
```

where:

ll	a two-character language code (usually in lower case)
TT	a two-letter country or territory code (usually in upper case)
codeset	the name of the associated character code set; this portion of the name is often optional.

By default, only a subset of supported locales appears in the drop list. To add other supported values to the drop list, you must manually modify the `\Data\Std\stdConnProps.xml` file in the product directory. For more information, see the appendix on using Connector Configurator Express in this guide.

The default value is `en_US`. If the connector has not been globalized, the only valid value for this property is `en_US`.

LogAtInterchangeEnd

Specifies whether to log errors to the integration broker's log destination. Logging to the broker's log destination also turns on e-mail notification, which generates e-mail messages for the `MESSAGE_RECIPIENT` specified in the `InterchangeSystem.cfg` file when errors or fatal errors occur.

For example, when a connector loses its connection to its application, if `LogAtInterChangeEnd` is set to `true`, an e-mail message is sent to the specified message recipient. The default is `false`.

MaxEventCapacity

The maximum number of events in the controller buffer. This property is used by flow control.

The value can be a positive integer between 1 and 2147483647. The default value is 2147483647.

MessageFileName

The name of the connector message file. The standard location for the message file is `\connectors\messages`. Specify the message filename in an absolute path if the message file is not located in the standard location.

If a connector message file does not exist, the connector uses `InterchangeSystem.txt` as the message file. This file is located in the product directory.

Note: To determine whether a specific connector has its own message file, see the individual adapter user guide.

MonitorQueue

The logical queue that the connector uses to monitor duplicate events. It is used only if the `DeliveryTransport` property value is JMS and `DuplicateEventElimination` is set to `TRUE`.

The default value is `CONNECTORNAME/MONITORQUEUE`

OADAutoRestartAgent

Specifies whether the connector uses the automatic and remote restart feature. This feature uses the MQ-triggered Object Activation Daemon (OAD) to restart the connector after an abnormal shutdown, or to start a remote connector from System Monitor.

This property must be set to `true` to enable the automatic and remote restart feature. For information on how to configure the MQ-triggered OAD feature, see the *Installation Guide for Windows*.

The default value is `false`.

OADMaxNumRetry

Specifies the maximum number of times that the MQ-triggered OAD automatically attempts to restart the connector after an abnormal shutdown. The `OADAutoRestartAgent` property must be set to `true` for this property to take effect.

The default value is `1000`.

OADRetryTimeInterval

Specifies the number of minutes in the retry-time interval for the MQ-triggered OAD. If the connector agent does not restart within this retry-time interval, the connector controller asks the OAD to restart the connector agent again. The OAD repeats this retry process as many times as specified by the `OADMaxNumRetry` property. The `OADAutoRestartAgent` property must be set to `true` for this property to take effect.

The default is `10`.

PollEndTime

Time to stop polling the event queue. The format is `HH:MM`, where *`HH`* represents 0-23 hours, and *`MM`* represents 0-59 seconds.

You must provide a valid value for this property. The default value is `HH:MM`, but must be changed.

PollFrequency

The amount of time between polling actions. Set `PollFrequency` to one of the following values:

- The number of milliseconds between polling actions.
- The word `key`, which causes the connector to poll only when you type the letter `p` in the connector's Command Prompt window. Enter the word in lowercase.

- The word no, which causes the connector not to poll. Enter the word in lowercase.

The default is 10000.

Important: Some connectors have restrictions on the use of this property. To determine whether a specific connector does, see the installing and configuring chapter of its adapter guide.

PollQuantity

Designates the number of items from the application that the connector should poll for. If the adapter has a connector-specific property for setting the poll quantity, the value set in the connector-specific property will override the standard property value.

PollStartTime

The time to start polling the event queue. The format is *HH:MM*, where *HH* represents 0-23 hours, and *MM* represents 0-59 seconds.

You must provide a valid value for this property. The default value is HH:MM, but must be changed.

RequestQueue

The queue that is used by InterChange Server Express to send business objects to the connector.

The default value is CONNECTOR/REQUESTQUEUE.

RepositoryDirectory

The location of the repository from which the connector reads the XML schema documents that store the meta-data for business object definitions.

This value must be set to <REMOTE> because the connector obtains this information from the InterChange Server Express repository.

ResponseQueue

Applicable only if DeliveryTransport is JMS.

Designates the JMS response queue, which delivers a response message from the connector framework to the integration broker. InterChange Server Express sends the request and waits for a response message in the JMS response queue.

RestartRetryCount

Specifies the number of times the connector attempts to restart itself. When used for a parallel connector, specifies the number of times the master connector application-specific component attempts to restart the slave connector application-specific component.

The default is 3.

RestartRetryInterval

Specifies the interval in minutes at which the connector attempts to restart itself. When used for a parallel connector, specifies the interval at which the master connector application-specific component attempts to restart the slave connector application-specific component. Possible values ranges from 1 to 2147483647.

The default is 1.

SourceQueue

Applicable only if `DeliveryTransport` is JMS and ContainerManagedEvents is specified.

Designates the JMS source queue for the connector framework in support of guaranteed event delivery for JMS-enabled connectors that use a JMS event store. For further information, see "ContainerManagedEvents" on page 138.

The default value is `CONNECTOR/SOURCEQUEUE`.

SynchronousRequestQueue

Applicable only if `DeliveryTransport` is JMS.

Delivers request messages that require a synchronous response from the connector framework to the broker. This queue is necessary only if the connector uses synchronous execution. With synchronous execution, the connector framework sends a message to the `SynchronousRequestQueue` and waits for a response back from the broker on the `SynchronousResponseQueue`. The response message sent to the connector bears a correlation ID that matches the ID of the original message.

The default is `CONNECTORNAME/SYNCHRONOUSREQUESTQUEUE`

SynchronousResponseQueue

Applicable only if `DeliveryTransport` is JMS.

Delivers response messages sent in reply to a synchronous request from the broker to the connector framework. This queue is necessary only if the connector uses synchronous execution.

The default is `CONNECTORNAME/SYNCHRONOUSRESPONSEQUEUE`

SynchronousRequestTimeout

Applicable only if `DeliveryTransport` is JMS.

Specifies the time in minutes that the connector waits for a response to a synchronous request. If the response is not received within the specified time, then the connector moves the original synchronous request message into the fault queue along with an error message.

The default value is 0.

WireFormat

This is the message format on the transport. The setting is`CwBO`.

Chapter 8. Configuring database connection pools

At run-time, it takes time for processes to establish new connections to the database. You can minimize that time by establishing database connection pools in advance for use by your collaboration and map processes. Database connection pools can improve performance and enable direct database access from within a collaboration.

A database connection pool can be used by multiple collaborations and maps, and each collaboration or map can use multiple database connection pools.

This chapter contains the following sections:

When to use database connection pools

This section describes some of the situations in which you might want to use database connection pools.

Performing routing

The logic of a business process may dictate that a business object must be routed to different destination applications depending on the value in one or more fields in the business object.

For instance, a site might store and process customer entities in different applications depending on the value in an attribute such as `CustomerType`. The collaboration template would need to retrieve and evaluate the value in that attribute and make a decision as to which destination application to send the business object to based on the value. Although this can be accomplished with control flow structures in Java, the pairings between the values and destination applications are then hard-coded in the collaboration template. If they need to be changed because of a change in procedure, or added to because a new value and application are introduced into the interface, then the collaboration template must be modified and re-compiled, it must be re-deployed, and so forth. A much more flexible implementation stores the pairings of values and destination applications in a database table. To implement this sort of approach, do the following:

1. Create a database table which has one column to store the routing values and another column to store some information that associates the appropriate destination application with the routing value. Table 14 is an example of such a table.

Table 14. Routing table example

Routing value	**Destination application value**
Customer	AppA

Table 14. Routing table example (continued)

Routing value	Destination application value
Federal	AppB
Reseller	AppC
Academic	AppD

2. Create a database connection pool and database connection in System Manager as described in “Creating database connection pools and database connections” on page 148.
3. In the collaboration template design the logic to do the following:
   a. Retrieve the value in the attribute to be used for routing and store the value in a variable.
   b. Obtain a connection to the database.
   c. Execute a SQL query that retrieves the value in the column that stores the destination application values where the value in the column that stores routing values is equal to the value stored in the variable in the collaboration template.
   d. Use a decision node to cause the collaboration logic to branch depending on the value returned from the database table. The different branches should lead to different service call nodes responsible for sending the business object out to the appropriate connector and thereby sending it out to the appropriate destination application.

Performing lookups in database tables

You might need to translate one value into another by looking up its equivalent in a table. Frequently you perform these operations by implementing lookup relationships, though using a lookup relationship does not always make sense. Lookup relationships are designed primarily for situations where each application involved in an interface needs its own way of representing a piece of data—a participant is created for each application and the lookup relationship itself connects all the participants much in the same way that the integration broker connects the applications. Sometimes you have a need to transform a value into one of several other values, but there is not the need to maintain a separate representation of that data for each application involved in the interface. In such a case you should create a table in the repository to store the associated values and then use a database connection and SQL select statements to retrieve the desired value.

Furthermore, the API provided for lookup relationships makes it very easy to abstract related data across applications, but does not make it easy to perform more complicated queries. The lookup relationship API is designed to take a piece of data and return the key value that the data shares with the other pieces of application data in the relationship, or to take a key value and return a particular piece of data associated with it. The lookup relationship API cannot return multiple column values, however, or execute stored procedures, which the APIs of the `CwDBConnection` class are able to do.

You can also satisfy this requirement with Java code rather than with a database query, by using control structures such as “if/else” and “switch/case” statements. Consider the following advantages and disadvantages of the different approaches and make the appropriate choice based on the situation:

- It is easier to add new records (such as a new routing value paired with a new destination application) when using a database connection because you only have to add a new row to the database; no components have to be modified, and the new pairing is effectively immediate. Adding new associations in Java code, however, is less convenient because you must modify the code of the component, recompile it, and possibly stop and restart it (if the component is a collaboration). You should pause the components of the interface beforehand as well, to ensure that there no transactions are in progress while the component is being compiled, or they may fail.
- Performing a lookup in Java code should be faster than making a connection to a database and performing a query, though this depends on how many associations there are that the Java control structure must iterate through.
- It is less convenient to migrate an interface that uses a database connection to another environment than it is to migrate an interface that exclusively relies on Java code, because you must remember to migrate not only the map or collaboration that makes the connection, but must also remember to recreate the tables involved in the new environment.

Persisting information

Some customers want to make information about the operation of the business integration system persistent by storing that information in a database so that it can be referenced for problem resolution or historical analysis.

To satisfy this requirement, do the following:

1. Create a database table with as many columns as necessary to store the desired data.
2. Create a database connection pool and database connection in System Manager as described in "Creating database connection pools and database connections" on page 148.
3. In the collaboration template design the logic to do obtain a connection to the database and execute a SQL query that inserts the desired data into the appropriate column.

Typically this requirement involves persisting information contained in the business object being processed by the collaboration (such as the primary key of the entity being processed), or information about the system itself (such as the successful processing of a business object request). You can persist business object data like primary key values by using the Business Object Probe feature; for more information, see the *Collaboration Development Guide*.

When not to use database connection pools

You should not use database connection pools to connect directly to the database of an application. To interact with an application database you should only use an adapter. Database connections should only be made to databases that do not support an application because adapters use the business logic provided by programming interfaces of the application. If you connect directly to an application database to perform a SQL update statement, for example, then you circumvent any related logic that the API would have performed in response to an update operation. This violates the integrity of the application and the business process.

If you have a need to retrieve information from an application, but do not want to use an adapter because of the impact on performance of sending a business object request to the application and receiving the response, you have several alternatives:

- You can design a custom business object that only has attributes for the fields you need to affect in the operation. For instance, you may have a large application-specific business object designed for an application entity involved in a particular business process. If you need to retrieve just a small subset of information about that entity as part of another business process, you might be concerned about the performance impact of retrieving the full business object for just those few fields. To minimize the impact, you can make a copy of the application-specific business object and reduce its structure to only those fields that are required. That way only a small amount of information must be processed.

  This approach is best for retrieving information in the application database that changes frequently, because each request will retrieve information that is current at the time of the request.
- You can replicate the application information on the same server that hosts the InterChange Server Express database. You can use the utilities provided by the database vendor to script the structure and data of a table into a file and then reproduce the table and its data in a database other than the one that the application itself uses.

  This approach is most useful for small amounts of information that is flat and static, such as a lookup table. You should not, however, use this approach for large entities that span multiple tables because the queries involved would be difficult to create and maintain, whereas using business objects to represent the entity would be easy for you to develop and for others to maintain. It is also not very good for tables that are very dynamic and have new records added frequently, because you either must manually update the replicated table frequently, or risk the interface having outdated information.

Creating database connection pools and database connections

A database connection pool consists of a number of reserved database connections. The reserved database connections are made available only to the collaboration and map processes that you design to use the pool.

To create a database connection pool, you define the values necessary for making a database connection beforehand. The integration system saves this database connection information and uses it at runtime to more quickly establish connections for collaboration and mapping processes that you have assigned.

The configuration of database connection values that you define can be used by one or more pools. For each pool you will specify a number of connections; these connections will be allocated, used, and freed back to the pool.

Note: It is recommended that you connect to the InterChange Server Express instance that will use the database connection pool in System Manager before you create the pool, so that you can validate the connection.

Do the following to create a database pool:

1. Right-click the **Database Connection Pools** folder in your integration component library in System Manager and choose **Create New Database Connection** from the context menu.

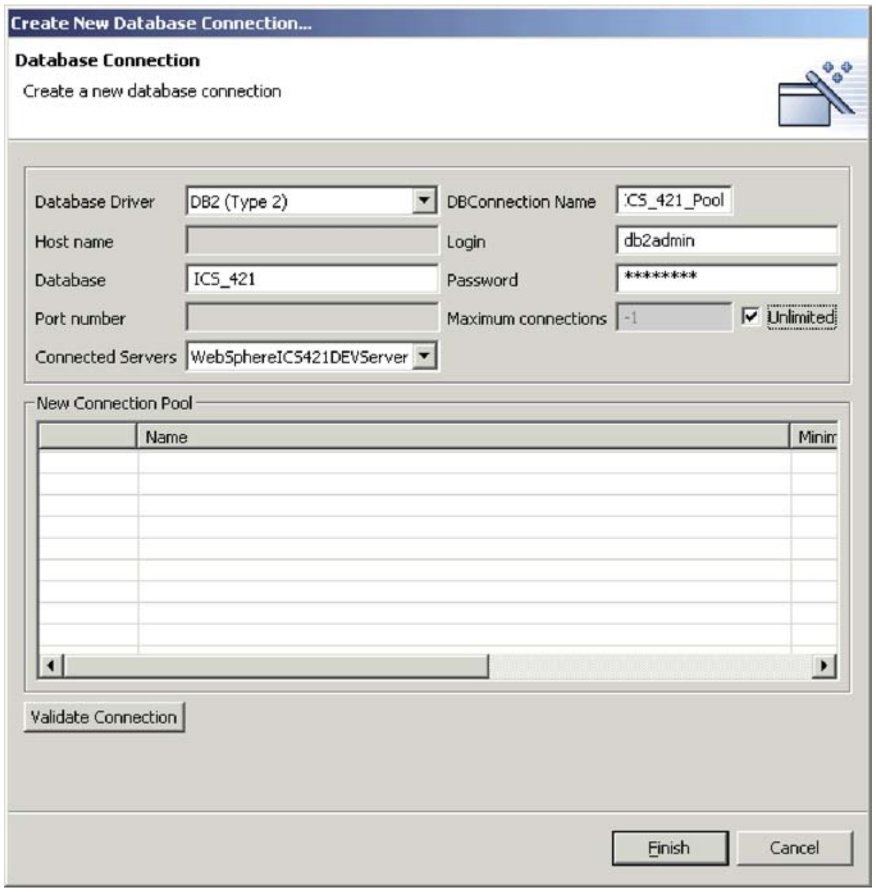
The “Database Connection” dialog appears.

Note: Although the name of the dialog would suggest that what you are creating at this point is a database connection, it is really a database connection pool. In later steps you will define database connections within the pool.

2. Select the appropriate value in the **Database Driver** drop-down menu—either `DB2 (Type 2)`,or `MQ SQL Server(Type 4)`—depending on the database vendor.
3. If you chose `MQ SQL Server(Type 4)`in the **Database Driver** drop-down menu, then you must type the name of the computer on which the database server resides in the **Host name** field.
4. Type the name of the database in the **Database** field.
5. If you chose `MQ SQL Server(Type 4)`in the **Database Driver** drop-down menu, then you must type the port number through which clients communicate with the database server in the **Port number** field.
6. If you are connected to the InterChange Server Express instance that will use the pool, select that instance from the **Connected Servers** drop-down menu.
7. Type a name for the pool in the **DBConnection Name** field. You specify this database connection pool name when writing Java code to establish the connection in maps or collaboration templates.
8. Type the user name that should be used by InterChange Server Express to log in to the specified database in the **Login** field.
9. Type the password for the user name.
10. Type the maximum number of connections that should be established by the pool for all the individual database connection objects you plan to create within it in the **Maximum connections** field, or enable the **Unlimited** checkbox to allow as many connections to be established as are permitted by the database server configuration and licensing.

    Warning: Be careful when working in this field. It is not a single-line text field, even though it should be, so you can accidentally press **Enter**. You will not see the value you entered then, and will justifiably try to re-type the value. Then when you try to finish creating the database connection pool you receive an error that a valid value must be entered for the field. Do not press **Enter** in this field.

    Figure 34 on page 150 shows the “New Database Connection Pool” wizard.



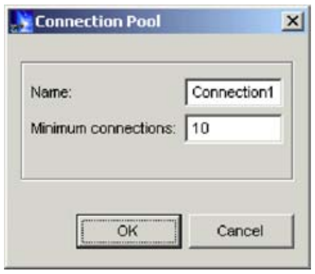
Database Driver	DB2 (Type 2)	DBConnection Name	ICS_421_Pool
Host name		Login	db2admin
Database	ICS_421	Password	********
Port number		Maximum connections	-1 ☑ Unlimited
Connected Servers	WebSphereICS421DEVServer		

Figure 34. Create New Database Connection wizard

11. To create a new database connection object in the pool, right-click a row in the **New Connection Pool** table at the bottom of the wizard and choose **New Connection Pool** from the context menu.

    Note: Although the interface would suggest that what you are creating at this point is also a database connection pool, it is really a database connection object that you are creating within the pool you just created.

    The "Connection Pool" dialog appears.
12. Type a name for the database connection object in the **Name** field.
13. In the **Minimum connections** field, enter the minimum number of connections that the database connection object should establish. Note that the sum of the minimums for all the pools that you create in pool cannot exceed the maximum number that you specify for the database connection pool object itself. System Manager does not allow you to attempt to add a connection object or modify an existing connection object so that the maximum number of connections for the pool would be exceeded. Figure 35 on page 151 shows the "Connection Pool" dialog.



Name: Connection1

Minimum connections: 10

Figure 35. “Connection Pool” dialog

14. Click **Finish**.

    System Manager saves the database connection pool object and an icon appears under the **Database Connection Pools** folder in the integration component library.
15. Use the wizard to create another database connection pool at this time or click **Cancel** to close the wizard.

Validating database connection pools

After creating a database connection pool you should validate it to make sure it can be used by maps and collaborations at runtime.

Do the following to validate a database connection pool:

1. Start the InterChange Server Express instance.
2. Connect System Manager to the InterChange Server Express instance.
3. Do one of the following:
   - Right-click either the database connection pool object in an integration component library or a shortcut to it in a user project, then select **Validate Connection** from the context menu, then select the InterChange Server Express instance name from the submenu.
   - Right-click either the database connection pool object in the server repository in the InterChange Server Express Component Management view, then select **Validate Connection** from the context menu.
4. If the database connection pool is valid then System Manager displays a prompt indicating that the validation was successful.

   If System Manager displays an error that validation was not successful, resolve the problem by verifying the availability of the database, the privileges of the user account specified, and the accuracy of the configuration information.

Modifying database connection pools

You can modify some properties of both database connection pools and database connection objects.

To modify a database connection pool, do one of the following:

- Right-click either the database connection pool component in an integration component library, or the shortcut to it a user project and select **Properties** from the context menu.
- Select the database connection pool component in an integration component library, or the shortcut to it a user project and do one of the following:
  – Select **Component > Properties** from the menu bar.

  – Press **Alt+Enter**.

The following sections document the properties of database connection pools and database connection objects that you can change:

- “Modifying login information”
- “Modifying the number of connections”

Modifying login information

Many companies follow a practice of changing passwords at a regular interval to guard against old accounts being used to violate security. If this is the case at the site then you may need to change the login information for the database connection pool.

To change the login name used to access the database resource, type the new value into the Login field in the “Database Connection” screen.

To change the password for the specified login, click **Change** in the “Database Connection” screen, then do the following at the “Password Change” prompt:

1. Type the current password in the **Old Password** field.
2. Type the new password in the **New Password** field.
3. Type the new password again in the **Confirm Password** field.

Modifying the number of connections

You can modify the maximum number of connections for a database connection pool, and the minimum number of connections for a database connection object. The sum of all the minimum connections of all the database connection objects defined in a database connection pool may not exceed the maximum number of connections specified for the pool itself.

Modifying the maximum connections for a database connection pool

Do the following to modify the maximum number of connections for a database connection pool:

1. Access the properties of the database connection pool you want to edit, as described in “Modifying database connection pools” on page 151.
2. Do one of the following to modify the maximum number of connections that can be established by the total number of connection objects in the pool:
   - Type the maximum number of connections that should be established by the pool in the **Maximum connections** field.
   - Enable the **Unlimited** checkbox to allow as many connections to be established as are permitted by the database server configuration and licensing.
3. Click **OK**.

Modifying the minimum connections for a database connection object

Do the following to modify the maximum number of connections for a connection object:

1. Access the properties of the database connection pool you want to edit, as described in “Modifying database connection pools” on page 151.

2. In the “New Connection Pool” pane, right-click the connection object for which you want to change the minimum number of connections and select **Edit** from the context menu.
3. Type the new value in the **Minimum connections** field.
4. Click **OK**.

Deleting database connection objects

Do the following to delete a database connection object:

1. Access the properties of the database connection pool as described in “Modifying database connection pools” on page 151.
2. In the “New Connection Pool” pane, right-click the connection object you want to delete and select **Delete** from the context menu.
3. Click **OK**.

Using database connection pools in collaborations and maps

To use database connection pools in maps and collaboration templates, you use the `getDBConnection()` method in the `BaseCollaboration` class:

```
CwDBConnection getDBConnection(String ConnectionPoolName);
```

The string *`ConnectionPoolName`* must match exactly a pool name that you have created or will create in System Manager. If the name does not match exactly, or if you set connection pool name values in a collaboration template or map but do not create the database connection pool in System Manager, the collaboration object or map will fail.

For collaborations, you can set pool names in a template either before or after you have created the collaboration objects from that template; you do not need to reconfigure new collaboration objects to match a new connection pool name.

For more information on developing collaboration templates to use database connection pools, see the *Collaboration Development Guide*.

For more information on developing maps to use database connection pools, see the *Map Development Guide*.

Configuring transaction bracketing

You can configure a collaboration or a map to use database connection pools in either Implicit Database Transaction mode (in which the transaction begins as soon as the connection is made and ends when the process is finished) or Explicit Database Transaction mode (in which you begin and end the transaction programmatically).

To configure transaction bracketing for collaboration objects, see “Implicit database transaction” on page 168.

To configure transaction bracketing for maps, see the *Map Development Guide*.

Chapter 9. Configuring collaboration objects

This chapter describes configuration of a collaboration object and contains the following sections:
- "Collaboration objects and groups"
- "Creating a collaboration object" on page 156
- "Modifying collaboration objects" on page 158

This chapter focuses on the task of configuing a collaboration object, but contains some references to the development tasks of designing and creating collaboration templates. For information about designing and creating a collaboration template (tasks that require the WebSphere Business Integration Server Express Plus version of this product), and for information about using Process Designer Express, see the *Collaboration Development Guide*.

Collaboration objects and groups

A collaboration object is an instance of a collaboration template. To configure a collaboration, you must:
1. Create a new collaboration object from a collaboration template. For more information, see "Creating a collaboration object" on page 156.
2. Bind the ports of the collaboration object, either internally (to a connector or another collaboration) or externally (to an external process that calls the collaboration, such as an e-business web application servlet). For more information, see "Binding collaboration object ports" on page 161.
3. Configure the general properties of the collaboration object. For more information, see "Configuring collaboration object general properties" on page 164.
4. Configure the collaboration-specific properties for the collaboration object. For more information, see "Configuring collaboration-specific properties" on page 169.

When you create an internal binding, you can bind only to connectors or collaborations that support the business object expected by the collaboration port. This allows communication between the bound components, allowing the collaboration to send and receive business objects, as well as receive responses to requests. A collaboration cannot run until you bind all its ports.

You can create multiple collaboration objects from a single collaboration template. For example, two collaborations from the same template can implement the same logic across two sets of applications at your site.

Collaboration-object groups

A collaboration-object group is a set of two or more collaboration objects that are bound to one another.

When you configure a collaboration object's port to receive an incoming business object from another collaboration object or to send an outgoing business object to another collaboration object, the combination of collaboration objects is a collaboration-object group. Any number of collaboration objects can be bound into a group.

The behavior of collaboration objects that are members of a collaboration-object group is different from the behavior of those that are not members of a group. When you apply a command such as start, pause, or stop to a collaboration object that is a member of a group, the command affects all members of the group.

All members of a collaboration-object group must support the same transaction level in order for one member of the group to execute transactionally. If you bind a collaboration object that has transactional level `None` to a collaboration object that has transaction level `Best Effort`, both run at level `None`.

Creating a collaboration object

This section describes how to create a new collaboration object using the System Manager wizard. It describes the general flow of the wizard and references other sections in this chapter to address the specifics of collaboration object configuration. Do the following to create a collaboration object using the wizard:

1. In System Manager, right-click on the **Collaboration Objects** folder in an integration component library and choose **Create New Collaboration Object** from the context menu.

   The "Create New Collaboration" wizard appears, listing the installed templates in the **Template Name** column and a description (if available) in the **Description** column. The dialog is the first screen in a wizard that steps you through the initial creation and configuration of a collaboration object. You can later modify the configuration values.
2. Select the collaboration template upon which the object should be based from the list of templates. The name you select is displayed in the **Selected Collaboration template** field below the list.

   You can also type text in the **Find** field and the dialog will select a collaboration template whose name begins with the characters you type in the field. You can then use the up and down arrows to navigate among templates that share the pattern of characters, if there are several that match.
3. Type a name for the collaboration object in the **Collaboration object name** field.

   Note: Do not give the collaboration object exactly the same name as the collaboration template from which it is derived. That is, if the collaboration template name is *SalesOrderProcessing*, the name of the collaboration object cannot be simply *SalesOrderProcessing*. If the names are completely identical, you will not be able to access the collaboration object using repos_copy with the -e (Entity) option because repos_copy searches for the collaboration template before searching for a collaboration object. A typical approach to naming the collaboration object is to include the name of the template along with the names of the source and destination applications that the collaboration object integrates, as in the following syntax:
   `CollabTemplateName_SourceAppName_to_DestAppName`

   The name cannot exceed 80 characters in length.
4. Click **Next** to advance the wizard to the Bind ports screen.

The “Bind ports” screen displays the ports defined in the collaboration template, the business object definition supported by each port, the type of component to which the port might be bound, and the particular components of the specified type to which each port can be bound.

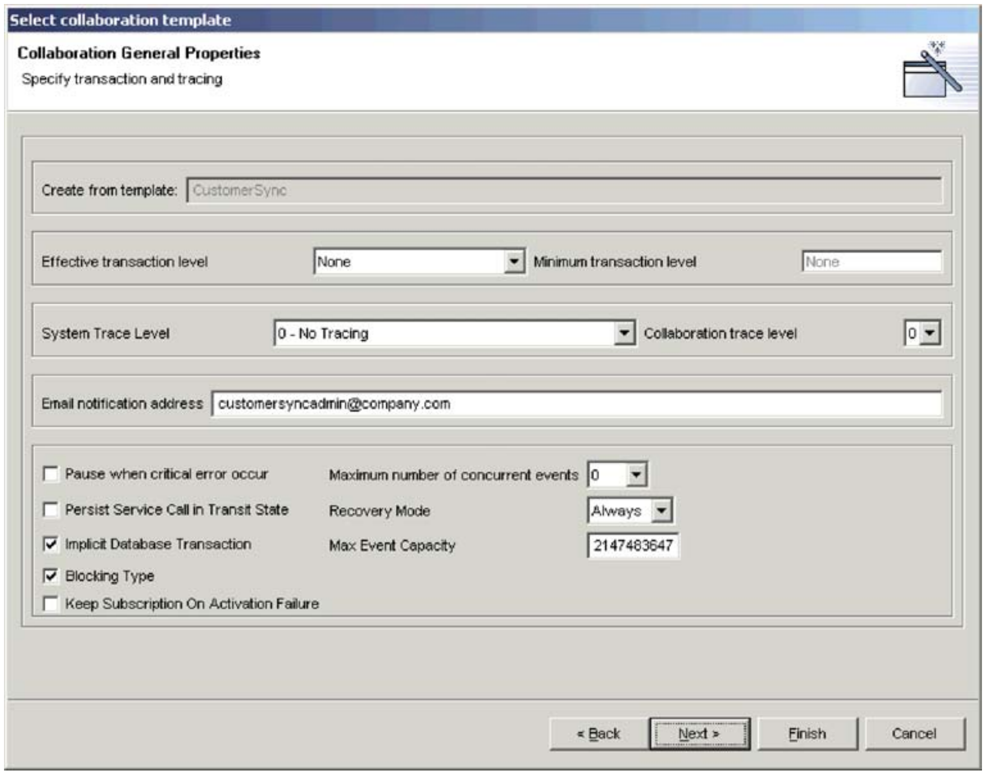
Depending on the value in the **Type** column, the **Bind With** column displays either the connectors or the collaborations that support the business object type in the **Business Object Definition** field. For each port displayed, choose either `Collaboration`, `Connector` or `Web Service` from the **Type** column and then choose the desired component in the **Bind With** column.

To configure a port so that it can receive business object requests from an external programmatic entity (such as a servlet), you must follow the instructions in “Configuring external port bindings” on page 162.

You can bind some, all, or none of the ports at this screen of the wizard. If you do not bind all the ports at this point you can always modify the port bindings after finishing the wizard. This way, if a particular component that you have to bind to a port does not support the necessary business object definition you can finish the collaboration object wizard, add support for the business object definition to the required component, and then reconfigure the collaboration object.

For more information about binding ports, and information about how to modify port bindings, see “Binding collaboration object ports” on page 161.

5. Click **Next** to advance the wizard to the “Collaboration General Properties” screen. This screen allows you to configure the properties that belong to all collaboration templates defined in the WebSphere Business Integration Server Express business integration system. Figure 36 shows the “Collaboration General Properties” screen:


Select collaboration template

Collaboration General Properties

Specify transaction and tracing

Create from template: CustomerSync

Effective transaction level: None

Minimum transaction level: None

System Trace Level: 0 - No Tracing

Collaboration trace level: 0

Email notification address: customersyncadmin@company.com

- [ ] Pause when critical error occur
- [ ] Persist Service Call in Transit State
- [x] Implicit Database Transaction
- [x] Blocking Type
- [ ] Keep Subscription On Activation Failure

Maximum number of concurrent events: 0

Recovery Mode: Always

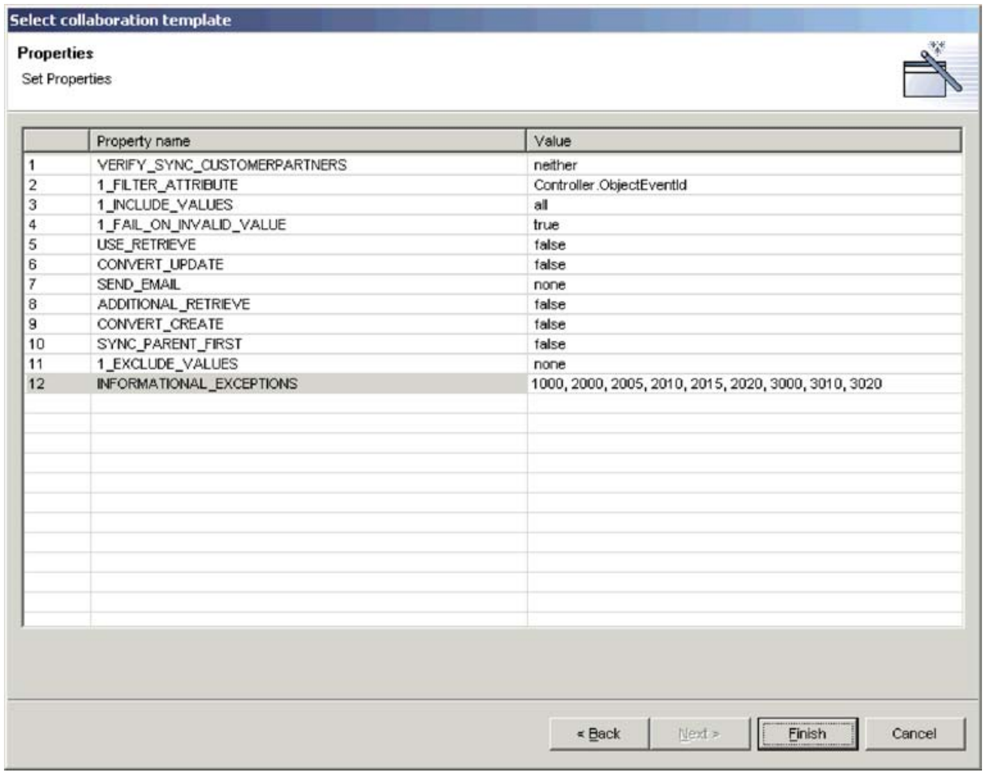
Max Event Capacity: 2147483647

Figure 36. Configuring collaboration object general properties

Set each property to the desired value.

For more information about collaboration general properties and how to modify them for an existing collaboration object, see “Configuring collaboration object general properties” on page 164.

6. Click **Next** to advance the wizard to the Properties screen. This screen allows you to configure the properties that are specific to the particular collaboration template.



	Property name	Value
1	VERIFY_SYNC_CUSTOMERPARTNERS	neither
2	1_FILTER_ATTRIBUTE	Controller.ObjectEventId
3	1_INCLUDE_VALUES	all
4	1_FAIL_ON_INVALID_VALUE	true
5	USE_RETRIEVE	false
6	CONVERT_UPDATE	false
7	SEND_EMAIL	none
8	ADDITIONAL_RETRIEVE	false
9	CONVERT_CREATE	false
10	SYNC_PARENT_FIRST	false
11	1_EXCLUDE_VALUES	none
12	INFORMATIONAL_EXCEPTIONS	1000, 2000, 2005, 2010, 2015, 2020, 3000, 3010, 3020

Figure 37. Configuring collaboration template-specific properties

Enter a value for each property or accept the default value. For information about each property, refer to the documentation for the particular collaboration template.

Click **Finish** to complete the collaboration object creation wizard. The new collaboration object is displayed in the Graphical View tab. For more information about the collaboration object graphical view and tree view, see “Collaboration object views” on page 159.

Modifying collaboration objects

This section describes how to modify a collaboration object. You might have to modify the port bindings or properties of a collaboration object for the following reasons:

- You created the collaboration object using the wizard as described in “Creating a collaboration object” on page 156, but had to bind a port to an external programmatic entity, which cannot be done through the wizard
- You created the collaboration object using the wizard as described in “Creating a collaboration object” on page 156, but the necessary components did not support the business object definitions associated with the ports, so you had to modify the component definitions

- You imported the collaboration object definition from another integration component library or InterChange Server Express and you need to make environment-specific changes
- You need to modify a collaboration object that has been running successfully in a production environment, but requires an administrative modification, such as changing the e-mail alias to which e-mail notifications are sent in the event of an error

To modify a collaboration object, double-click the collaboration object in the **Collaboration Objects** folder either in the Integration Component Libraries folder or in the **User Projects** folder in System Manager.

Collaboration object views

There are two view representations for collaboration objects: graphical view and tree view. The following sections describe them and their benefits.

Collaboration object graphical view

The collaboration object graphical view is the view shown by default after creating a new collaboration object. As shown in Figure 38 on page 160, the graphical view represents the collaboration object with an icon in the center of the view and with icons representing each port in the collaboration radiating outward. This view is useful to demonstrate the “flow” of the business process: you can drag-and-drop ports around the view pane to position them in places that suggest the directionality of the collaboration communication. For instance, you can position the port that receives the triggering business object on the left-hand side of the pane and position the port that sends the business object to the destination application on the right-hand side of the pane.

Note: There is no way for the system to be able to automatically display the ports that receive triggering business objects on one side and the ports that send business objects to destination applications on another side. The developer of a collaboration template determines the naming and function of its ports and there is no way for the system to know the role a developer had in mind for a particular port. You should read the documentation for a collaboration template and then reposition the ports in the graphical view of any collaboration objects based on the template in a way that best facilitates your use of the tools.

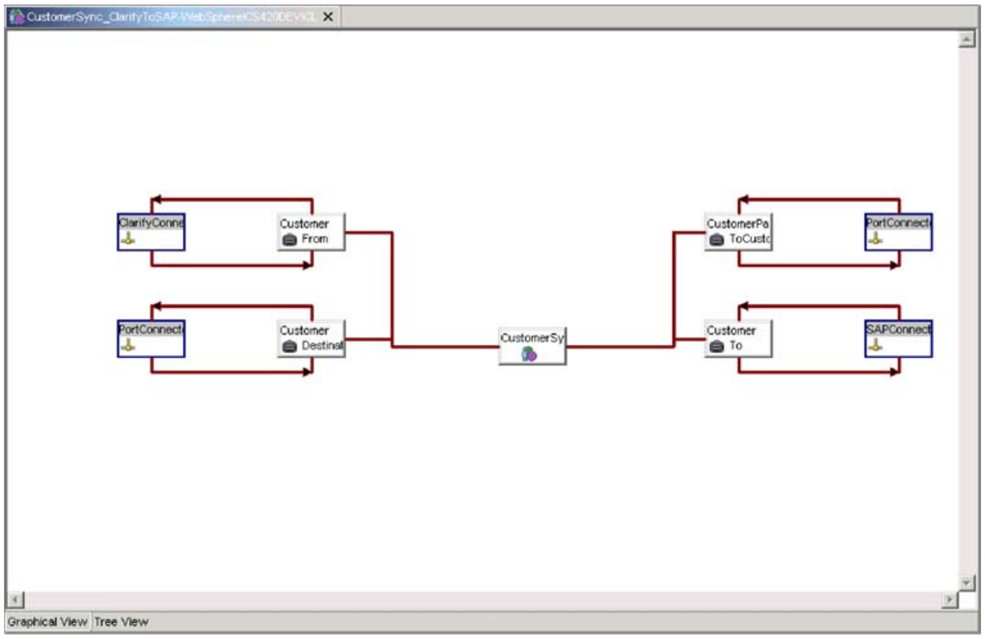


Figure 38. Collaboration object graphical view

Collaboration object tree view

You can switch to the collaboration object tree view from the graphical view by clicking the Tree View tab. As shown in Figure 39 on page 161, the tree view represents the collaboration object with an icon at the top of a hierarchical tree and represents the ports with icons branching underneath it. Although this view does not suggest the “flow” of the business process very well, it presents the ports and their bound components in a very orderly manner, making it easier to find a particular port that you might need to reconfigure. This can be particularly helpful with collaboration objects based on templates that have many ports, which can appear very confusing in the graphical view. You cannot rearrange the icons in the collaboration object tree view.

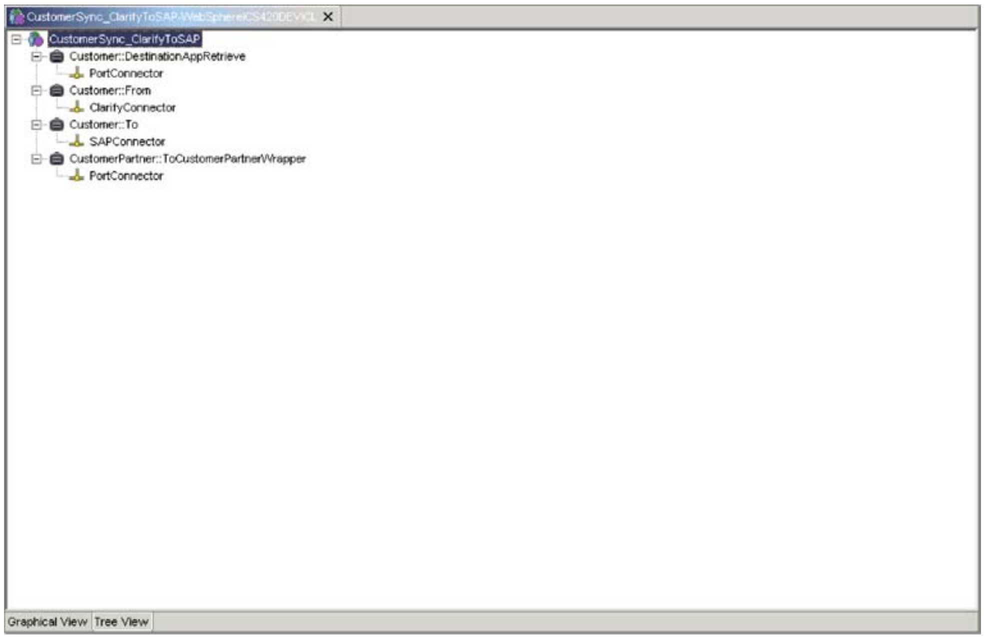

CustomerSync_ClarifyToSAP-WebSphereICS420DEVICL
CustomerSync_ClarifyToSAP
Customer::DestinationAppRetrieve
PortConnector
Customer::From
ClarifyConnector
Customer::To
SAPConnector
CustomerPartner::ToCustomerPartnerWrapper
PortConnector
Graphical View
Tree View

Figure 39. Collaboration object tree view

Binding collaboration object ports

Collaboration ports are the interfaces through which collaborations send and receive business objects. You bind collaboration object ports to different components to configure them to exchange the type of business object supported by the port.

You can bind collaboration object ports when creating the collaboration object initially by using the wizard as described in "Creating a collaboration object" on page 156. You can also edit the port bindings after creating the collaboration object. To edit the port bindings in the graphical view, either double-click the icon for a port or right-click the icon for a port and choose **Bind Port** from the context menu. To edit the port bindings in the tree view, right-click the icon for a port and choose **Bind Port** from the context menu.

To bind a collaboration port to an internal component such as a connector or the port of another collaboration object, leave the radio button labeled **Internal** in the Type pane enabled and see "Configuring internal port bindings."

To bind a collaboration port to an external programmatic entity such as a web servlet, enable the radio button labeled **External** in the Type pane and see "Configuring external port bindings" on page 162.

Configuring internal port bindings

To bind a collaboration port to a connector or to the port of another collaboration object, do the following:

1. Enable the radio button labeled **Internal** in the Type pane of the port configuration dialog.
2. Select the appropriate component type, either `Connector` or `Collaboration` in the Bind With pane.
3. Select the specific component from the list.

4. Click **OK**.

Figure 40 shows the port configuration dialog with an internal binding type.

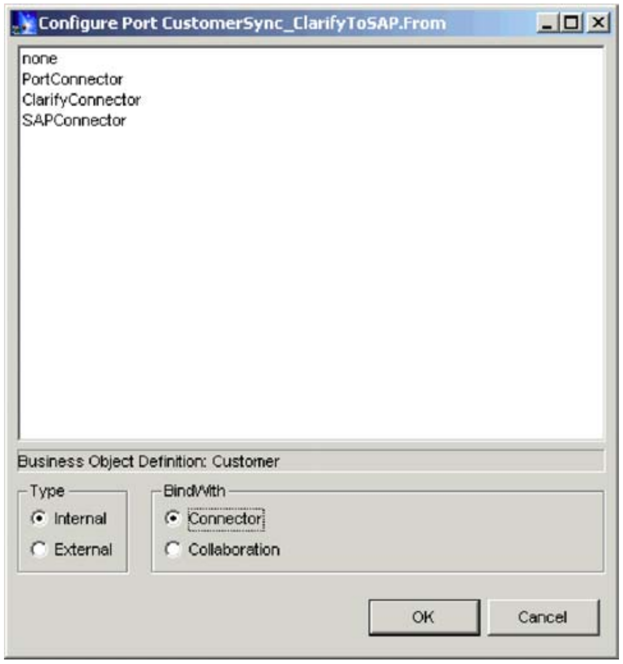


Figure 40. Configuring internal collaboration object ports

Only the components that support the type of business object definition shown in the **Business Object Definition** field are listed in the dialog. If you do not see a component you expect to see listed then it is either not of the specified type (and you should change the **Bind With** radio button), or the component does not support the business object definition. If the **Bind With** radio button is set to the proper component type and the component you expect to see is not shown, then modify the definition for the particular component to add support for the business object definition and then launch the port configuration dialog afterwards.

You must bind all of the ports of a collaboration object to start it, so a collaboration object cannot run until you have bound all of its ports.

Some collaboration templates have ports defined that support an optional course of business logic. For instance, many collaboration templates are designed so that they can attempt to retrieve the entity that might have just been created in the destination application to ensure that the operation was successful. This sort of behavior is frequently optional and configurable through the collaboration-specific properties of the collaboration. All of the ports of a collaboration object must be bound to a component for it to start, as mentioned earlier, so even if you do not intend to take advantage of the optional behavior you must configure ports that might exist only to support it. In this case, just add support for the business object definition to an unused component (such as the `PortConnector`) and be sure to configure the collaboration properties to not use the optional functionality.

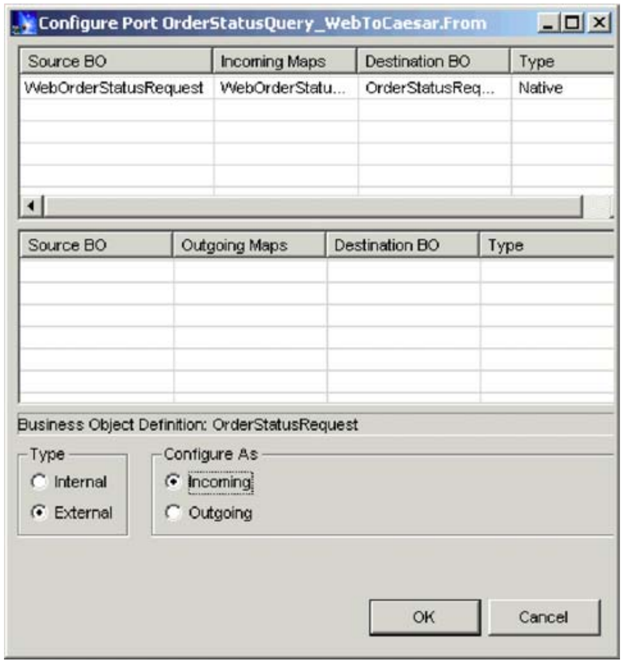
Configuring external port bindings

To bind a collaboration port to a connector or to the port of another collaboration object, do the following:

1. Enable the radio button labeled **External** in the Type pane of the port configuration dialog.
2. In the Configure As pane, select the **Incoming** radio button if business object requests are received by the port, or select the **Outgoing** radio button if business object responses are sent out of the port.
3. Do one of the following to associate the desired components with the port:
   - Drag-and-drop a business object definition from the integration component library folder into either the pane with the **Incoming Maps** or **Outgoing Maps** column, depending on whether business object requests are received by the port or if business object responses are sent out of the port.

     When presented with the Business Object Type dialog, either select the **Source Business Object** or **Destination Business Object** radio button, depending on whether the business object you drag-and-dropped is a source or destination object in the map that transforms it.
   - Drag-and-drop a map definition that transforms the type of business object supported by the port from the integration component library folder into either the pane with the **Incoming Maps** or **Outgoing Maps** column, depending on whether business object requests are received by the port or if business object responses are sent out of the port.
4. Click **OK**.

Figure 41 shows the port configuration dialog with an external binding type.



Configure Port OrderStatusQuery_WebToCaesar.From

Source BO	Incoming Maps	Destination BO	Type
WebOrderStatusRequest	WebOrderStatu...	OrderStatusReq...	Native

Source BO	Outgoing Maps	Destination BO	Type

Business Object Definition: OrderStatusRequest

Type: Internal [✓ External]

Configure As: [✓ Incoming] Outgoing

Figure 41. Configuring external collaboration object ports

For more information on implementing integrations with external programmatic entities, see the *Access Development Guide*.

Configuring collaboration object general properties

Collaboration general properties are properties that belong to all collaboration objects, regardless of how the developer designed the template upon which the objects are based. They affect the behavior of the collaboration object in the system as a whole.

To modify a collaboration object's general properties, do the following:

1. Access the collaboration object Properties dialog by doing one of the following:
   - Right-click the collaboration object in an integration component library and select **Properties** from the context menu; the dialog displays the Collaboration General Properties tab by default
   - Double-click the collaboration object icon in graphical view; the dialog displays the Collaboration General Properties tab by default
   - Right-click the collaboration object icon in either graphical view or tree view and choose **Properties** from the context menu; the dialog displays the Collaboration General Properties tab by default
2. Set the property to the desired value. For a description of each property and their possible values, see Table 15 and the sections that follow.
3. Click **OK**.

Table 15. Collaboration object general properties

Property name	**Possible values**
Create from template	Read-only name of the template upon which the collaboration object is based
Effective transaction level	`None`, `Minimal Effort`, `Best Effort`, or `Stringent`
Minimum transaction level	`None`, `Minimal Effort`, `Best Effort`, or `Stringent`, depending on the design of the collaboration template

System trace level	`0 - No Tracing,`
	`1 - Collaboration operations,`
	`2 - And collaboration events,`
	`3 - And state transactions,`
	`4 - And incoming/outgoing messages,`
	`5 - And detailed message contents`

Collaboration trace level	`0` through `5`
Email notification address	any valid e-mail alias
Pause when critical error occurs	checked or unchecked
Maximum number of concurrent events	**0** through **9999**
Persist service call in transit state	checked or unchecked
Recovery mode	**Always** or **Deferred**
Implicit database transaction	checked or unchecked
Max event capacity	an integral value between 1 and 2147483647
Blocking type	checked or unchecked

Create from template

This read-only text field displays the name of the collaboration template on which the collaboration object is based.

Effective transaction level

The effective transaction level is a range between the highest maximum transaction level of all collaboration objects and the lowest maximum transaction level of all connectors that are bound to the object.

You can lower the effective transaction level of a collaboration if you need to bind to a connector that does not support a certain transaction level.

To change the effective transaction level, choose the desired value from the **Effective transaction level** drop-down list and then click **OK**.

Note: You cannot bind objects whose transaction levels are incompatible. If you need to bind such objects to a collaboration, you must adjust the supported transaction levels accordingly.

Minimum transaction level

The minimum transaction level specifies the lowest transaction level of the collaboration template and any collaboration objects based on it.

For instance, if the developer of the collaboration template specified a minimum transaction level of `Best Effort` for the template, then all objects based on the template must operate at `Best Effort` or `Stringent`. As described in the section on "Effective transaction level," all components bound to the collaboration object must support its effective transaction level. The minimum transaction level is, then, a way for the collaboration template developer to dictate the lowest transaction level at which the entire interface in which collaboration objects based on the template will execute.

The **Minimum transaction level** field is read-only when configuring a collaboration object; it can only be changed in the collaboration template.

For more information about making changes to collaboration templates, see the *Collaboration Development Guide*.

System trace level

You can configure collaboration objects so that information about the execution of the collaboration is reported in the server output. To do so, select the desired value in the **System trace level** drop-down menu. Table 16 describes the different levels and the types of information reported at each:

Table 16. System trace levels

System trace level	**Reported information**
`0 - No Tracing`	No information is traced at this level
`1 - Collaboration operations`	Traces the receipt of business objects from connectors and the starting of scenarios
`2 - And collaboration events`	Prints messages for level 1, as well as the start and completion of each scenario, including both forward execution and rollback
`3 - And state transactions`	Prints messages for levels 1 and 2, as well as the execution of each scenario decision block or action node
`4 - And incoming/outgoing messages`	Prints messages for levels 1 through 3, as well as the sending and receipt of each business object by each scenario

Table 16. System trace levels (continued)

System trace level	Reported information
`5 - And detailed message contents`	Prints messages for levels 1 through 4, as well as printing the structure of the business object being processed, with the value of each attribute

Collaboration trace level

Collaboration developers can code collaboration templates with template-specific tracing. While system tracing (described in "System trace level" on page 165) offers tracing information about the collaboration runtime in general, collaboration tracing provides information about the specific collaboration; for instance:

- The developer might design the collaboration to report business decisions that are made based on the data in the business object being processed by the collaboration.
- The developer can provide system-type tracing but only do so at certain points in the execution of the collaboration. For instance, system trace level 5 provides full dumps of the business object data in the server output, which can be very helpful for debugging problems, but can also be very cumbersome to read through when it occurs several times during the course of a single collaboration execution. A collaboration developer can provide a full business object dump that is executed during a level of collaboration tracing, and only do it at one particularly problematic point in the flow of the collaboration, so that there is only one business object dump the user must find and read in the server output.

To set the collaboration trace level, select the desired value in the range of `0` through `5` in the **Collaboration trace level** drop-down menu.

To find out what kind of information is reported at the different trace levels for a particular collaboration, reference the documentation for that collaboration template.

For more information about making changes to collaboration templates to implement collaboration tracing, see the *Collaboration Development Guide*.

Email notification address

You can configure a collaboration object so that e-mail notifications are sent in the event of errors related to that specific object. This facilitates interface-specific administration if that is desired. For instance, a site might have one administrator responsible for errors related to InterChange Server Express itself, another administrator specifically for a customer account synchronization interface, and another administrator specifically for an order processing interface.

Type the SMTP-compliant e-mail addresses to which you want error notifications sent into the **Email notification address** field. You can type multiple addresses as long as they are separated by commas. For more information on enabling the system to send e-mail notifications, see "Configuring e-mail notification properties using System Manager" on page 106 and the *System Administration Guide*.

Many collaboration templates feature greater configurability with regard to when e-mail notifications are sent, above and beyond the setting of the value of the **Email notification address** field. The level of configurability depends entirely on the design of the collaboration template, so reference the documentation for the specific collaboration template for information on how it approaches e-mail notification.

Pause when critical error occurs

Errors can occur that prevent a collaboration from having its business object requests processed by connectors to which it sends those requests. Some such errors are:

- Inability of the connector to log in to the application.
- A time-out in the communication between the connector and the application.
- A connector agent changing to an unknown state.

If these types of errors occur, the collaboration sends the request to the connector but then the flow fails because of the problem. This happens for any request sent until the problem is resolved, and can result in a large number of failed flows for the interface if the transaction volume is high and the error persists for a significant amount of time.

You can configure a collaboration object so that it stops sending requests to a connector after a request fails due to a critical error. To do so, enable the **Pause when critical error occurs** checkbox.

For more information on critical errors, a collaboration object's response to them, and the functionality of this feature, see the *System Administration Guide*.

Maximum number of concurrent events

You can configure collaboration objects to process multiple event-triggered flows concurrently, which can improve throughput for the interface. To do so, set the **Maximum number of concurrent events** drop-down menu to the number of events between `0` and `9999` that you want the collaboration object to process concurrently.

To truly receive the benefit of this collaboration ability you must also configure other components that participate in the interface to behave similarly. For more information, see "Implementing concurrent processing of event-triggered flows" on page 279.

Persist service call in transit state

It is possible for an error to occur after a collaboration has sent a business object request to any destination connectors to which it is bound and before it has received a response from them as to whether or not they successfully processed the request in their respective applications. If the connectors were unable to process the request then it needs to be processed again when the error is resolved. If the connectors were able to successfully process the request, though, and are in the process of sending InterChange Server Express a notification of that successful processing when the error occurs then InterChange Server Express would not receive the notification. The last record of the state of the request would indicate that the request still needs to be processed. As a result of this inaccurate state record the request would be processed a second time, resulting in duplicate data.

To guard against this complication when configuring non-transactional collaborations you can enable the **Persist service call in transit state** checkbox. This causes InterChange Server Express to persist any business object requests that are in transit to destination applications when an error occurs. The requests are not sent when the system recovers, reducing the risk of the request being processed twice in the destination applications. You can then use Flow Manager to examine the requests and can investigate the destination applications to determine whether or not the requests were processed successfully before the error occurred. You should discard any requests that were processed successfully and resubmit any that were not processed successfully.

There are also programmatic measures that can be taken to handle these types of transport-related failures. For more information, refer to the information on handling service call transport exceptions in the *Collaboration Development Guide*.

Recovery mode

In previous releases, if a fatal system error occurred then all flows being processed at the time of failure would be recovered when the system was restarted. The flows were all read from persistent storage and resubmitted for processing. If there were many such flows, the system memory could actually be consumed entirely by the act of retrieving the events into memory, possibly resulting in another fatal system error related to lack of sufficient memory. Furthermore, you could not effectively manage the system with the tools until InterChange Server Express had completed its recovery processing.

It is possible to defer recovery for a collaboration object by setting the **Recovery mode** drop-down menu to the value `Deferred`. If you configure a collaboration object for deferred recovery in this manner, any flows that are in progress at the time of a fatal system error are treated as failed flows and are not recovered immediately when the system is restarted. The administrator can then use Flow Manager to resubmit the flows after the system has restarted and any desired administrative actions have been taken prior to the recovery efforts.

Important: For some interfaces it is important that event be processed in the order in which they are received. If event sequencing is not important to the integrity of an interface then you can implement deferred recovery without regard to the order in which failed or new flows are processed. If event sequencing is important, however, and you need to implement deferred recovery then you must document and follow administrative procedures to ensure that event sequencing is maintained. For instance, the administrator must make sure that the collaboration does not receive and process new flows when the system is restarted before the failed flows are resolved. They can do this managing the startup or polling of the source connector agents that would send new events to the collaboration in question.

The need for deferred recovery has been mitigated by optimized recovery measures. Rather than reading the entire business object data for in-progress transactions into memory, the server only reads enough information in memory to locate the business object in persistent storage. Although deferred recovery is still available, you may not require it because of the optimized recovery approach.

Implicit database transaction

If the collaboration template upon which the collaboration object you are configuring is based implements transactional database logic then you need to configure the collaboration object for either implicit or explicit transaction bracketing. If the collaboration was developed so that the transaction semantics are handled explicitly by the code written by the developer then you should leave the **Implicit database transaction** checkbox cleared. If the collaboration was not developed with explicit management of the database transactions, however, then you should enable the **Implicit database transaction** checkbox.

For more information about implicit and explicit transaction bracketing, see the *Collaboration Development Guide.*

Max event capacity

As business objects are received by InterChange Server Express they are queued in memory for processing. It is possible for interfaces that have a very high volume of transactions and a low rate of processing to have so many business objects queue up that InterChange Server Express experiences a fatal out-of-memory error.

You can mitigate the risk of out-of-memory errors by configuring a maximum event capacity for a collaboration object. To do so, set the **Max event capacity** field to a value that specifies the maximum number of events you want queued for a collaboration object. The system will not queue more events than the number configured for this property in memory; depending on how you configure the "Blocking type" property of the collaboration object, the system then responds differently to the receipt of new business objects for the collaboration.

The valid range of values for this property is from `1` to `2147483647`.

For more information on the **Blocking type** property, see "Blocking type."

For more information on flow control properties of connector definitions, and on how to use Connector Configurator to modify connector definitions, see Chapter 7, "Configuring connectors," on page 121.

For more information on system properties related to flow control, see the *System Administration Guide*.

Blocking type

Use the "Blocking type" property in conjunction with the "Max event capacity" property to regulate the flow of business objects to a collaboration.

If you enable the **Blocking Type** checkbox when configuring a collaboration object and the number of events in the collaboration's in-memory queue equals the number specified for its **Max event capacity** property, then the connector controller responsible for sending business objects to the collaboration will cease to do so. When the number of events in the collaboration's queue is no longer equal to the number specified for its **Max event capacity** property, then the connector controller will resume sending events to the collaboration.

If you leave the **Blocking Type** checkbox cleared when configuring a collaboration object and the number of events in the collaboration's in-memory queue equals the number specified for its **Max event capacity** property, then new events submitted to the collaboration by the connector controller are stored persistently in the database. The events are then read from the database into memory as the events currently in the collaboration's queue are processed.

For more information on the **Max event capacity** property, see "Max event capacity."

For more information on system properties related to flow control, see the *System Administration Guide*

Configuring collaboration-specific properties

Collaboration developers can design collaboration templates so that they have their own individual properties that can then be used to affect the business process logic executed at runtime. This provides a very flexible architecture so that collaboration templates can be customized to implementation-specific requirements.

To modify a collaboration object's template-specific properties, do the following:

1. Access the Properties dialog for the collaboration object by doing one of the following:
   - Right-click the collaboration object in an integration component library and select **Properties** from the context menu and then click the Properties tab
   - Double-click the collaboration object icon in graphical view and then click the Properties tab
   - Right-click the collaboration object icon in either graphical view or tree view and choose **Properties** from the context menu, then click the Properties tab
2. Type the desired value in the **Value** field for the property listed in the **Property name** field.
3. Click **OK**.

For information about the properties specific to a collaboration template and their valid values, see the documentation for the collaboration template.

For information on adding properties to a template or modifying the usage of existing properties in a collaboration template, see the *Collaboration Development Guide*.

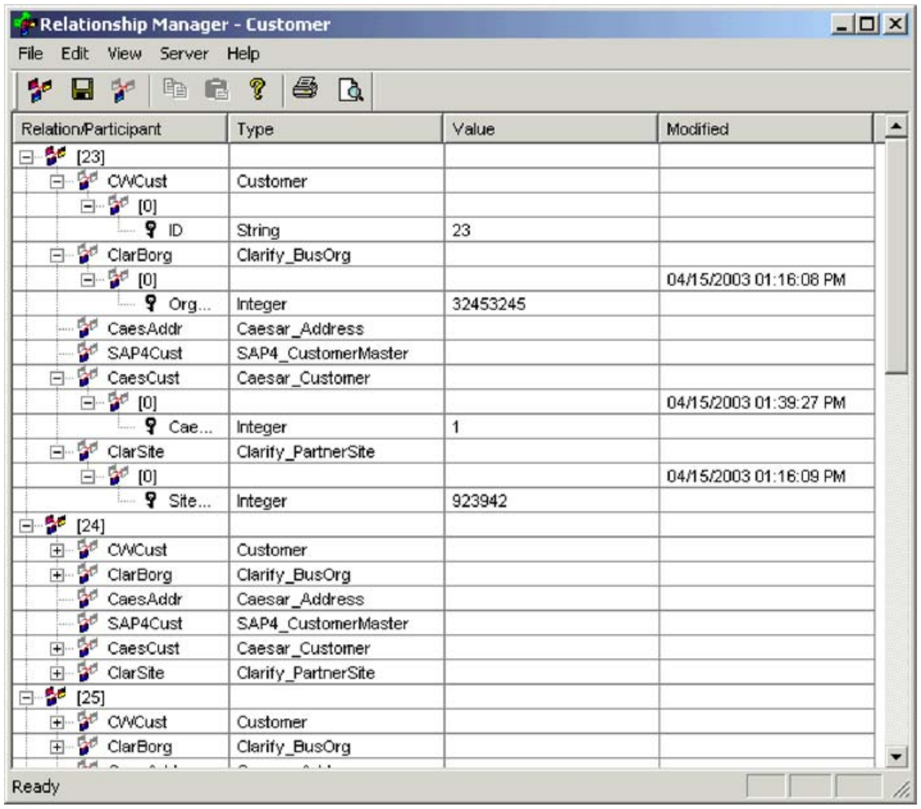
Chapter 10. Using Relationship Manager

Relationship Manager allows you to view and perform operations on relationship runtime data, including participants and their data. For background information about relationships, see the *Map Development Guide*.

You create relationship definitions with Relationship Designer Express. At runtime, instances of the relationships are populated with the data that associates information from different applications. This relationship instance data is created when the maps that use the relationships execute. The data is stored in the relationship tables specified in the relationship definition. Relationship Manager provides a graphical interface to interact with the relationship tables regardless of the database vendor.

For each relationship instance, Relationship Manager displays a hierarchical listing of its participant definitions and participant instances, which are a set of key and non-key attributes. The relationship tree also provides detailed information about each of the participants in the relationship instance such as the type of entity, its value, and the date it was last modified. A relationship instance ID is automatically generated when the relationship instance is saved in the relationship table. Relationship Manager displays this instance ID at the top level of the relationship tree.

Figure 42 on page 172 shows a sample of a relationship tree in Relationship Manager for an identity relationship.



Relationship Manager - Customer

File Edit View Server Help

Relation/Participant	Type	Value	Modified
[23]			
CWCust	Customer		
[0]			
ID	String	23	
ClarBorg	Clarify_BusOrg		
[0]			04/15/2003 01:16:08 PM
Org...	Integer	32453245	
CaesAddr	Caesar_Address		
SAP4Cust	SAP4_CustomerMaster		
CaesCust	Caesar_Customer		
[0]			04/15/2003 01:39:27 PM
Cae...	Integer	1	
ClarSite	Clarify_PartnerSite		
[0]			04/15/2003 01:16:09 PM
Site...	Integer	923942	
[24]			
CWCust	Customer		
ClarBorg	Clarify_BusOrg		
CaesAddr	Caesar_Address		
SAP4Cust	SAP4_CustomerMaster		
CaesCust	Caesar_Customer		
ClarSite	Clarify_PartnerSite		
[25]			
CWCust	Customer		
ClarBorg	Clarify_BusOrg		

Ready

Figure 42. Relationship Manager

You can use Relationship Manager to work on entities at all levels: the relationship instance, participant instance, and attribute levels. For example, you can use Relationship Manager to:

- Create and delete relationship instances
- Modify the contents of a relationship instance, such as adding and deleting participants
- Add and save a participant's data, load a participant's data from or save data to a file, and copy and paste the data of a participant from another relationship into the relationship instance, thus creating a new participant (as long as the participant types are identical)
- Activate and deactivate participants
- Retrieve participants based on instance IDs, business object attribute values, or data
- Filter a participant's activity within a time interval
- Salvage a situation when problems with data arise. For example, when corrupt or inconsistent data from a source application has been sent to the generic and destination application relationship tables, you can use Relationship Manager to rollback (or clean up) the data back to a point of time when you know the data is reliable.

Starting Relationship Manager

Do one of the following to start Relationship Manager:

- Select **Start > Programs > IBM WebSphere Integration Express Plus> Toolset Express> Administrative > Relationship Manager**

- In Relationship Designer Express, select a relationship definition and then select **Tools > Relationship Manager** from the menu bar

Relationship Manager starts. At this point it is disconnected from the server; you must connect to an InterChange Server Express instance as described in "Connecting to InterChange Server Express" to proceed further.

Connecting to and disconnecting from a server

You must connect Relationship Manager to InterChange Server Express to work with relationship instances and data. Follow the instructions in the following sections to connect Relationship Manager to a server and disconnect Relationship Manager from it:

- "Connecting to InterChange Server Express"
- "Disconnecting from InterChange Server Express" on page 174

Connecting to InterChange Server Express

Do the following to connect Relationship Manager to InterChange Server Express:

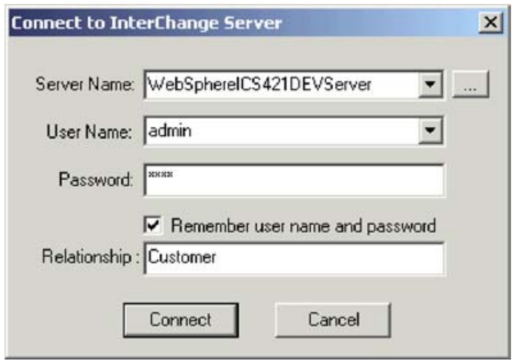
1. Select **Server > Connect to Server** from the menu bar of Relationship Manager.
2. Do one of the following to populate the name of the InterChange Server Express instance to which you want to connect in the **Server Name** field:
   - Type the name of the InterChange Server Express instance in the **Server Name** field.

     Important: The name of an InterChange Server Express instance is case-sensitive, so you must be sure to be accurate when specifying the name.
   - Select a cached server name from the drop-down menu.
   - Do the following to browse for the InterChange Server Express instance on the network:
     a. Click the browse button.
     b. At the "Servers" dialog, select the desired InterChange Server Express instance from the list.
     c. Click **OK**.
3. Type the user name to interact with the InterChange Server Express instance in the **User Name** field.
4. Type the password for the user name supplied in step 3 in the **Password** field.
5. If you do not want to have to supply the user name and password each time you have to connect to the InterChange Server Express instance in System Manager then enable the **Remember user name and password** checkbox.
6. If you want to open a relationship at this time type the name of the relationship definition in the **Relationship** field.

   If you do not want to open a relationship at this time you can open it after connecting to the server. For more information, see "Opening a relationship" on page 174.
7. Click **Connect**.

   If you connect to InterChange Server Express in Relationship Manager and specify a relationship to open as described in step 6, then Relationship Manager displays the "retrieve relationships" window, described in "Retrieving relationship instances" on page 175.

Figure 43 shows the "Connect to InterChange Server Express" dialog.



Server Name: WebSphereICS421DEVServer
User Name: admin
Password: ****
☑ Remember user name and password
Relationship : Customer

Figure 43. Connecting to InterChange Server Express

Disconnecting from InterChange Server Express

To disconnect Relationship Manager from InterChange Server Express, select **Server > Disconnect** from the menu bar of Relationship Manager.

Working with relationships

Once you have started Relationship Manager and connected it to an InterChange Server Express, you can use Relationship Manager to work with relationship data as described in the following sections.

Opening a relationship

Do the following to open a relationship definition in Relationship Manager after it is already connected to the server:

1. Select **File > Open** from the menu bar of Relationship Manager.
2. At the "Open Relationship" window, select the name of the relationship you want to open.

   Figure 44 on page 175 shows the "Open Relationship" window.

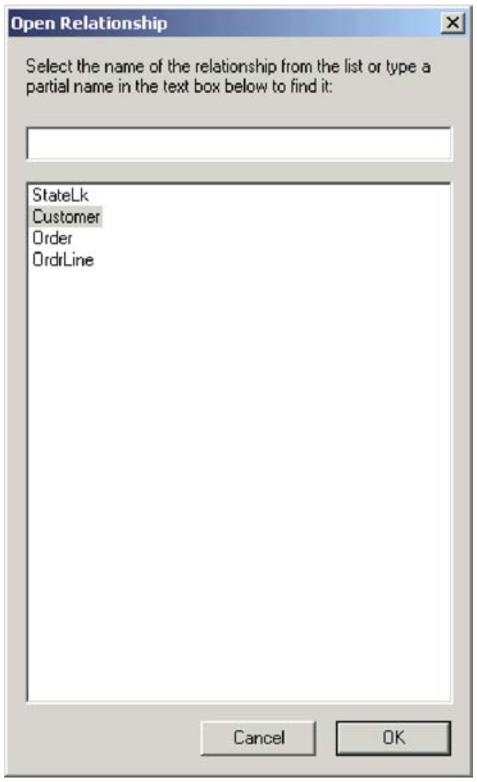

Open Relationship
Select the name of the relationship from the list or type a partial name in the text box below to find it:
StateLk
Customer
Order
OrdrLine
Cancel
OK

Figure 44. Opening a relationship

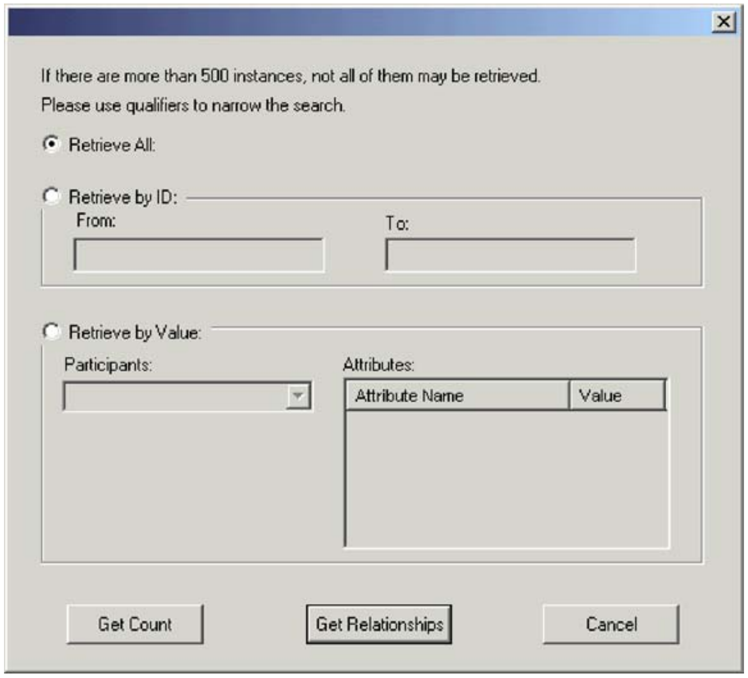
3. Click **OK**.

   When you open a relationship, Relationship Manager displays the "retrieve relationships" window, described in "Retrieving relationship instances."

Retrieving relationship instances

Select **File > Retrieve Relationship** from the menu bar of Relationship Manager to retrieve relationship instances or return a count of how many instances there are for a relationship. This displays the "retrieve relationships" window shown in

Figure 45 on page 176 shows the "retrieve relationships" window.


If there are more than 500 instances, not all of them may be retrieved.
Please use qualifiers to narrow the search.

(●) Retrieve All:

( ) Retrieve by ID:
From:
To:

( ) Retrieve by Value:
Participants:
Attributes:

Attribute Name	Value

Figure 45. Retrieving relationship instances

The "retrieve relationships" window is also displayed when you open a relationship or connect to InterChange Server Express and specify a relationship to open.

You can perform the following operations with the "retrieve relationships" window:

- Retrieve the first 500 instances for the relationship, as described in "Retrieving all instances."
- Retrieve a range of instances for the relationship based on the relationship instance ids, as described in "Retrieving by relationship ID."
- Retrieve a relationship instance that contains a participant of a particular value you specify, as described in "Retrieving by participant data" on page 177.
- Return a count of the number of instances for the relationship, as described in "Returning a count of the relationship instances" on page 177.

Depending on the number of participants in the relationship definition and the number of participant instances in each relationship instance, these retrieval queries may take some time.

Retrieving all instances

Do the following to retrieve the first 500 instances for a relationship:

1. At the "retrieve relationships" window click **Retrieve All**.
2. Click **Get Relationships**.

   Relationship Manager displays the first 500 instances for the relationship.

Retrieving by relationship ID

Do the following to retrieve a range of up to 500 instances:

1. At the "retrieve relationships" window click **Retrieve by ID**.
2. Type the id of the first instance in the range you want to retrieve in the **From** field.

3. Type the id of the last instance in the range you want to retrieve in the **To** field.
4. Click **Get Relationships**.

   Relationship Manager displays up to 500 instances in a range of the ids you specify.

Retrieving by participant data

Do the following to retrieve a relationship instance based on values for key or nonkey attributes of selected participants:

1. At the “retrieve relationships” window click **Retrieve by Value**.
2. Select the participant whose value you want to search on from the **Participants** drop-down menu.

   For identity relationships, the drop-down menu lists the participant names followed by the business object definition with which the participant is associated.

   For lookup relationships, the drop-down menu lists the participant names followed by the word “Data”.
3. Type the one of the types of values listed in Table 17 in the **Value** column in the “Attributes” pane.

Table 17. Supported values for retrieving relationship instances by participant data

Value	Description
Participant data	The data of the selected participant. For example, if the relationship is an identity relationship you would specify the id of the participant instance that you know to find the relationship instance in which it exists. If the relationship is a lookup relationship you would specify the non-key data value of the participant instance.
`%`	Any string of characters. This option is case-sensitive; numbers are included in the character set. For example, if `%A` were specified for a participant that stores abbreviated forms of the names of the United States, the values `CA`, `GA`, `IA`, `LA`, `MA`, `PA`, `VA`, and `WA` would be returned.
`_`	Any single character. As an example, `_00` would retrieve `100`, `200`, `a00`, `b00`, and so forth.

4. Click **Get Relationships**.

   Relationship Manager displays the first 500 relationship instances that match the specified value.

Returning a count of the relationship instances

To return the number of relationship instances that satisfy a retrieval criteria, select the options for the criteria as described in “Retrieving all instances” on page 176, “Retrieving by relationship ID” on page 176, or “Retrieving by participant data,” and then click **Get Count** instead of **Get Relationships**.

Creating relationship instances

Do the following to create a new instance for a relationship:

1. Create the new relationship instance by performing any of the following tasks:
   - Select **File > New** from the menu bar.
   - Use the keyboard shortcut **Ctrl+N**.

- Click **New Relationship** in the toolbar.

  Relationship Manager displays the new relationship instance.

  Highlighted at the top of the hierarchal relationship tree, on the entry line with the relationship icon is the placeholder for the relationship instance ID, which displays three question marks (???). Once you save the relationship instance or any of its participants, InterChange Server automatically generates the new relationship instance ID and Relationship Manager replaces the question marks with this instance ID.

2. Expand the new relationship instance by clicking on the plus (+) sign next to the ??? placeholder icon.

   The relationship tree displays participant definitions, participant instances, and participant key and non-key attributes beneath the relationship instance in descending order.

3. Do the following to create a new participant instance in the relationship instance:
   a. In the relationship tree, select the participant definition for which you want to create an instance.
   b. Do one of the following to add an instance for the participant:
      - Right-click a participant definition in the listing and choose **Add Participant** from the context menu.
      - Click **Add Participant** in the standard toolbar.
   c. Expand the new participant instance by clicking on the plus (+) sign next to it.
   d. Select the new participant instance.
   e. Click the **Value** column for the participant instance once, then type the desired value into the cell.

      Note: If the **Value** field for the attribute displays three question marks (???), the participant is managed by InterChange Server Express. You cannot enter values for these participants because InterChange Server Express automatically generates them when you save the relationship instance. The value is the same value as the relationship instance ID.

At this point, you can perform any of the tasks in Table 18.

Table 18. Tasks for Participant Data

Task	Action
Save the participant instance.	To save the new participant instance, right-click the participant instance and choose **Save Participant** from the context menu. Relationship Manager saves in the appropriate relationship table the data for this participant. The **Modified** column for the participant instances displays the date the participant was saved, which is the create date, in this case. **Note:** Once the participant data has been saved, it cannot be changed. To change its data, the participant must be deleted and another created.
Add more participant instances.	Repeat repeat step 3 in the previous list. **Note:** If you are working with an identity relationship, you cannot create more than one participant instance for a participant definition.

Table 18. Tasks for Participant Data (continued)

Task	Action
Delete a participant.	If necessary, you can delete a saved participant instance by right-clicking on the participant instance and choosing **Delete Participant** from the context menu. Relationship Manager removes the participant instance from the relationship table. If you do not want to remove the participant instance from the database, use the **Deactivate Participant** option (see "Deactivating and activating participants" on page 179). A deactivated participant retains its instance ID and its values.
Save the relationship instance.	Save the relationship instance by performing one of the following tasks: • Select **File > Save** from the menu bar (activated when a relationship instance is selected). • Right-click the relationship instance and choose **Save Relationship** from the context menu. InterChange Server Express generates the relationship instance ID and Relationship Manager replaces the ??? placeholder with this new ID. Relationship Manager updates the modified date on all saved participant instances to this date. **Note:** At least one participant instance and all key attribute data must be created before the relationship instance can be saved.
Save all relationship instances.	Select **File > Save All** from the menu bar. InterChange Server Express generates the relationship instance IDs for any relationship instances that do not have one. Relationship Manager replaces any "???" placeholders with the new IDs. Relationship Manager updates the modified date on all saved participant instances to this date.

Deleting relationship instances

To delete a relationship instance from the relationship tables, select the relationship instance you want to delete and perform one of the following actions:

- Select **File > Delete Relationship** from the menu bar.
- Right-click the relationship instance and select **Delete** from the context menu.

The relationship instance and its data are deleted from the relationship tables for the current relationship.

Deactivating and activating participants

A participant instance can be deactivated, or made inactive. Deactivating a participant instance removes it from the relationship instance and prevents it from displaying in the Relationship Manager window, but its record remains in the relationship table so it can be re-activated in the future.

Deactivating a participant

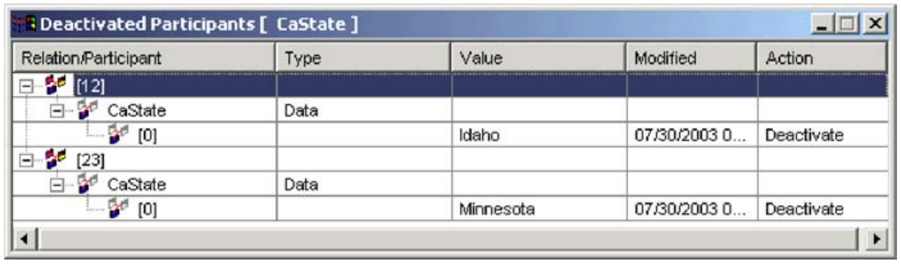
To deactivate a participant instance, right-click the participant instance you want to deactivate and choose **Deactivate Participant** from the context menu. The participant is removed from the Relationship Manager display but not from the relationship tables.

Activating a participant

To activate a participant instance, take the following steps:

1. Select **View > Deactivated Participants** from the menu bar.

   The Deactivated Participants window displays as shown in Figure 46 on page 180.


Deactivated Participants [CaState]

Relation/Participant	Type	Value	Modified	Action
[12]				
CaState	Data			
[0]		Idaho	07/30/2003 0...	Deactivate
[23]				
CaState	Data			
[0]		Minnesota	07/30/2003 0...	Deactivate

Figure 46. Deactivating participants

2. Select the relationship instance that contains the deactivated participant you want to activate from the list.
3. Expand this relationship instance until the deactivated participant instances display in the list.
4. Right-click the participant instance to reactive and choose **Activate** from the context menu.
5. Select **Edit > Refresh** from the menu bar.

   The activated participant instance displays in its relationship instance in the Relationship Manager window.

   Note: If a participant instance in an identity relationship is deactivated and another participant is added in its place (that is, assigned the same instance ID), the original participant is removed from the Deactivated Participants listing, but remains in the database.

Copying participants

You can create a new participant instance by copying an existing participant instance. To copy a participant instance, take the following steps:

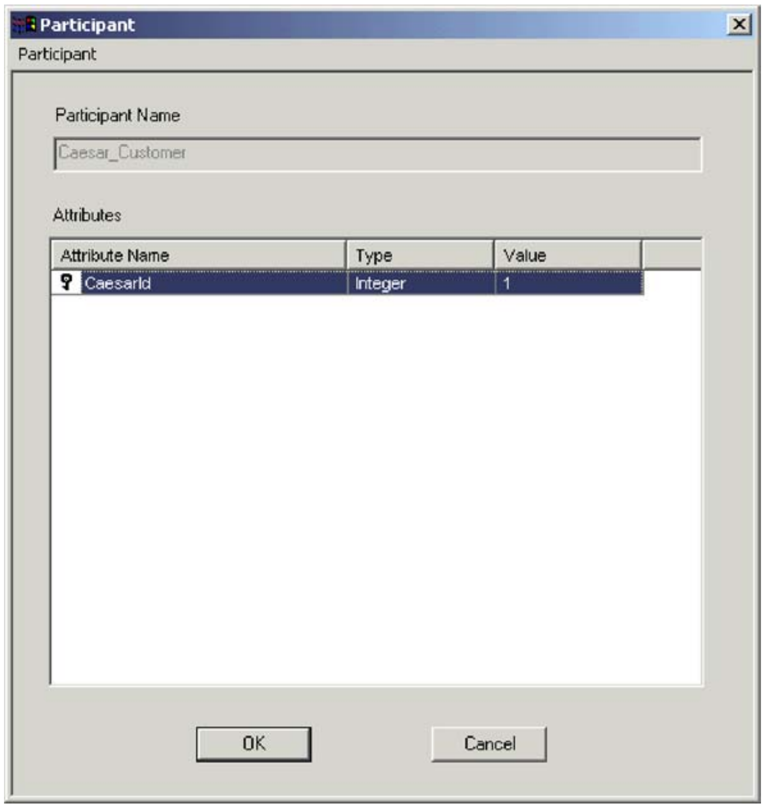
1. In the relationship instance, right-click the participant definition and choose **Add Participant** from the context menu.
2. Right-click the participant instance you want to copy and choose **Copy Participant** from the context menu.
3. Right-click the newly created participant instance and choose **Paste Participant** from the context menu.

Loading and unloading business object files

You can load a business object file of the same type into a participant. To load a business object data file into a participant, take the following steps:

1. Right-click the participant instance where you want to load the business object file and choose **Load Participant with Business Object**.

   The Participant window displays the business object associated with that participant instance, as shown in Figure 47 on page 181.


Participant

Participant Name: Caesar_Customer

Attributes

Attribute Name	Type	Value
CaesarId	Integer	1

Figure 47. Loading participants with business objects

2. Select **Participant > Load** from the menu bar.
3. Navigate to and open the business object file you want to load.
4. Click **OK**.

Note: Only the first instance of a relationship is loaded if more than one instance exists in the file.

Working with relationship data

An important feature of Relationship Manager is its ability to access and manipulate relationship runtime data contained in the relationship tables. This section describes how to use Relationship Manager to manipulate and access runtime data.

Searching for participants

You can search for participant instances based on different criteria. Depending on how specific your search criteria is, your searches can locate a unique participant instance or a group of participant instances. You can find participant instances in the following ways:

Finding instances by business object

This option searches for instances whose data type is an attribute in a business object.

Do the following to search for instances by business object:

1. Select a participant instance in Relationship Manager.

2. Select **Edit > Find Instances by Business Object** from the menu bar.

   Relationship Manager displays the "Participant" window, as shown in Figure 47 on page 181.
3. Type the participant value by which you want to search in the **Value** cell.
4. Click **OK**.

   Relationship Manager displays any matching instances in a dialog box.
5. Double-click any of the instances in the dialog displayed by Relationship Manager to navigate to and highlight the instance.

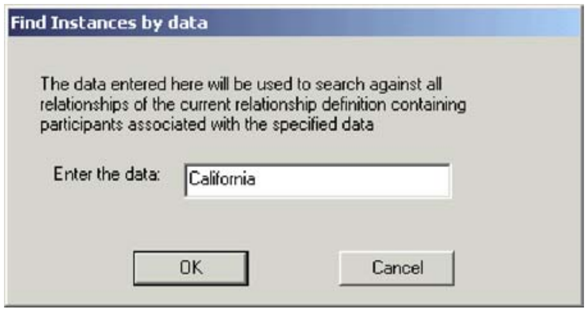
Finding instances by data

This option searches for instances whose type is Data.

Do the following to search for instances by data:

1. Select a participant instance in Relationship Manager.
2. Select **Edit > Find Instances by Data** from the menu bar.

   Relationship Manager displays the "Find Instances by Data" window, as shown in Figure 48.



Find Instances by data

The data entered here will be used to search against all relationships of the current relationship definition containing participants associated with the specified data

Enter the data: California

Figure 48. Finding instances by data

3. Type the participant value by which you want to search in the **Enter the data** cell.
4. Click **OK**.

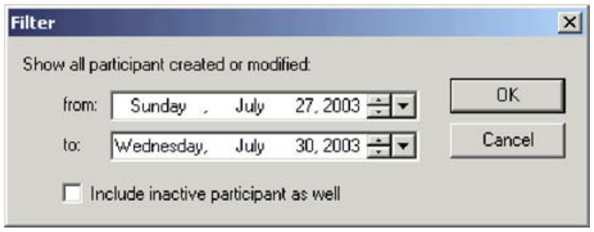
   Relationship Manager displays any matching instances in a dialog box.
5. Double-click any of the instances in the dialog displayed by Relationship Manager to navigate to and highlight the instance.

Filtering the displayed participants

You can filter the participants to only display those created or modified between certain dates. Do the following to filter the displayed participants:

1. Select a participant in Relationship Manager.
2. Select **View > Filter** from the menu bar.

   Relationship Manager displays the "Filter" dialog, as shown in


Show all participant created or modified:

from: Sunday , July 27, 2003

to: Wednesday, July 30, 2003

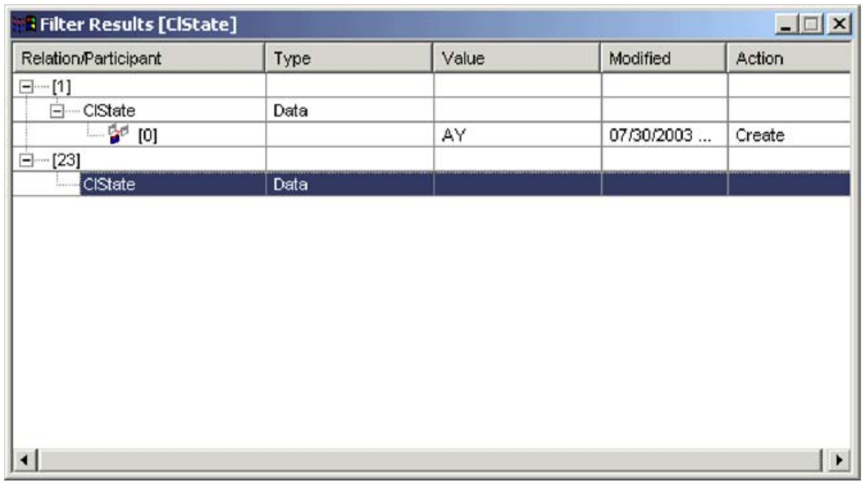
☐ Include inactive participant as well

Figure 49. Filtering participant results

3. In the “Filter” dialog, enter the earliest date of creation or modification for the participant in the **from** field and the latest date of creation or modification for the participant in the **to** field.

   Use the following techniques to enter the date value:
   - Type letters in the text field to cycle through days of the week. For instance, type **S** to cycle through dates that occur on Saturdays and Sundays.
   - Click the small up and down arrows to increment or decrement the date by one day.
   - Click the large down arrow to display a calendar that you can use to select a date.
4. Click the **Include inactive participants as well** checkbox if you want to include inactive participants in the resulting display.
5. Click **OK**.

   Relationship Manager displays the history of activity for the filtered interval in the “Filter Results” dialog. The filtered display includes inactive participants if the **Include inactive participants as well** option box is checked. Figure 50 shows the “Filter Results” dialog.



Relation/Participant	Type	Value	Modified	Action
[1]				
ClState	Data			
[0]		AY	07/30/2003 ...	Create
[23]				
ClState	Data			

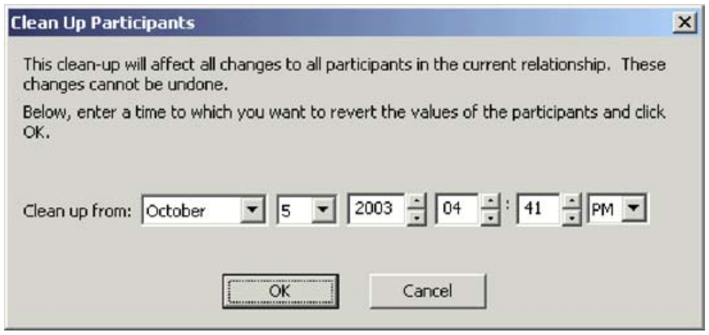
Figure 50. Viewing filtered participant data

Cleaning up participants

Do the following to clean up participants due to inconsistent or corrupt data in the source application or generic object:

1. Select a participant in Relationship Manager.
2. Select **View > Clean Up Participants** from the menu bar.

Relationship Manager displays the “Clean up Participants” dialog, as shown in Figure 51.


Clean Up Participants

This clean-up will affect all changes to all participants in the current relationship. These changes cannot be undone.

Below, enter a time to which you want to revert the values of the participants and click OK.

Clean up from: October 5 2003 04 : 41 PM

Figure 51. Cleaning up participants

3. In the “Clean Up Participants” dialog box, enter the date to which you want to revert the participant values to in the **Clean up from** field.

   Use the following techniques to enter the date value:
   - Type letters in the text field to cycle through days of the week. For instance, type **S** to cycle through dates that occur on Saturdays and Sundays.
   - Click the small up and down arrows to increment or decrement the date by one day.
   - Click the large down arrow to display a calendar that you can use to select a date.
4. Click **OK**.

   All participant adds, deactivations, and activations since that point in time are erased from the database. A participant that has been deleted or whose value has been modified cannot be cleaned up.

Printing relationship data

Relationship Manager allows you to print information about a relationship’s runtime data. It creates a tree representation of the runtime data, much like the data appears in the tool’s main window. You can manage the printing of this output tree with the following print features:

- **Print—**Send the formatted runtime data to your system printer.
- **Print Preview—**Send the formatted runtime data to a window of Relationship Manager.
- **Page Setup—**Change the layout of the runtime data so that it fits onto a page

Sending relationship data to a printer

The printing ability of Relationship Manager sends the current contents of the relationship tree in the main window to the printer. Do the following to print relationship runtime data:

1. Expand the relationship tree of Relationship Manager so that the information you want to print is displayed.
2. Print relationship runtime data in any of the following ways:
   - Select **File > Print** from the menu bar.
   - Use the keyboard shortcut **Ctrl+P**.
   - Click **Print** in the toolbar.

Using Print Preview with relationship data

Do the following to preview the output of relationship data as it would be printed:

1. Expand the relationship tree of Relationship Manager so that the information you want to print is displayed.
2. Initiate the print preview in any of the following ways:
   - Select **File > Print Preview**.
   - Use the keyboard shortcut **Ctrl+Alt+P**.
   - Click **Print Preview** in the toolbar.

Setting up the page

Relationship Manager tries to print the output tree as it appears in the tool's main window. If the output tree is narrow enough to fit on the page, you do not need to take any special formatting tasks. If, however, the output tree does not fit on the page, you can set formatting options to change the layout of the output tree so that it fits on the page.

You can set up the print format with the "Print Page Setup" feature, which you initiate in any of the following ways:

- Select **File > Page Setup** from the menu bar to display the "Print Page Setup" dialog, where you can configure information such as printer setting, paper size and orientation.
- Use the keyboard shortcut **Shift+Alt+P**.

Any of these commands invokes the "Print Page Setup" dialog.

From the "Print Page Setup" dialog, you can provide the following formatting information:

- Column formatting options appear in the "Column Settings" area of the "Print Page Setup" dialog. Table 19 lists the options contained in this area.

Table 19. Column formatting options

Column formatting option	Description
Column headers on every page	When this option is enabled, the column headers (such as Relation, Type, Value) appear on every page, not just the first one.
Resize columns to fit all data	When this option is enabled, Relationship Manager attempts to enlarge each column to fit any strings wider than the column itself. If the resulting tree does not fit onto the printed page, Relationship Manager tries to eliminate any extra room inside the columns. If this field and its subfields are greyed out, their values indicate the default values that Relationship Manager uses when formatting the output tree. Leave these fields in their greyed-out state if you want to confirm the formatting every time Relationship Manager generates the output tree. If you activate these fields, Relationship Manager uses the options you set for *every* output tree it generates.
Use smaller font if necessary	When this option is enabled, Relationship Manager tries to reduce the font size if data is still larger than the page. It reduces font size in three steps. At each step, it reduces the font size by 20%.

Table 19. Column formatting options (continued)

Column formatting option	Description
Confirm new font size	When this option is enabled, Relationship Manager prompts you to choose how to continue when it has reduced the font size sufficiently to fit the output tree on a page. It provides you with the following choices: • Print with this font • Restore and print with the original font size • Cancel printing In the process of reducing font size, this size can become as small as half the original size. If this option is disabled, printing starts when Relationship Manager finds a font size that fits the output tree on a page.
If altered during print/preview, adjust column widths	When this option is enabled, Relationship Manager saves any changes to the column sizes that it has made to get the output tree to fit on a page. It uses the saved settings for future printings of the output tree.

The **Resize columns to fit data** field and its associated subfields indicate the actions that Relationship Manager takes when it formats the output tree. These fields can appear in one of three states:

- Checkmark not selected (field is empty)—indicates that Relationship Manager does not perform the associated task when it formats the output tree.
- Checkmark selected and greyed out—indicates that Relationship Manager conditionally performs the associated task when it formats the output tree; that is, Relationship Manager displays a confirmation dialog before it takes the associated action.
- Checkmark selected—indicates that Relationship Manager unconditionally performs the associated task when it formats the output tree; that is, Relationship Manager does not display a confirmation dialog before it takes the associated action.

As Table 19 indicates, these fields initially display in a greyed-out state. You can set these fields from the confirmation dialogs that display during the formatting process. For example, if the values in a column do not fit in the current width of this column, Relationship Manager displays a confirmation dialog with the **Yes** and **No** options and a **Make this the default choice** field. If you select the **Make this the default choice** check box, Relationship Manager sets the **Resize columns to fit data** field based on the option you have selected from the confirmation dialog, as follows:

- **Yes**—Relationship Manager deselects the associated check boxes and therefore does not perform the associated action.
- **No**—Relationship Manager selects the associated check box and therefore unconditionally performs the associated action.
- Margin formatting options appear in the Margin Settings area of the “Print Page Setup” dialog. Table 20 lists the options contained in this area.

Table 20. Margin formatting options

Margin formatting option	Description
Use maximum page area	When this option is enabled, Relationship Manager uses the maximum printing area of the page (according to your printer's capabilities).
Use these margins (inches)	When this option is enabled, you can set the left, right, top and bottom margins to specified values.

- Page formatting options appear in the "Page Settings" area of the "Print Page Setup" dialog. Table 21 lists the options contained in this area.

Table 21. Page formatting options

Page formatting option	Description
Print page header on every page	When this option is enabled, Relationship Manager puts the relationship name and print time at the top of every page, not just the first one.
Print page footer	When this option is enabled, Relationship Manager puts page numbers at the bottom of every page.
Synchronize document with print preview	When this option is enabled, Relationship Manager displays in its main window the relationship data that you were last viewing in the print preview. For example, if you had paged through the print preview to the third "page" of instances and then exited from the print preview, Relationship Manager scrolls through the relationship data so that this same "page" of instances appears in its main window. When this option is disabled, Relationship Manager displays in its main window the same data it displayed before you performed the print preview.

- Grid formatting options appear in the "Grid Settings" area of the "Print Page Setup" dialog. Table 22 lists the options contained in this area.

Table 22. Grid formatting options

Grid formatting option	Description
Print grid (uncheck to reset)	When this option is enabled, Relationship Manager includes grid in the output tree to separate rows and columns. When you click this field twice, you include both vertical and horizontal g rid lines in the output tree.
Horizontal grid only	When this option is enabled, Relationship Manager includes only a horizontal grid (to separate rows) in the output tree. If the output tree does *not* include a horizontal grid, each printed page can fit an additional 2-3 rows of data.
Vertical grid only	When this option is enabled, Relationship Manager includes only a vertical grid (to separate columns) in the output tree.

Chapter 11. Using Test Connector

Test Connector simulates the activities of a connector to allow you to test your integration components without the complexity of running an actual connector. This chapter consists of the following sections:

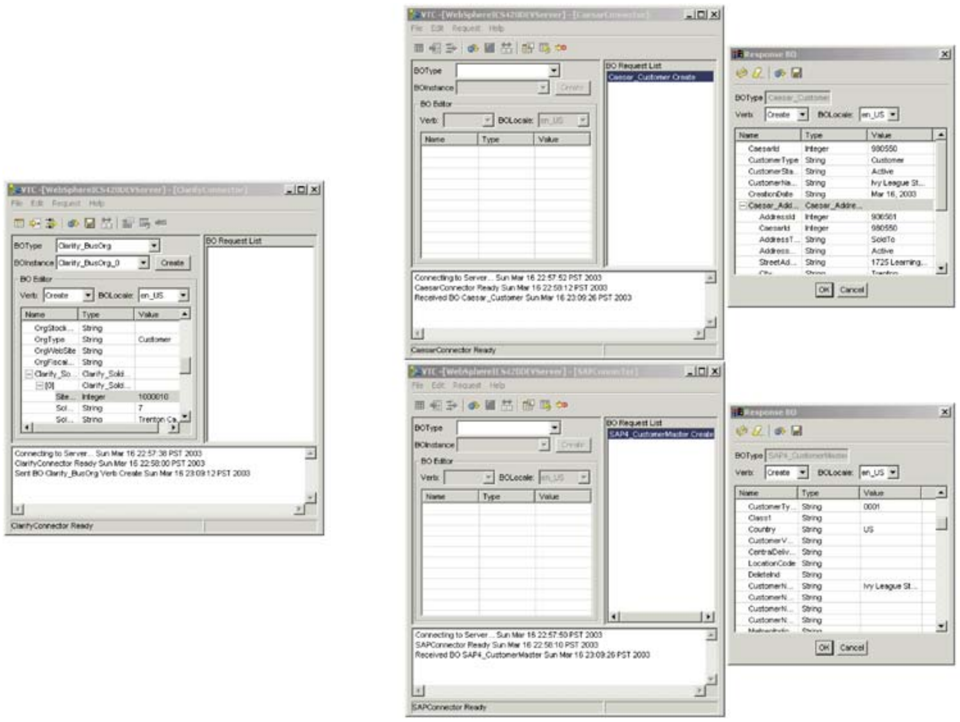
- "Recommended testing procedure"
- "Starting Test Connector" on page 190
- "Shutting down Test Connector" on page 191
- "Creating and editing connector profiles" on page 191
- "Emulating a connector" on page 192
- "Working with business objects" on page 192

Recommended testing procedure

This is the recommended test procedure for testing components in the WebSphere business integration system:

1. Consider using the System View view, which can be very helpful in determining if a flow you have sent ends in success or failure.

   For more information, see the *System Administration Guide*.
2. Set up Test Connector to emulate a source connector.
   a. Launch Test Connector as described in "Starting Test Connector" on page 190.
   b. Create a profile for the source connector in the interface as described in "Creating a new profile" on page 191.
   c. Connect Test Connector to the agent to begin emulating the source connector, as described in "Emulating a connector" on page 192.
3. Set up instances of Test Connector to emulate each destination connector involved in the interface.
   a. Launch Test Connector as described in "Starting Test Connector" on page 190.
   b. Create a profile for a destination connector as described in "Creating a new profile" on page 191.
   c. Connect Test Connector to the agent to begin emulating the destination connector as described in "Emulating a connector" on page 192.
   d. Repeat 3a through 3c above for all destination connectors involved in the interface.
4. Arrange the instances of Test Connector on your screen so that you can easily identify the connector being emulated in each Test Connector window. For example, in Figure 52 on page 190 the source Test Connector is arranged to the left of the destination Test Connector.



BOType: Clarify_BusOrg
BOInstance: Clarify_BusOrg_0
Verb: Create
BOLocale: en_US

Name	Type	Value
OrgStock...	String	
OrgType	String	Customer
OrgWebSite	String	
OrgFiscal...	String	
Clarify_So...	Clarify_Sold...	
[0]	Clarify_Sold...	
Site...	Integer	1000010
Sol...	String	7
Sol...	String	Trenton Ca...

```
Connecting to Server... Sun Mar 16 22:57:38 PST 2003
ClarifyConnector Ready Sun Mar 16 22:58:00 PST 2003
Sent BO Clarify_BusOrg Verb Create Sun Mar 16 23:09:12 PST 2003
```

ClarifyConnector Ready

BO Request List: Caesar_Customer Create

```
Connecting to Server... Sun Mar 16 22:57:52 PST 2003
CaesarConnector Ready Sun Mar 16 22:58:12 PST 2003
Received BO Caesar_Customer Sun Mar 16 23:09:26 PST 2003
```

CaesarConnector Ready

BOType: Caesar_Customer
Verb: Create
BOLocale: en_US

Name	Type	Value
CaesarId	Integer	980550
CustomerType	String	Customer
CustomerSta...	String	Active
CustomerNa...	String	Ivy League St...
CreationDate	String	Mar 16, 2003
Caesar_Add...	Caesar_Addre...	
AddressId	Integer	906501
CaesarId	Integer	980550
AddressT...	String	SoldTo
Address...	String	Active
StreetAd...	String	1725 Learning...

BO Request List: SAP4_CustomerMaster Create

```
Connecting to Server... Sun Mar 16 22:57:50 PST 2003
SAPConnector Ready Sun Mar 16 22:58:10 PST 2003
Received BO SAP4_CustomerMaster Sun Mar 16 23:09:26 PST 2003
```

SAPConnector Ready

BOType: SAP4_CustomerMaster
Verb: Create
BOLocale: en_US

Name	Type	Value
CustomerTy...	String	0001
Class1	String	
Country	String	US
CustomerV...	String	
CentralDeliv...	String	
LocationCode	String	
DeleteInd	String	
CustomerN...	String	Ivy League St...
CustomerN...	String	
CustomerN...	String	
CustomerN...	String	

Figure 52. Source and destination instances of Test Connector.

5. Send a request business object from the source connector. From the source Test Connector, do the following:
   a. Create a business object that is managed by the interface you need to test as described in “Creating request business objects” on page 193.
   b. Save the business object to a file to use in subsequent tests as described in “Saving a business object” on page 197.
   c. Send the business object as described in “Sending request business objects” on page 194.
6. Simulate the response to the request business object from the destination connector. From the destination Test Connector window, do the following:
   a. Accept the request business object as described in “Accepting a request business object” on page 198.
   b. Send the business object as a response as described in “Sending a response business object” on page 199.
7. Repeat step 5 through step 6 as many times as necessary to test each interface.

Starting Test Connector

To start Test Connector, select **Start > Programs > IBM WebSphere Integration Express >Toolset Express> Development > Test Connector**.

The Test Connector window includes the following panes:

- The “Supported Business Objects” pane in which you can create business object instances to send
- The “BO Request List” pane, which displays any business object requests that the connector has received
- The “Output” pane, which displays messages about Test Connector’s operations, such as when a business object has been sent.

Shutting down Test Connector

To shut down Test Connector and cause it to stop emulating a connector agent, select **File > Exit** from the menu bar. When presented with the “Shutdown” prompt, click **Yes**.

Creating and editing connector profiles

Test Connector uses profiles to store the information it needs to emulate a connector. You must create a profile for each connector you want to emulate. You can edit and delete existing profiles.

Saving the connector definition to a file

To emulate a connector using Test Connector, you must save the connector definition to a file. Do the following to save a connector definition to a file:

1. Open the connector definition in Connector Configurator.
2. Select **File > Save As > To File** from the menu bar.
3. Navigate to the directory in which you want the file saved, type a name in the **File name** field, ensure that the value `Configuration (*.cfg)` is displayed in the Save as type drop-down menu, and click **Save**.

   Connector Configurator saves the connector definition to a file with the specified name.

Creating a new profile

You must create a profile for any connector you want to emulate in Test Connector. The profile specifies information such as the name of the connector, and the configuration file to be used. To create a new connector profile, do the following:

1. Select **File > Create/Select Profile** from the menu bar to display the “Connector Profile” window.
2. In the “Connector Profile window”, select **File > New Profile** from the menu bar.
3. In the “New Profile” window, click **Browse** and then navigate to the configuration file for the connector you preparing in “Saving the connector definition to a file.”
4. Type the name of the connector in the **Connector Name** field. You must type the exact name of the connector definition as it exists in the integration broker repository. For the adapter for JText, for instance, you must type `JTextConnector`, without any spaces between the words `JText` and `Connector`, and with each letter being the proper case.
5. Select the proper integration broker in the **Broker Type** drop-down menu—`ICS`, `WMQI` or `WAS`.

   Note: Select `WMQI` if your broker is any WebSphere message broker.
6. If you selected `ICS` as your broker type in step 5, do the following as well:
   a. Type the name of the InterChange Server Express instance in the **Server** field.

      Be sure to type the name precisely; it is case-sensitive and Test Connector will not be able to communicate with InterChange Server Express if the name is not correct.
   b. Type the password for the `admin` user account in the **Password** field. The default password is `null`.

7. Click **OK** to close the “New Profile” window.

   The “Connector Profile” window displays the name of the connector in the **Connector** column, the name of the InterChange Server Express instance in the **Server** column, and the path and name of the connector configuration file in the **Configuration File** column.
8. Click **OK** to close the “Connector Profile” window.

Editing a profile

Follow the steps below to make changes to an existing connector profile:

1. Select **File > Create/Select Profile** from the menu bar of Test Connector or use the keyboard shortcut **Ctrl+N** to display the Connector Profile window.
2. In the “Connector Profile” window select the profile you want to edit and then select **Edit > Edit Profile** from the menu bar.
3. Type new values in the fields of the “New Profile” window and use the **Browse** button to change the configuration file as necessary to make your edits.
4. Click **OK** to close the “New Profile” window.

Deleting a profile

Do the following to delete a connector profile:

1. Select **File > Create/Select Profile** from the menu bar of Test Connector or use the keyboard shortcut **Ctrl+N** to display the “Connector Profile” window.
2. In the “Connector Profile” window, select the profile you want to delete and then select **Edit > Delete Profile** from the menu bar.

Emulating a connector

After creating a profile for a connector, you may use that profile to connect Test Connector to the agent. Once you connect Test Connector to the agent, Test Connector begins emulating the connector defined in the selected profile.

To connect Test Connector to the agent, do the following:

1. Select **File > Create/Select Profile** from the menu bar of Test Connector.
2. In the “Connector Profile” window, select the name of the connector whose profile you want to open.
3. Click **OK**.
4. Select **File > Connect** from the menu bar.

   Test Connector displays messages in the “Output” pane as it attempts to emulate the connector. When it finishes connecting, it displays a message indicating that it is “ready” in the “Output” pane and populates the **BOType** list in the “Supported Business Objects” pane.

Working with business objects

To test whether a business process interface has been developed correctly, you need to verify that business objects can be successfully exchanged and processed. This section describes how to:

- Create, modify, delete, and save business object test data
- Compare the attribute values of business objects to easily and quickly view changes made during processing
- Send and receive business objects

Working with request business objects

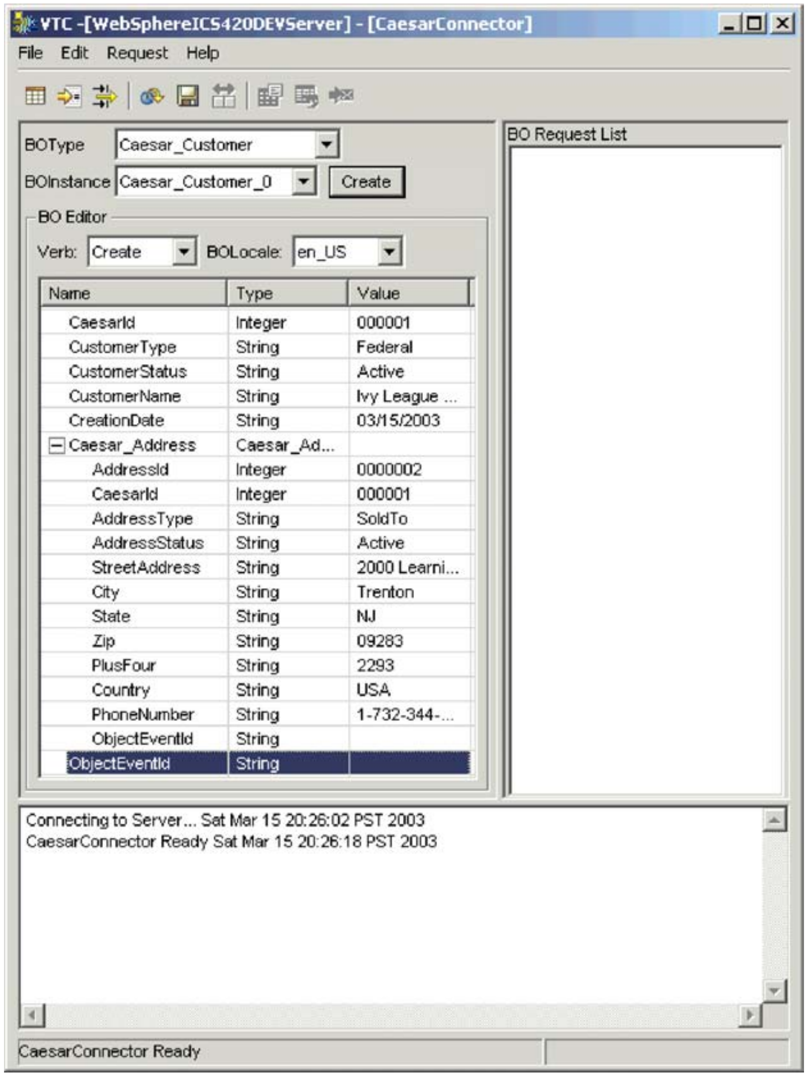
Request business objects are those that you send from Test Connector when it is emulating a connector that is the source of the events that trigger an interface. Working with request business objects consists of creating a business object instance, populating it with data, and sending the request.

Creating request business objects

To create a new business object in Test Connector, do the following:

1. In the "Supported Business Objects" pane, select the name of the business object you want to create from the **BOType** drop-down menu.
2. Click **Create** next to the **BOInstance** field.
3. When presented with the "New Instance" dialog, type a name for the instance in the **Enter Name** field.
4. Select the desired verb from the **Verb** drop-down menu.
5. Select the desired locale from the the **BOLocale** drop-down menu.
6. Provide values for the simple attributes and child business objects within the top-level object, as described in "Setting values for business object attributes" on page 196.

   Figure 53 on page 194 shows a business object named `Caesar_Customer` with the `Create` verb, the `en_US` locale, values specified for each of its simple attributes, and a single instance of the `Caesar_Address` child business object.



BOType: Caesar_Customer
BOInstance: Caesar_Customer_0
Verb: Create
BOLocale: en_US

Name	Type	Value
CaesarId	Integer	000001
CustomerType	String	Federal
CustomerStatus	String	Active
CustomerName	String	Ivy League ...
CreationDate	String	03/15/2003
Caesar_Address	Caesar_Ad...	
AddressId	Integer	0000002
CaesarId	Integer	000001
AddressType	String	SoldTo
AddressStatus	String	Active
StreetAddress	String	2000 Learni...
City	String	Trenton
State	String	NJ
Zip	String	09283
PlusFour	String	2293
Country	String	USA
PhoneNumber	String	1-732-344-...
ObjectEventId	String	
ObjectEventId	String	

```
Connecting to Server... Sat Mar 15 20:26:02 PST 2003
CaesarConnector Ready Sat Mar 15 20:26:18 PST 2003
```

CaesarConnector Ready

Figure 53. Populating a business object with data

7. Click **OK**.

Sending request business objects

Once you have created or loaded a business object and specified values for its attributes, you have several ways to send the business object as a request to InterChange Server Express.

Sending request business objects asynchronously: When a source connector sends a request business object in asynchronous mode, it does not expect to get back a response business object. Once the request business object is dispatched, the source connector’s role in the transaction is finished. The response business object is typically processed by InterChange Server Express. The default mode for Test Connector is asynchronous.

To send a business object asynchronously, do the following:

1. Select **Request > Mode >Asynchronous** from the menu bar.

   Note: Test Connector operates in “Asynchronous” mode by default, so you only have to perform this step if you previously were sending synchronous requests from the connector. Furthermore, you do not have to set the mode before sending each request.

2. Select **Request > Send** from the menu bar.

If the broker specified in the connector definition is InterChange Server Express then the business object request is sent to the server for processing.

If the broker specified in the connector definition is one of the supported message brokers or WebSphere Application Server then the business object is placed on the queue specified in the `RequestQueue` standard property.

Sending request business objects synchronously: When a source connector sends a request business object synchronously, it expects to get back a response business object from the integration broker after any destination applications have processed the request. In synchronous mode, Test Connector puts the response business object on the queue specified by the source connector's Synchronous Request Queue property. The default mode for Test Connector is asynchronous.

1. Set Test Connector to synchronous mode by selecting **Request > Mode > Synchronous** from the menu bar.
2. Select **Request > Send** from the menu bar.
3. If the broker specified in the connector definition is InterChange Server Express then the "Select Collaboration" dialog is displayed. Select the collaboration to which the business object should be sent from the **Collaboration** drop-down menu and click **OK**.

The business object request is sent to the configured port of the collaboration object chosen for processing.

Sending request business objects in batch mode: In batch mode, Test Connector lets you specify the number of instances of a particular business object you want to send, as well as one attribute in the top-level object —a primary key attribute, for example—that you want set to a unique value for each instance. Test Connector copies the business object as many times as you have specified, incrementing the value of the single attribute you specified, and sends each business object. This option allows you to create a large number of business objects quickly and easily.

If the selected attribute is a key field that participates in dynamic cross-referencing as part of an identity relationship, then you must guarantee that the initial value and all those that follow it are unique. Otherwise, the cross-referencing logic will fail, causing the request business objects to fail.

To ensure that the values are unique, you can use Relationship Manager or execute SQL statements against the table for the relationship participant as follows.

- Determine the highest current value for the participant and set the Initial Value field to an even higher value. The first business object instance in the batch and all those that follow will then be unique.
- Delete the existing table entries for the participant, thus guaranteeing that no entries have the same attribute value as any of the batch business objects.

To send business objects in batch mode, do the following:

1. Select the name of the business object you would like to send from the **BOType** drop-down menu.
2. Select **Request > Send Batch** from the menu bar.

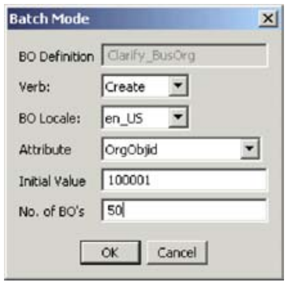
3. In the “Batch Mode” window, select the desired verb from the **Verb** drop-down menu.
4. Select the desired locale from the the **BOLocale** drop-down menu.
5. Select from the **Attribute** list the attribute in the top-level business object that you want incremented with each business object request in the batch.

   The selected attribute should typically be an attribute that uniquely identifies the business object, such as a primary key.
6. In the **Initial Value** field, type the starting value for the attribute to be incremented.
7. In the **No. of BO’s** field, type the number of business object instances you want generated and sent.
8. Click **OK**.

   Test Connector generates the number of business objects you specified, all identical with the exception of the one specified attribute, whose value is incremented for each instance.

   The business object request is sent to InterChange Server Express for processing.

Figure 54 shows a batch mode configuration in which:
- Fifty business objects are to be sent.
- The value of the attribute `OrgObjid` is to be incremented.
- The starting value for the attribute is 100001.


Batch Mode

BO Definition	Clarify_BusOrg
Verb:	Create
BO Locale:	en_US
Attribute	OrgObjid
Initial Value	100001
No. of BO's	50

Figure 54. The Batch Mode Window

Setting values for business object attributes

The following sections describe the various ways you can set the values of simple and compound attributes in a business object instance:
- “Setting values for simple attributes”
- “Adding child business objects”
- “Removing child business objects” on page 197
- “Setting the verb of a child business object” on page 197

Setting values for simple attributes

To provide a value for a simple attribute, click its cell in the **Value** column and enter a value.

Adding child business objects

To add an instance of a child business object, right-click the attribute that represents the child object and select **Add Instance** from the context menu.

A plus sign (+) is added next to the attribute that represents child business object to show that there is at least one child business object instance. If you expand the child object attribute, numbered entries are displayed for each instance. The individual instances also have plus signs (+) next to them, so you can expand them and set values for their attributes.

To add more child business object instances, right-click the attribute that represents the child object and select **Add Instance** from the context menu.

Note: If the **Card** property of the attribute that references the child business object is set to the value 1 (indicating it is of single-cardinality), then you will only be able to add one instance of the child object.

Removing child business objects

To remove an instance of a child business object, right-click the instance and select **Remove Instance** from the context menu.

To remove all instances of a child business object, right-click the attribute that represents the child business object and select **Delete All Instances** from the context menu.

Setting the verb of a child business object

You can set the verb of a child business object to test the effect that value has on the business process. This can be helpful when you are troubleshooting logic that involves the cross-referencing of child objects.

To set the verb of a child business object instance, right-click it and choose **Set Verb** from the context menu. When presented with the “Select Verb” prompt, selected the desired verb and click **OK**.

Using the Response BO toolbar

You can edit the attributes of a business object received by a destination connector before you send it as a response. The toolbar of the “Response BO” dialog that you use when doing so has several toolbar buttons that can be used to set the values of the business object. For more information, see “Editing response business objects” on page 198.

Saving a business object

You can save a business object in Test Connector so that it can be used for later tests, shared with technical support (to help troubleshoot problems), or used as response data. You can save any business object, including ones that you have created and ones that appear as requests in the Test Connector window of a destination connector. By default, business objects are saved to a file with a business object extension (`.bo`).

It is recommended that you create a directory or directory structure specifically for test data files, with subdirectories dedicated to each interface or to each connector, as appropriate. This organization makes the necessary files are easy to locate and makes testing more efficient. Furthermore, it is recommended that you give the test data file for a business object the same name as the business object definition itself.

Saving request business objects

Do the following to save a business object instance that you have created as a request:

1. Select the business object you want to save.

2. From the menu bar, select **Edit > Save BO**.
3. Navigate to the desired directory and specify a name for the file in the **File name** field.
4. Click **Save**.

Saving response business object

Do the following to save a business object instance that has been received by a destination instance of Test Connector and will be sent as a response:

1. Select the business object instance in the "BO Request List" pane.
2. Select **Request > Edit Response** from the menu bar.
3. Click **Save BO**.
4. Navigate to the desired directory and specify a name for the file in the **File name** field.
5. Click **Save**.

Loading a business object

To load a business object that has been saved to a file, do the following:

1. Select **Edit > Load BO** from the menu bar of Test Connector.
2. Navigate to the business object test data file and open it.
3. When presented with the "New Instance" dialog, type a name for the instance in the **Enter Name** field.
4. Click **OK**.

Deleting a business object

To delete a business object from Test Connector, select **Edit > Delete BO** from the menu bar.

Note: This action only removes the business object from the Test Connector. It does not remove the connector's support for the business object definition.

Accepting a request business object

When you send a business object as a request, the business object appears in the "BO Request List" pane of any Test Connector instances that are emulating destination connectors in the interface, provided that the transaction did not fail.

After you have accepted the request business object, you can edit it if necessary as described in "Editing response business objects."

Working with response business objects

Response business objects are those that you send from Test Connector when it is emulating a connector that is the recipient of business object requests in an interface. Working with request business objects consists of editing the values in the business object instance and sending the response back to the broker.

Editing response business objects

When you receive a business object request in a destination instance of Test Connector, you commonly want to edit the values of the attributes. For instance, you will want to provide unique values for primary key attributes that participate in relationships, or you will want to modify the value of other attributes to test map or collaboration logic that responds differently depending on the exact values in the business object. Do the following to set the values of business object attributes :

1. Select the business object instance in the “BO Request List” pane.
2. Select **Request > Edit Response** from the menu bar.
3. Do the following to edit the attributes of the business object:
   - Use one of the techniques described in “Setting values for business object attributes” on page 196 to modify the values of the business object attributes.
   - Click **Reset BO to default** to set the values of the business object attributes to their default values as specified in the business object definition.
   - Click **Clear BO values** to clear the values of all the attributes in the business object.
   - Click **Load BO** to populate the attributes of the business object with test data from a file.

     The ability to load saved data into a business object request is very useful in situations where you have to populate a response business object with data before sending it as a reply. Instead of manually typing a value for each attribute that requires response data, you can type the values once, save the business object (as described in “Saving a business object” on page 197), and then load the saved data on subsequent tests.

Sending a response business object

After you accept a request business object, edit the business object, if needed, and send it back as a reply.

Table 23 lists Test Connector’s reply options and shows their corresponding connector return codes for both C++ and Java connectors.

Table 23. Test Connector reply types and connector return codes.

Test Connector reply type	**C++ connector return code**	**Java connector return code**
Success	BON_SUCCESS	SUCCESS
Fail	BON_FAIL	FAIL
Multiple Hits	BON_MULTIPLE_HITS	MULTIPLE_HITS
Retrieve By Content Fail	BON_FAIL_RETRIEVE_BY_CONTENT	RETRIEVEBYCONTENT_FAILED
Not Found	BON_BO_DOES_NOT_EXIST	BO_DOES_NOT_EXIST
Value Duplicate	BON_VALDUPES	VALDUPES

To reply to a request business object, do the following:

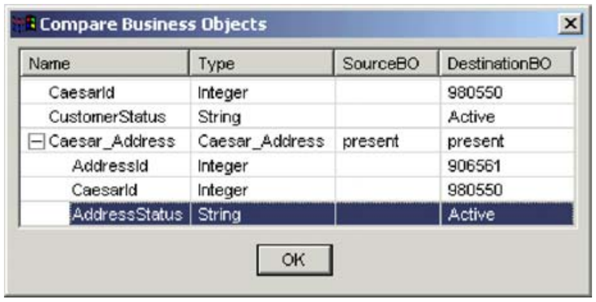
1. Select the business object in the “BO Request List” pane.
2. From the menu bar, select **Request > Reply**.
3. Select an item from the **Reply** submenu.

Comparing business object instances

Test Connector can compare two business objects of the same type and display the attributes that differ in value. You can use this function to view changes to a business object at different points in the execution of a transaction (for instance, you could compare a business object that has been sent to the integration broker with the same business object after the integration broker has updated it). To compare two business objects, do the following:

1. Create a request business object instance by following the instructions in either "Creating request business objects" on page 193 or "Loading a business object" on page 198.
2. Select the response business object instance in the "BO Request List" pane that you would like to compare the request business object instance to.
3. From the menu bar, select **Edit > Compare BO's**.

   Test Connector opens the "Compare Business Objects" window with a table that displays the attributes which have different values in the two business objects. Figure 55 on page 200 shows a comparison between two business object instances.



Name	Type	SourceBO	DestinationBO
CaesarId	Integer		980550
CustomerStatus	String		Active
⊟ Caesar_Address	Caesar_Address	present	present
AddressId	Integer		906561
CaesarId	Integer		980550
AddressStatus	String		Active

Figure 55. The Business Object Comparison Window.

4. Click **OK** to close the window.

Chapter 12. Using Integrated Test Environment

Integrated Test Environment is a workbench perspective that enables you to conveniently test an entire business process interface. An interface centers around a collaboration object, so a test performed in Integrated Test Environment centers around a collaboration object as well.

To perform a test in Integrated Test Environment, you must do a number of tasks such as register InterChange Server Express 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.

Most of the sections in this chapter describe tasks you must perform as part of the greater task of performing a test, or describe particular elements of the Integrated Test Environment interface.

The section “Performing a test using Integrated Test Environment” on page 255 describes the workflow you will typically follow to perform a single test of a single interface. It is a list of references to the sections in this chapter that document the specific sub-tasks and interface elements, and it provides a task-oriented structure for using Integrated Test Environment, whereas the other sections are more interface-oriented. In situations where you might perform a task in one of several ways it references the way that is most effective and efficient in most situations. To use this chapter as best as possible, it is recommended that you follow the workflow described in “Performing a test using Integrated Test Environment” on page 255, but substitute particular sub-tasks from other sections where the recommended approach is not appropriate for your environment or situation.

Registering InterChange Server Express as a test server

To test an interface using Integrated Test Environment, you must register the InterChange Server Express instance so that it starts in test mode. This allows Integrated Test Environment to do things such as the following:

- Locate the batch file that starts InterChange Server Express, so that it can be started from within the Integrated Test Environment interface.
- Re-direct the InterChange Server Express logging and tracing information to a view within Integrated Test Environment, rather than to the console.

To register InterChange Server Express in test mode, see “Registering an InterChange Server Express instance” on page 43 and perform each of the steps necessary to register the server instance as a local test server. Integrated Test Environment can only work with a server instance that is installed on the same machine on which the workbench is installed.

Starting Integrated Test Environment

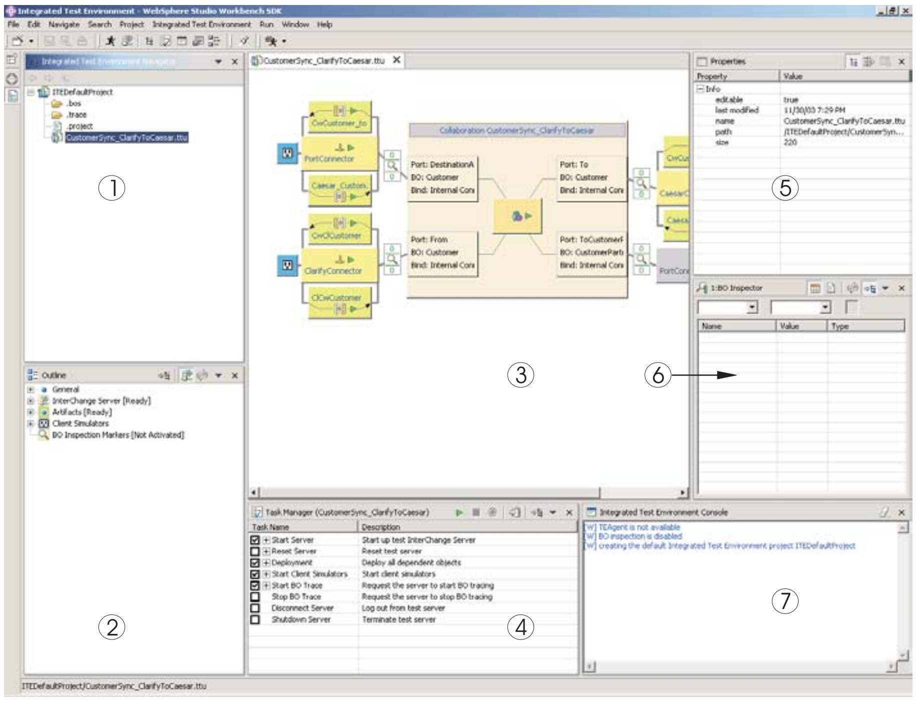
To start Integrated Test Environment, do the following:

1. Select **Start > Programs > IBM WebSphere Business Integration Express>Toolset Express> Administrative > System Manager**.
2. Select **Window > Open Perspective > Other** from the menu bar.
3. Select `Integrated Test Environment` from the list of perspectives and then click **OK**.

   The workbench starts and appears. Figure 10 on page 40 shows the Integrated Test Environment perspective and "Integrated Test Environment interface" describes the interface and its elements.

Integrated Test Environment interface

The Integrated Test Environment perspective has several views and one editor in the default configuration it opens with. Figure 10 on page 40 shows the default Integrated Test Environment perspective.



Integrated Test Environment - WebSphere Studio Workbench SDK
File Edit Navigate Search Project Integrated Test Environment Run Window Help
Integrated Test Environment Navigator
ITEDefaultProject
.bos
.trace
.project
CustomerSync_ClarifyToCaesar.ttu
CustomerSync_ClarifyToCaesar.ttu
Collaboration CustomerSync_ClarifyToCaesar
CwCustomer_to
PortConnector
Caesar_Custom
CwClCustomer
ClarifyConnector
ClCwCustomer
Port: DestinationA
BO: Customer
Bind: Internal Con
Port: To
BO: Customer
Bind: Internal Con
Port: From
BO: Customer
Bind: Internal Con
Port: ToCustomerF
BO: CustomerPart
Bind: Internal Con
PortCon
Properties
1:BO Inspector
Name
Value
Type
Outline
General
InterChange Server [Ready]
Artifacts [Ready]
Client Simulators
BO Inspection Markers [Not Activated]
Task Manager (CustomerSync_ClarifyToCaesar)
Integrated Test Environment Console
ITEDefaultProject/CustomerSync_ClarifyToCaesar.ttu
1
2
3
4
5
6
7

Property	Value
Info	
editable	true
last modified	11/30/03 7:29 PM
name	CustomerSync_ClarifyToCaesar.ttu
path	/ITEDefaultProject/CustomerSyn...
size	220

Task Name	Description
☑ Start Server	Start up test InterChange Server
☐ Reset Server	Reset test server
☑ Deployment	Deploy all dependent objects
☑ Start Client Simulators	Start client simulators
☑ Start BO Trace	Request the server to start BO tracing
☐ Stop BO Trace	Request the server to stop BO tracing
☐ Disconnect Server	Log out from test server
☐ Shutdown Server	Terminate test server

```
[W] TEAgent is not available
[W] BO inspection is disabled
[W] creating the default Integrated Test Environment project ITEDefaultProject
```

Figure 56. Integrated Test Environment perspective

Table 24 describes the interface elements of the Integrated Test Environment perspective, identified by the numbers in Figure 56:

Table 24. Integrated Test Environment perspective interface elements

Interface element number	Interface element name
1	"Integrated Test Environment Navigator view" on page 203

Table 24. Integrated Test Environment perspective interface elements (continued)

Interface element number	Interface element name
2	"Outline view"
3	"Test Unit editor"
4	"Task Manager view"
5	"Properties view"
6	"BO Inspector view" on page 204
7	"Integrated Test Environment Console view" on page 204

Integrated Test Environment Navigator view

This view lists the projects and test units you create. For more information on working with test projects and test units, see "Working with test projects and units" on page 209.

Outline view

This view lists the components and elements that comprise the interface and are displayed in the "Test Unit editor." As you select items in this view, their properties are displayed in the "Properties view." This view also allows you to determine if the interface dependencies have been resolved, and if other required conditions have been satisfied prior to testing.

This is a default workbench view.

Test Unit editor

This editor provides a graphical representation of the interface. It allows you to do things such as the following:

- Deploy components, as described in "Managing the repository using the Test Unit view" on page 231.
- Manage the states of components, as described in "Managing component states using the Test Unit view" on page 231.
- View business object data, as described in "Using the BO Inspector view" on page 247.

Task Manager view

The Task Manager view allows you execute tasks that must be performed as part of a test, such as starting InterChange Server Express and starting Client Simulator views. For more information on this view, see "Using the Task Manager view" on page 220.

Properties view

You can select items in the ITE Navigator, Outline, and Test Unit views to display properties of the items in this view. The property information is largely useless to you and you will probably want to overlay this view with a Client Simulator view or BO Inspector view.

For more information on the Client Simulator view, see "Using the Client Simulator view" on page 234.

For more information on the BO Inspector view, see "Using the BO Inspector view" on page 247.

This is a default workbench view.

BO Inspector view

This view allows you to view business object data. For more information, see "Using the BO Inspector view" on page 247.

Integrated Test Environment Console view

The Integrated Test Environment Console displays information about the testing tasks. For instance, entries are written to it when you start the InterChange Server Express instance and start Client Simulator views. Use the information in it to troubleshoot the testing process itself.

If you start InterChange Server Express from within Integrated Test Environment, then a second tab named InterChange Server Express Console is added to this view. The InterChange Server Express Console displays the logging and tracing information of InterChange Server Express. Use the information displayed in it to troubleshoot the interface you are testing.

This is a default workbench view.

Selecting a server configuration

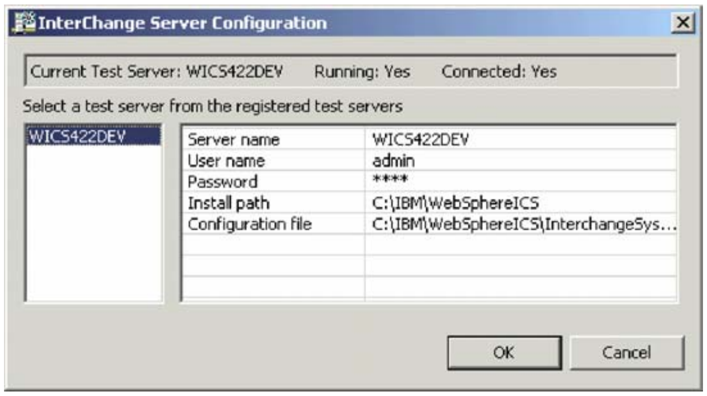
If you have registered two InterChange Server Express instances as local test servers then you must select the server configuration you want to work with in Integrated Test Environment. Do the following to select a server configuration:

Note: The InterChange Server Express must be registered in the System Manager perspective before performing these steps. For more information, see "Registering an InterChange Server Express instance" on page 43.

1. Do one of the following to display the InterChange Server Express Configuration window:
   - Select **Integrated Test Environment > Test Server Configuration** from the menu bar of Integrated Test Environment.
   - Expand the "InterChange Server" node in the Outline view, then right-click the "Registration" node and choose **Show test server configuration** from the context menu
2. At the InterChange Server Express Configuration window, select the test server you want to use from the pane at the left.

   When you select a server, the configuration information for it is presented in the pane on the right-hand side of the window. The read-only text field at the top of the window indicates whether the server is running, and if Integrated Test Environment is connected to it.

   Figure 57 on page 205 shows the InterChange Server Express Configuration window.


InterChange Server Configuration

Current Test Server: WICS422DEV Running: Yes Connected: Yes

Select a test server from the registered test servers

WICS422DEV

Server name	WICS422DEV
User name	admin
Password	****
Install path	C:\IBM\WebSphereICS
Configuration file	C:\IBM\WebSphereICS\InterchangeSys...

Figure 57. Test Server Configuration window

3. Click **OK**.

Note: Although you can register up to two InterChange Server Express instances as test servers, they must both be installed on the computer on which Integrated Test Environment is running. You cannot use Integrated Test Environment to test interfaces running on a server that is installed on a computer other than the one on which Integrated Test Environment is installed.

Configuring RMI settings

Integrated Test Environment uses a software agent that communicates with InterChange Server Express to perform tests. This agent must communicate with InterChange Server Express and the tools through a port using Remote Method Invocation (RMI). Integrated Test Environment also requires a block of nine contiguous ports over which to communicate with each connector that is emulated in the interface being tested.

You must configure your environment so that there are ports available for these communication requirements. The default RMI port is 1099, and this is the port that the Integrated Test Environment agent is configured to use by default. Integrated Test Environment is configured by default to use ports 1100 and up to communicate with the connectors being emulated in the interface.

Other programs installed on your system may be configured to use port 1099, however, so you must determine which ports you can use. After determining which ports you can use you must configure settings in Integrated Test Environment to use the desired port as well as set a property in the `start_server.bat` batch file.

Determining which ports to use

Before attempting to use Integrated Test Environment, you should use the techniques described in the following sections to determine if the ports it is configured to use by default are available or not.

Using the netstat command

You can execute the `netstat` command at a command line interface in Windows to generate a list of the ports that are currently in use. It is recommended that you use the -a option to list all of the ports in use, rather than just those used for TCP/IP. You might also want to use the -n option, which orders the ports numerically.

The following example shows the use of the netstat command and a portion of the returned output:

```
C:\>netstat -a -n
Active Connections
Proto  Local Address          Foreign Address        State
TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
TCP    0.0.0.0:371            0.0.0.0:0              LISTENING
TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
TCP    0.0.0.0:1029           0.0.0.0:0              LISTENING
TCP    0.0.0.0:1030           0.0.0.0:0              LISTENING
```

The first entry shows that port 80 is being used and the last line shows that port 1030 is being used. Examine the lines to determine if ports 1099 through 1108 are being used. It is recommended that you copy the output into a text file where you can search for the port numbers.

If ports 1099 through 1108 are in use already then you must change the settings as described in "Setting the RMI port in start_server.bat" on page 208 and "Setting the RMI port in start_server.bat" on page 208.

Checking the services file

Besides using the netstat command to determine which ports are currently in use, you can also check the services file to determine which ports are reserved. Do the following:

1. Use a text editor to open the file named services in the Winnt\System32\drivers\etc directory in your file system.
2. Search the entries in the file to determine if there are any services listed that are associated with ports 1099 through 1108.

   If you find any such entries, you might experience complications when trying to use Integrated Test Environment with the default configuration. Change the settings as described in "Setting the RMI port in start_server.bat" on page 208 and "Setting the RMI port in start_server.bat" on page 208 if so.

Setting the RMI port in the Integrated Test Environment preferences

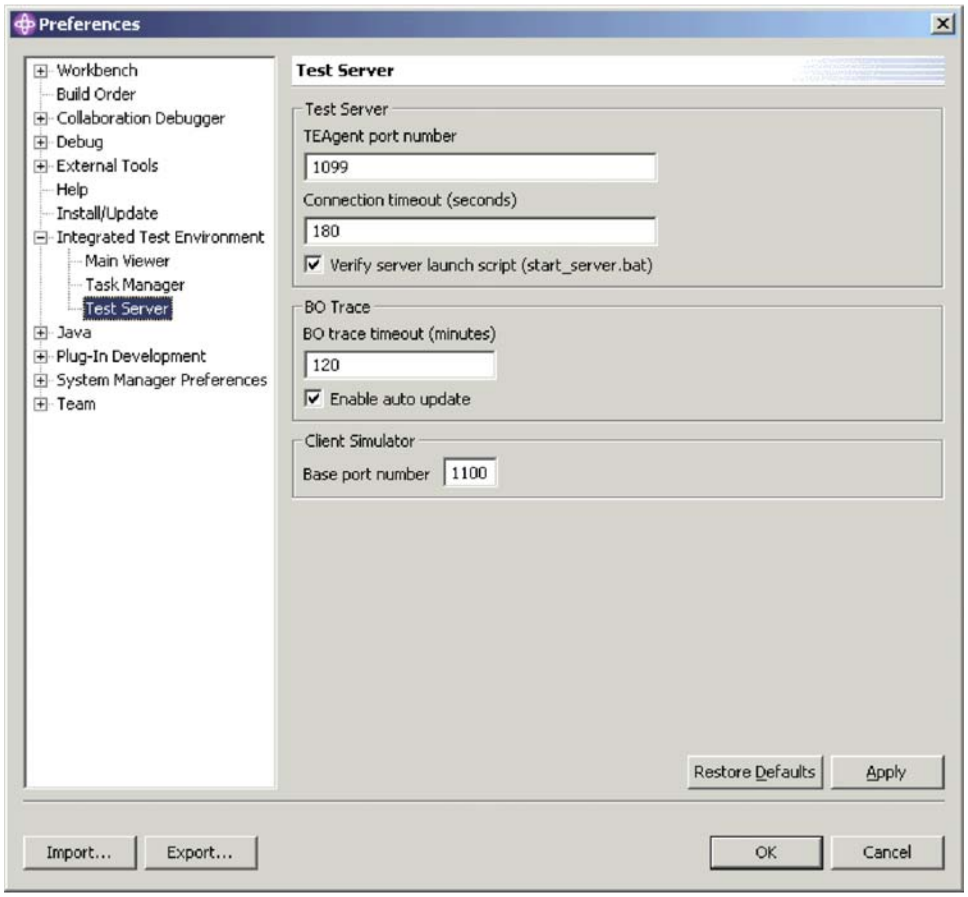
If you follow the instructions in "Determining which ports to use" on page 205 and determine that the ports used by Integrated Test Environment by default are already in use then you must set the preferences for Integrated Test Environment to use ports that are available.

Integrated Test Environment requires a single port for the test environment agent to communicate with the server. It also requires nine ports to communicate with up to nine connector agents that can be emulated during a test. You configure the port number that Integrated Test Environment will use for the first of the nine possible connector agents, and then it uses the eight ports above that number for the others, so you must ensure that there is a block of nine contiguous ports available. If you do not have nine connectors that require emulation in the interface then you can use a smaller block of ports. Ideally, you should find a block of ten available ports so that all of the ports related to the Integrated Test Environment are contiguous.

Do the following to configure the port usage for Integrated Test Environment:

1. Select **Window > Preferences** from the menu bar of the workbench.
2. Expand **Integrated Test Environment**.
3. Select **Test Server**.

   Figure 58 shows the Test Server preferences.



Test Server

Test Server

TEAgent port number: 1099

Connection timeout (seconds): 180

☑ Verify server launch script (start_server.bat)

BO Trace

BO trace timeout (minutes): 120

☑ Enable auto update

Client Simulator

Base port number: 1100

Figure 58. Test Server preferences

4. In the “Test Server” pane, enter the port number for the test environment agent in the **TEAgent port number** field.

   The default value is `1099`.
5. In the “Client Simulator” pane, enter the port number that Integrated Test Environment should use for the first of the nine possible connector agents or access clients in the **Base port number** field.

   The default value is `1100`. Integrated Test Environment uses the value in the **Base port number** field for the first connector agent that must be emulated, and use the subsequent port numbers for other connector agents.
6. Follow the instructions in “Enabling Integrated Test Environment to create a custom batch file” on page 209 to have Integrated Test Environment use a custom batch file to which it adds the necessary port configuration information specified in the previous steps.
7. Click **OK**.

Setting the RMI port in start_server.bat

If you do not configure the Integrated Test Environment preferences to use a custom batch file to start the server then you must add the RMI port configuration information to the `start_server.bat` batch file yourself.

Even if you plan to use the default RMI port number of 1099, you must add the property to the batch file and set it equal to the default value. Do the following to do so:

1. Use a text editor to open the batch file named `start_server.bat` in the `bin` directory of the product installation.
2. Add the **-DTEAgent** Java system property to the line in the batch file where the Java program is executed to start the server, and set it equal to the value in the **TEAgent port number** field in the Integrated Test Environment preferences. The following example shows how the batch file would appear if the **-DTEAgent** property were set to the default RMI port number of `1099`:

   ```
   %CWJAVA% -Djava.ext.dirs=%JRE_EXT_DIRS%;"%MQ_LIB%";"%DB2_LIB%"
   -Duser.home="%CROSSWORLDS%" -mx%CW_MEM_HEAP%m -DTEAgent=1099
   -DCW_MEMORY_MAX=%CW_MEM_HEAP% %ORB_PROPERTY% -classpath %JCLASSES%
   ServerWrapper -s%SERVERNAME% %2 %3
   ```

   Note: You might want to make the modifications described in "Configuring InterChange Server Express to start in design mode" at this point, as those modifications also involve the `start_server.bat` batch file.
3. Save and close the file.

Configuring InterChange Server Express to start in design mode

If you plan to deploy the components you will test through Integrated Test Environment then you must modify the `start_server.bat` batch file so that the InterChange Server Express instance starts in design mode. This is because Integrated Test Environment may deploy the components in such an order that dependencies are not immediately resolved. For the deployment to succeed in such a situation, the server must run in design mode.

As described in "InterChange Server modes" on page 45, you typically start InterChange Server Express in design mode by passing the `-design` option at the command line or in the **Target** field of the shortcut. When you use Integrated Test Environment, you start the server through the Integrated Test Environment perspective, and it does not use the shortcut, so even if you modify your InterChange Server Express shortcut to use the `-design` option the server will not start in design mode.

To start the server in design mode when using Integrated Test Environment, you can either add the `-design` option to the `start_server.bat` batch file, or have Integrated Test Environment use a temporary batch file to which it adds the option itself. Follow the instructions in one of the following sections depending on your preference:

Manually editing the start_server.bat file

Do the following to edit the `start_server.bat` batch file yourself and add the `-design` option to it:

1. Use a text editor to open the batch file named `start_server.bat` in the `bin` directory of the product installation.

2. Add the **-design** option to the line in the batch file where the Java program is executed to start the server. You should add the **-design** option after the **-s%SERVERNAME%** option. The following example shows how the batch file should appear:
   ```
   %CWJAVA% -Djava.ext.dirs=%JRE_EXT_DIRS%;"%MQ_LIB%";"%DB2_LIB%"
   -Duser.home="%CROSSWORLDS%" -mx%CW_MEM_HEAP%m -DTEAgent=1200
   -DCW_MEMORY_MAX=%CW_MEM_HEAP% %ORB_PROPERTY% -classpath %JCLASSES%
   ServerWrapper -s%SERVERNAME% -design %2 %3
   ```
3. Save and close the file.

Enabling Integrated Test Environment to create a custom batch file

You can configure Integrated Test Environment to start InterChange Server Express in design mode automatically. With this approach, Integrated Test Environment reads in the `start_server.bat` file and determines if the `-design` option has been specified properly or not. It then creates a file named `start_server_ite_tmp.bat` as a copy of the `start_server.bat` file and adds the `-design` option if it has not been properly specified. If you start the server from within Integrated Test Environment then it uses this temporary file.

Do the following to configure the Integrated Test Environment preferences to create a temporary batch file:
1. Select **Window > Preferences** from the menu bar of the workbench.
2. Expand **Integrated Test Environment**.
3. Select **Test Server**.

   Figure 58 on page 207 shows the Test Server preferences.
4. Enable the **Verify server launch script (start_server.bat)** checkbox to have Integrated Test Environment use a custom batch file to which it adds the necessary configuration information.

   If you do not plan to use the custom batch file that Integrated Test Environment can create, leave the **Verify server launch script (start_server.bat)** checkbox disabled and be sure to follow the instructions in "Setting the RMI port in start_server.bat" on page 208 to add the configuration information yourself.
5. Click **OK**.

Important: For this approach to work, you must start the server from Integrated Test Environment. If you start the server using the program group shortcut or the command line then this approach will not work, because those startup techniques do not use the temporary startup script.

Working with test projects and units

All resources in an Eclipse-based platform are contained in projects, as described in "Projects" on page 38.. You must create integration component library projects to contain integration components, you must create user projects to contain shortcuts to components, and you must create **test projects** in the Integrated Test Environment to contain definitions for tests.

A test unit is a resource created in the workbench that defines a test. Integrated Test Environment is designed to let you test an entire interface, and an interface typically centers around a collaboration object, as described in "Interfaces" on page 21

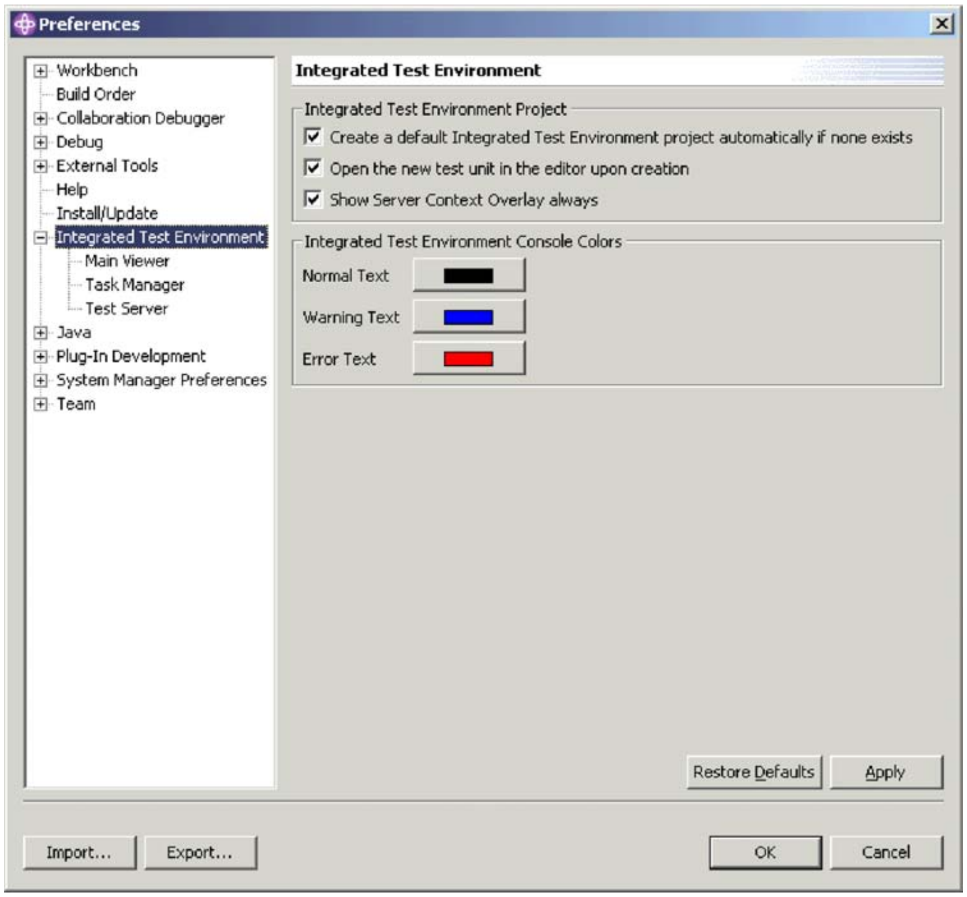
21, so a test unit centers around a collaboration object as well. You can define all the test units you want to run in a single test project, or can create multiple test projects and add test units to them as appropriate to organize them according to your preference.

Configuring test project preferences

Before creating either test projects or test units you should configure the preferences related to both. Do the following to configure your preferences related to test projects and units:

1. Select **Window > Preferences** from the menu bar of the workbench.
2. Select **Integrated Test Environment**.

   Figure 59 shows the Integrated Test Environment preferences.



Integrated Test Environment

Integrated Test Environment Project
- ✓ Create a default Integrated Test Environment project automatically if none exists
- ✓ Open the new test unit in the editor upon creation
- ✓ Show Server Context Overlay always

Integrated Test Environment Console Colors

Normal Text	⬛
Warning Text	🟦
Error Text	🟥

Figure 59. Integrated Test Environment preferences

3. Enable the **Create a default Integrated Test Environment project automatically if none exists** checkbox if you want Integrated Test Environment to create a default project named `DefaultITEProject` if you are creating a test unit and no project already exists.
4. Enable the **Open the new test unit in the editor upon creation** checkbox if you want to automatically open a test unit definition in the Test Unit Editor after you create it.

   For more information about opening test units if you decide not to enable this option, see "Opening a test unit" on page 214.

5. Enable the **Show Server Context Overlay always** checkbox if you want to always have the Server Context Overlay enabled.

   For more information about the Server Context Overlay, see "Using the Server Context Overlay" on page 229.
6. Click **OK**.

Creating a test project

Do the following to create a test project to store the individual test units you will create:

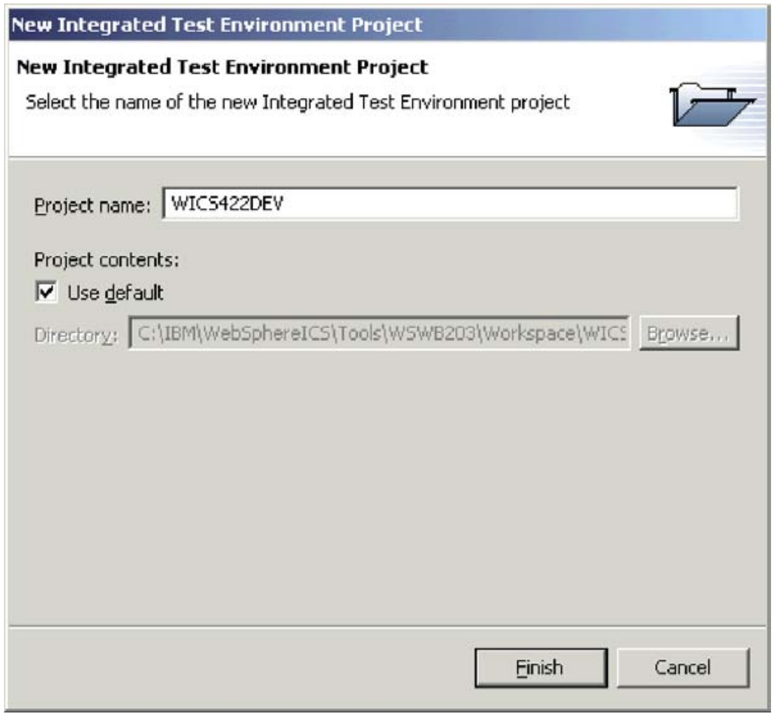
1. Select **File > New > Integrated Test Environment Project** from the menu bar.
2. At the "New Integrated Test Environment Project" screen, type a name for the test project in the **Project name** field.

   Project names can only contain alphanumeric characters and underscores, and must be specified in English.
3. To have the folder for the library created in the default location (your workspace) and with a name identical to the name specified for the library, leave the **Use default location** checkbox enabled.

   If you want to specify the name and location of the library folder, do the following:
   a. Clear the **Use default location** checkbox.
   b. Type the full path and name of the directory that you want to use for the library in the **Location** field, or click **Browse** to select an existing directory.

   Note: There is no way to create the folder for a library in the path of the workspace other than to let System Manager do it by use of the **Use default location** checkbox.

   Figure 60 on page 212 shows the "New Integrated Test Environment Project" wizard.


New Integrated Test Environment Project
New Integrated Test Environment Project
Select the name of the new Integrated Test Environment project
Finish
Cancel
Browse...

Project name: WICS422DEV

Project contents:

☑ Use default

Directory: C:\IBM\WebSphereICS\Tools\WSWB203\Workspace\WICS

Figure 60. Creating a new WBI ITE project

4. Click **Finish**.

   The project is created and a folder is added for it to the “Integrated Test Environment Navigator” view.

Creating test units

A test unit is the workbench resource that contains the configuration information for a test you want to perform. You can either create a test unit from within Integrated Test Environment, or from within System Manager.

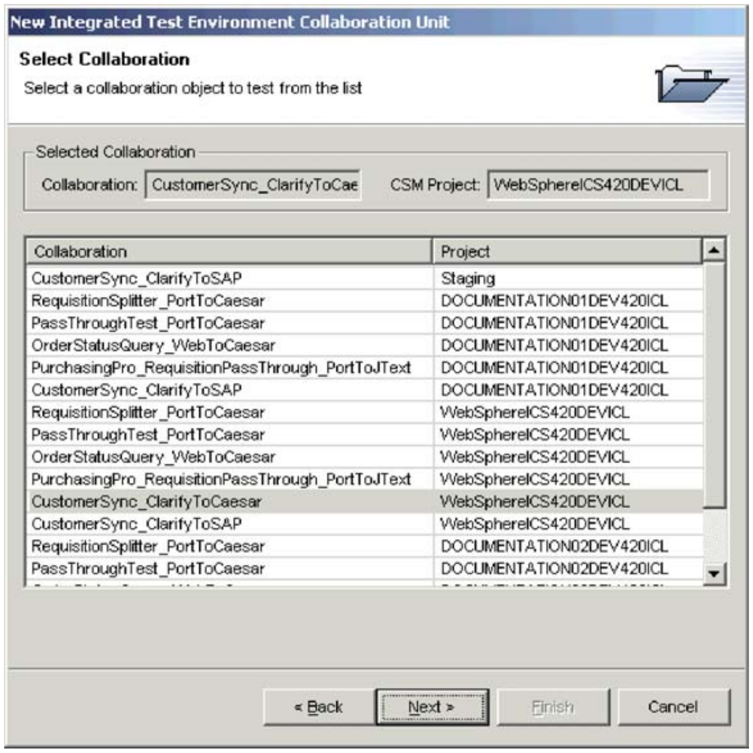
If you have the **Open the new test unit in the editor upon creation** checkbox enabled in the Integrated Test Environment preferences, then the test unit opens after you create it. If you do not have this preference enabled then you must open the test unit using the instructions in “Opening a test unit” on page 214. For more information on Integrated Test Environment preferences, see “Configuring test project preferences” on page 210.

Creating a test unit within Integrated Test Environment

Do the following to create a test unit within Integrated Test Environment:

1. Select **File > New > Integrated Test Environment Test Unit** from the menu bar.
2. At the “Select Collaboration” screen, 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.

   Figure 61 on page 213 shows the “Select Collaboration” screen.



New Integrated Test Environment Collaboration Unit

Select Collaboration

Select a collaboration object to test from the list

Selected Collaboration

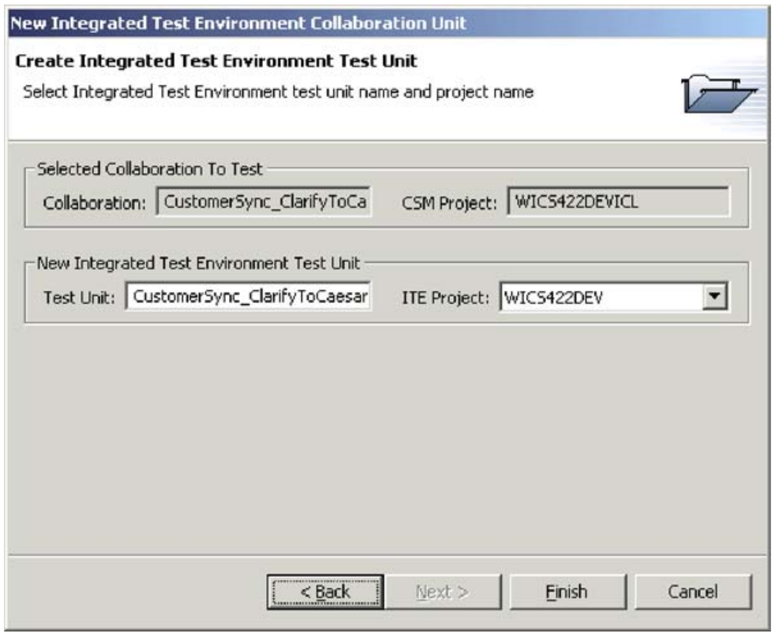
Collaboration: CustomerSync_ClarifyToCae   CSM Project: WebSphereICS420DEVICL

Collaboration	Project
CustomerSync_ClarifyToSAP	Staging
RequisitionSplitter_PortToCaesar	DOCUMENTATION01DEV420ICL
PassThroughTest_PortToCaesar	DOCUMENTATION01DEV420ICL
OrderStatusQuery_WebToCaesar	DOCUMENTATION01DEV420ICL
PurchasingPro_RequisitionPassThrough_PortToJText	DOCUMENTATION01DEV420ICL
CustomerSync_ClarifyToSAP	DOCUMENTATION01DEV420ICL
RequisitionSplitter_PortToCaesar	WebSphereICS420DEVICL
PassThroughTest_PortToCaesar	WebSphereICS420DEVICL
OrderStatusQuery_WebToCaesar	WebSphereICS420DEVICL
PurchasingPro_RequisitionPassThrough_PortToJText	WebSphereICS420DEVICL
CustomerSync_ClarifyToCaesar	WebSphereICS420DEVICL
CustomerSync_ClarifyToSAP	WebSphereICS420DEVICL
RequisitionSplitter_PortToCaesar	DOCUMENTATION02DEV420ICL
PassThroughTest_PortToCaesar	DOCUMENTATION02DEV420ICL

Figure 61. Selecting a collaboration to test

3. Click **Next**.
4. At the "Create Integrated Test Environment Test Unit" screen, do the following:
   a. Type a name for the test unit in the **Test Unit** field.
   b. Select the test project in which the test unit should be created from the **ITE Project** drop-down menu.

   Figure 62 on page 214 shows the "Create Integrated Test Environment Test Unit" screen.


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: CustomerSync_ClarifyToCa CSM Project: WICS422DEVICL

New Integrated Test Environment Test Unit

Test Unit: CustomerSync_ClarifyToCaesar ITE Project: WICS422DEV

Figure 62. Specifying the test unit name and project

5. Click **Finish**.

Creating a test unit within System Manager

Do the following to create a test unit from within System Manager:

1. Create a test project to contain the test unit as described in "Creating a test project" on page 211.
2. Right-click the collaboration object that represents the interface and select **Debug in Integrated Test Environment** from the context menu.
3. At the "Create Integrated Test Environment Test Unit" 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 **ITE Project** drop-down menu.

   Figure 62 shows the "Create Integrated Test Environment Test Unit" screen.
4. Click **Finish**.

Opening a test unit

Once you have created a test unit you can open it to view its layout and configure it. Do the following to open a test unit:

1. Expand the test project that contains the test unit in the "Integrated Test Environment Navigator" view.
2. Do one of the following to open a test unit:
   - Right-click the test unit you want to work with and choose **Open** from the context menu.
   - Double-click the test unit you want to work with.

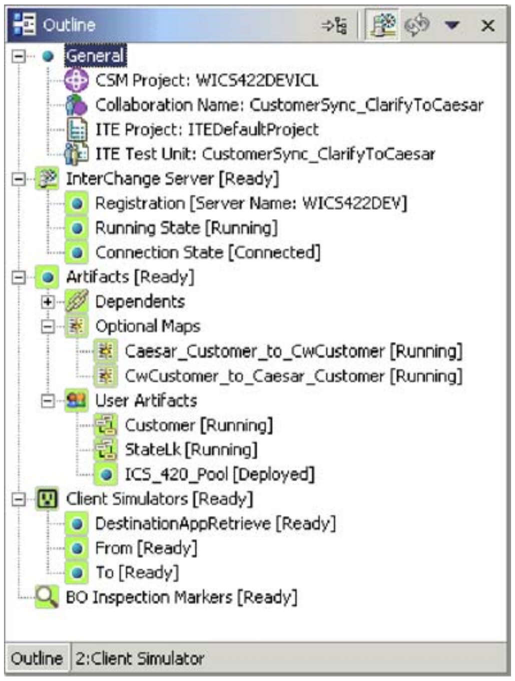
   The test unit opens, displaying a representation of the collaboration object in the Test Unit editor and populating the views of the perspective with information specific to the interface. Figure 56 on page 202 shows the appearance of the Integrated Test Environment perspective after you have opened a test unit.

Using the Outline view

The Outline view allows you to do a number of things described in the following sections:

- "Working with dependents using the Outline view"
- "Managing the repository using the Outline view" on page 217
- "Managing component states using the Outline view" on page 217
- "Executing tasks using the Outline view" on page 218
- "Verifying test readiness using the Outline view" on page 218
- "Showing and closing Client Simulator views using the Outline view" on page 219
- "Showing and closing BO Inspector views using the Outline view" on page 219
- "Refreshing BO Markers using the Outline view" on page 220

Figure 63 shows the Outline view.


Outline

General
CSM Project: WICS422DEVICL
Collaboration Name: CustomerSync_ClarifyToCaesar
ITE Project: ITEDefaultProject
ITE Test Unit: CustomerSync_ClarifyToCaesar
InterChange Server [Ready]
Registration [Server Name: WICS422DEV]
Running State [Running]
Connection State [Connected]
Artifacts [Ready]
Dependents
Optional Maps
Caesar_Customer_to_CwCustomer [Running]
CwCustomer_to_Caesar_Customer [Running]
User Artifacts
Customer [Running]
StateLk [Running]
ICS_420_Pool [Deployed]
Client Simulators [Ready]
DestinationAppRetrieve [Ready]
From [Ready]
To [Ready]
BO Inspection Markers [Ready]

Outline | 2:Client Simulator

Figure 63. Outline view showing User Artifacts

Working with dependents using the Outline view

As described in "Dependencies and references" on page 92, InterChange Server Express components depend on other components to function properly. To test an interface, you must make sure that all of its dependencies are resolved.

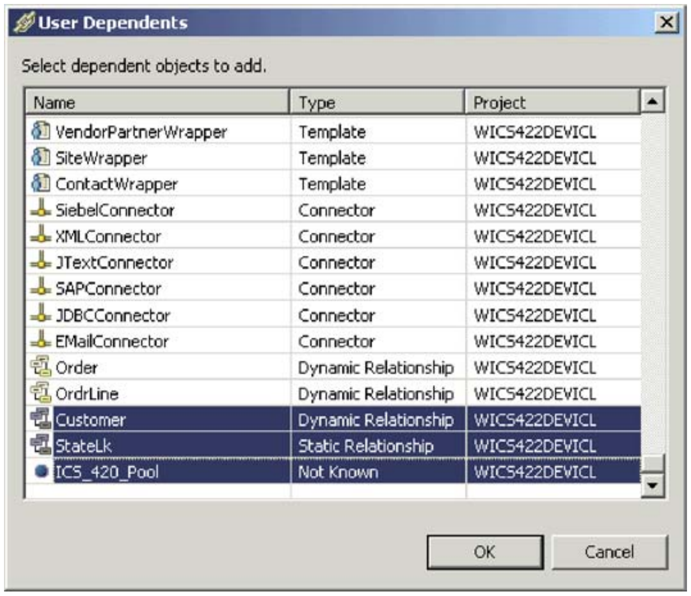
The Outline view in Integrated Test Environment lists the component dependencies that must be resolved to test the interface. The system is able to determine many dependencies among components, and lists them under the **Dependents** node beneath the **Artifacts** node in the Outline view.

"Dependencies that cannot be detected by the system" on page 94 describes the situations in which some components cannot be automatically detected by the system. If you use Integrated Test Environment views to deploy components when testing an interface then you must make sure that even the dependencies that cannot be detected by the system are resolved. To do this, you can specify those undetectable dependencies yourself. After you do so, the components are listed under the **User Artifacts** node beneath the **Artifacts** node in the Outline view.

Adding user dependents

Do one of the following to add a component to a test unit as a user dependent:

1. In the Outline view, expand the **Artifacts** node.
2. Right-click the **User Artifacts** node and select **Add User Artifacts** from the context menu.
3. At the **User Dependents** window 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 non-contiguous rows. Figure 64 shows the "User Dependents" dialog:



Select dependent objects to add.

Name	Type	Project
VendorPartnerWrapper	Template	WICS422DEVICL
SiteWrapper	Template	WICS422DEVICL
ContactWrapper	Template	WICS422DEVICL
SiebelConnector	Connector	WICS422DEVICL
XMLConnector	Connector	WICS422DEVICL
JTextConnector	Connector	WICS422DEVICL
SAPConnector	Connector	WICS422DEVICL
JDBCConnector	Connector	WICS422DEVICL
EMailConnector	Connector	WICS422DEVICL
Order	Dynamic Relationship	WICS422DEVICL
OrdrLine	Dynamic Relationship	WICS422DEVICL
Customer	Dynamic Relationship	WICS422DEVICL
StateLk	Static Relationship	WICS422DEVICL
ICS_420_Pool	Not Known	WICS422DEVICL

Figure 64. Adding user dependents

4. Click **OK**.

Note: Only components that are not already registered as dependents are displayed in the list.

Deleting user dependents

Expand the **Artifacts** node in the Outline view and do one of the following to remove components from the list of user artifacts:

- Right-click the **User Artifacts** node and select **Remove All User Artifacts** to remove all user artifacts from the list.
- Expand the **User Artifacts** node, then select the user artifacts you want to remove, right-click one of the selected artifacts, and choose Remove from the context menu.

  You can use standard Windows selection techniques to select multiple user projects for deployment at the same time, such as the following:
  - Hold down **Shift** to select contiguous items.
  - Hold down **Ctrl** to select non-contiguous items.

Storing user dependents

You can store the user dependents that you add to a test unit so that you do not have to add them each time. Do one of the following to store the user dependents:

1. Expand the **Artifacts** node in the Outline view.
2. Right-click the **User Artifacts** node and select **Save User Dependents** from the context menu.

Managing the repository using the Outline view

All of the components required for an interface must be deployed to the server repository to test the interface. The Outline view provides a way to work with the InterChange Server Express repository to ensure that it contains the required components.

When you deploy components, you can either use the **Deploy** option, which deploys the component, or the **Deploy with overwrite** option, which will deploy the component even if it already exists in the server repository. If the selected components already exist in the server repository and you use the **Deploy** option then the deployment will fail.

Do one of the following and choose either **Deploy** or **Deploy with overwrite** from the context menu to deploy the desired components:

- Right-click the "Artifacts" node to deploy all the components
- Right-click the "Dependents", "Optional Maps", or "User Artifacts" nodes to deploy all of the components that are listed under those nodes
- Right-click an individual component

To remove a component from the server repository using the Outline view, right-click the component in the view and choose **Remove from server** from the context menu.

Note: You can use standard Windows selection techniques to affect multiple components at the same time, such as the following:

- Hold down **Shift** to select contiguous items
- Hold down **Ctrl** to select non-contiguous items

Entries are written to the Integrated Test Environment Console and InterChange Server Express Console indicating whether the deployment was successful or not.

Managing component states using the Outline view

You can change the state of a component within Integrated Test Environment. This allows you to resolve problems you discover during testing—you can deactivate the component, modify it, re-deploy it, and re-activate it without using System Manager, repos_copy, or System Monitor.

To change the state of a component, right-click it in the Outline view and choose the desired state operation**—Start**, **Pause**, or **Stop**. Not all component types have all state operations available.

For information on component states, see the *System Administration Guide*.

Executing tasks using the Outline view

You can execute some of the tasks that are part of performing a test using the Outline view. For more information on tasks and task groups, see Table 25 on page 221.

To execute all of the tasks in the "Start Server" task group, right-click either the "InterChange Server" or "Running State" nodes in the Outline view and choose **Start Server** from the context menu.

To execute the "Shutdown Server" task, right-click either the "InterChange Server" or "Running State" nodes in the Outline view and choose **Stop Server** from the context menu.

To execute the "Connect To Server" task, right-click the "Connection State" node in the Outline view and choose **Connect To Server** from the context menu.

To execute the "Disconnect Server" task, right-click the "Connection State" node in the Outline view and choose **Disconnect Server** from the context menu.

To execute all of the tasks in the "Start Client Simulators" task group, right-click the "Client Simulators" node in the Outline view and choose **Start client simulators** from the context menu.

To execute all of the tasks in the "Start BO Trace" task group, right-click the "BO Inspection Markers" node in the Outline view and choose **Start BO Trace** from the context menu.

To execute the "Stop BO Trace" task, right-click the "BO Inspection Markers" node in the Outline view and choose **Stop BO Trace** from the context menu.

Verifying test readiness using the Outline view

The Outline view gives a visual indicator of the state of several conditions that must be satisfied to perform a test. This allows you to quickly confirm if you are ready to perform tests in Integrated Test Environment without having to search for indicators in log text. Each condition that must be satisfied has a green background if it has been satisfied and has a red background if it has not been satisfied. Each condition also has a phrase that indicates its status. The following are the different nodes whose conditions must be satisfied, along with descriptions of the requirements and the phrases displayed when the conditions have been satisfied:

- The "InterChange Server" node must be in a "Ready" state to test an interface. This node represents the state of the server and Integrated Test Environment's connection to it; it encompasses the following nodes:
  - The "Registration" node lists the InterChange Server Express instance that Integrated Test Environment has been configured to communicate with.

    For more information about specifying the InterChange Server Express instance to use, see "Selecting a server configuration" on page 204.
  - The "Running State" node indicates if the InterChange Server Express instance is running or not; it must be in the "Ready" state to perform a test.
  - The "Connection State" node indicates if Integrated Test Environment is connected to the InterChange Server Express instance; it must be in the "Connected" state to perform a test.

- The “Artifacts” node must be in a “Ready” state to test an interface. For this node to be in a “Ready” state, all of the components in each of the nodes beneath it must be deployed to the server and running.

  Components that have states, such as connectors and collaboration objects, must be in a “Running” state. Components that do not have states but that are required for the interface, such as business object definitions and collaboration templates, must be in the “Deployed” state.
- The “Client Simulators” node must be in a “Ready” state to test an interface. For this node to be in a “Ready” state, all of the ports listed beneath it must have Client Simulator views associated with them and connected to the server.
- The “BO Inspection Markers” node must be in a “Ready” state for business object tracing to work while testing an interface. For this node to be in a “Ready” state, the “Start BO Trace” task must be started successfully.

Showing and closing Client Simulator views using the Outline view

You use Client Simulator views to emulate connectors and access clients when testing an interface. Client Simulator views provide interfaces to create business object instances with test data, send them as business object requests, and reply to them as business object responses.

To load the profile for a connector into a particular Client Simulator view, right-click the desired port under the “Client Simulators” node and select **Use Client Simulator *x***, where ***x*** is the number of the Client Simulator view position you want to use for that connector.

Table 29 on page 235 shows the default interface positions in which the different Client Simulator views are displayed.

Views in Integrated Test Environment can be concealed by other views during different stages of a test; for example, a Client Simulator view might be concealed by a BO Inspector view. To reveal a Client Simulator view that is concealed, right-click the desired port under the “Client Simulators” node and select **Show Client Simulator** to give focus to the Client Simulator view.

To close a Client Simulator view, right-click the desired port under the “Client Simulators” node and select **Close Client Simulator**.

Note: This option not only closes the Client Simulator view, but also causes the Client Simulator view to stop emulating the connector.

For more information on working with Client Simulator views, see “Using the Client Simulator view” on page 234.

Showing and closing BO Inspector views using the Outline view

If you use business object tracing, Integrated Test Environment records business object data at certain points during the execution of the interface.

To show BO Inspector views for all the business object markers in the interface, right-click the “BO Inspection Markers” node in the Outline view and select **Show All BO Viewers** from the context menu. BO Marker icons are identified in Figure 67 on page 230.

To close all the displayed BO Inspector views in the interface, right-click the “BO Inspection Markers” node in the Outline view and select **Close All BO Viewers** from the context menu.

Table 31 on page 249 shows the default interface positions in which the different BO Inspector views are displayed.

For more information on using business object tracing, see Table 25 on page 221.

For more information on working with BO Inspector views, see “Using the BO Inspector view” on page 247.

Refreshing BO Markers using the Outline view

If you are using business object tracing but have disabled auto-update of the trace information then you must explicitly refresh the BO Markers to update them.

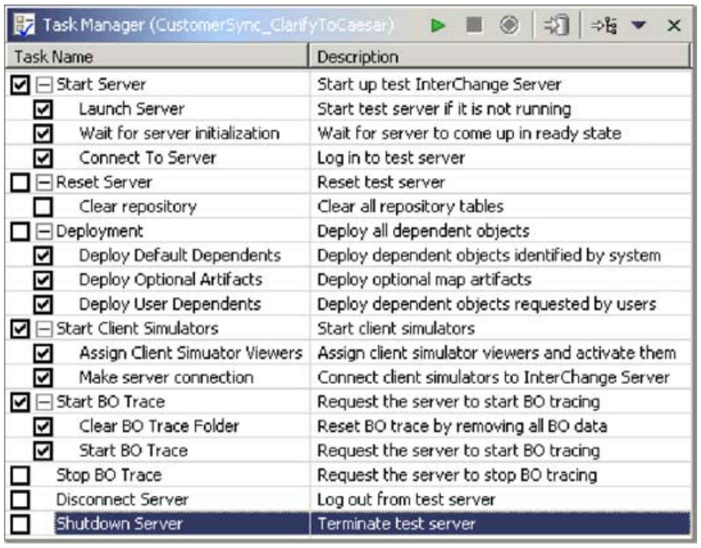
To update BO Markers, right-click the “BO Inspection Markers” node in the Outline view and select **Refresh BO Markers** from the context menu.

Using the Task Manager view

The Task Manager view lets you automate many of the tasks you have to perform when testing an interface.

Several tasks must commonly be executed at certain stages when performing a test. For instance, you must start InterChange Server Express and connect the Integrated Test Environment perspective to it when preparing to perform a test. Later on in the test you must launch Client Simulator views for the connectors and access clients in the interface, and connect them to the InterChange Server Express instance. To make it easier to execute all of the tasks that tend to relate to a particular stage during the running of a test, Task Manager displays related tasks in groups. The tasks for launching Client Simulator views and connecting them to the InterChange Server Express instance, for example, are collected in a task group named “Start Client Simulators”. Task Manager allows you to execute either a single task, or a task group as a whole.

Figure 65 on page 221 shows the Task Manager view with several tasks and task groups selected.



Task Manager (CustomerSync_ClarifyToCaesar)

Task Name	Description
✓ ⊟ Start Server	Start up test InterChange Server
✓ Launch Server	Start test server if it is not running
✓ Wait for server initialization	Wait for server to come up in ready state
✓ Connect To Server	Log in to test server
☐ ⊟ Reset Server	Reset test server
☐ Clear repository	Clear all repository tables
☐ ⊟ Deployment	Deploy all dependent objects
✓ Deploy Default Dependents	Deploy dependent objects identified by system
✓ Deploy Optional Artifacts	Deploy optional map artifacts
✓ Deploy User Dependents	Deploy dependent objects requested by users
✓ ⊟ Start Client Simulators	Start client simulators
✓ Assign Client Simuator Viewers	Assign client simulator viewers and activate them
✓ Make server connection	Connect client simulators to InterChange Server
✓ ⊟ Start BO Trace	Request the server to start BO tracing
✓ Clear BO Trace Folder	Reset BO trace by removing all BO data
✓ Start BO Trace	Request the server to start BO tracing
☐ Stop BO Trace	Request the server to stop BO tracing
☐ Disconnect Server	Log out from test server
☐ Shutdown Server	Terminate test server

Figure 65. Task Manager view

Table 25 describes the tasks and task groups in the Task Manager view.

Table 25. Tasks and task groups in the Task Manager view

Task name	**Task description**
Start Server	This is the group for all tasks related to starting InterChange Server Express.
Launch Server	This task starts the InterChange Server Express instance. This task belongs to the **Start Server** task group.
Wait for server initialization	This task causes Integrated Test Environment to wait until InterChange Server Express has started successfully before trying to connect to it. If you have not selected this task and Integrated Test Environment tries to connect to InterChange Server Express before the server starts, then Integrated Test Environment will not be able to connect to it. Enabling this option ensures that the task of connecting Integrated Test Environment to InterChange Server Express is automated.
Connect to Server	This task connects Integrated Test Environment to the InterChange Server Express instance so that you can manage and deploy components and view data. This task belongs to the **Start Server** task group. **Note:** The IBM Java Object Request Broker (ORB) must be running for Integrated Test Environment to connect to the server.
Reset Server	This is the group for the **Clear Repository** task.
Clear Repository	This task deletes the repository of the InterChange Server Express instance. You would want to perform this task if you plan to deploy a large number of components through Integrated Test Environment and some of the components already exist in the repository. This task belongs to the **Reset Server** task group.

Table 25. Tasks and task groups in the Task Manager view (continued)

Task name	**Task description**
Deployment	This is the group for all tasks related to deploying components to the InterChange Server Express repository through the Task Manager view. All of the dependent components for an interface must be deployed to test the interface. To deploy components using Integrated Test Environment, you must make sure the InterChange Server Express instance starts in design mode. For more information, see "Configuring InterChange Server Express to start in design mode" on page 208. For more information on the Dependency view, see "Using the Outline view" on page 215.
Deploy Default Dependents	This task deploys all the components that are determined by the system to be required for the interface to function, and that are listed beneath the **Default Dependents** node in the Dependency view. This task belongs to the **Deployment** task group.
Deploy Optional Maps	
Deploy User Dependents	This task deploys all the components that you add to the **User Artifacts** node in the Outline view. This task belongs to the **Deployment** task group.
Start Client Simulators	This is the group for all tasks relating to emulating connector agents and access clients.
Assign Client Simulator Viewers	This task starts Client Simulator view windows for all of the connectors and access clients involved in the interface. This task belongs to the **Start Client Simulators** task group.
Make server connection	This task causes the Client Simulator views to connect to the InterChange Server Express instance. Connector agents and access clients must be connected to InterChange Server Express to exchange business objects. By default a connector starts using its definition in the repository. You can use the Client Simulator view to cause the agent to connect with a configuration file as described in "Emulating a connector using the repository definition" on page 236. Client Simulators can only automatically connect to the server in this way if the `DeliveryTransport` property of the connector being emulated is set to the value `IDL`. This task belongs to the **Start Client Simulators** task group. **Note:** The IBM ORB must be running for Integrated Test Environment to connect to the server.

Table 25. Tasks and task groups in the Task Manager view (continued)

Task name	Task description
Start BO Trace	This is the group for all tasks relating to business object tracing. Business object tracing records business object data as business objects are processed by components in the system. For more information, see “Using the BO Inspector view” on page 247.
Clear BO Trace Folder	This task clears the folder in which business object trace data is stored. You might want to clear the business object trace folder for the following reasons: • Business object trace files take up disk storage and use resources to be loaded for viewing in the Integrated Test Environment perspective. You might want to clear the folder to preserve storage and resources. • You might not want old test data displayed in the Integrated Test Environment. You can clear the business object test data from previous tests so that only data from subsequent tests is displayed. This task belongs to the **Start BO Trace** task group.
Start BO Trace	This task starts business object tracing. You must start business object tracing for Integrated Test Environment to store business object data during the tests you perform. This task belongs to the **Start BO Trace** task group.
Stop BO Trace	This task stops business object tracing. Business object tracing can be a very resource-intensive task, so depending on your objectives for the testing you might want to stop business object tracing if it is more disruptive than helpful.
Disconnect Server	This task disconnects Integrated Test Environment from the InterChange Server Express instance. It does not shut down the InterChange Server Express instance.
Shutdown Server	This task shuts down the InterChange Server Express instance.

Making task selections

There are several ways to select tasks and task groups. Table 26 describes the different ways. It is recommended that you combine the different approaches to be most efficient.

Table 26. Ways of selecting tasks and task groups in the Task Manager view

To accomplish this...	Do this...
Check a single task within a group	Click the checkbox next to the task.
Check a group of tasks	Click the checkbox next to the task group.
Check all tasks in the view	Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Check All** from the menu.

Table 26. Ways of selecting tasks and task groups in the Task Manager view (continued)

To accomplish this...	Do this...
Uncheck all tasks in the view	Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Uncheck All** from the menu.
Check your preferred tasks	Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Preference Selection** from the menu. For more information, see "Configuring your preferred task selections" on page 226.
Check the default tasks	Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Default Selection** from the menu. The default task and task group selections are: • Start Server – Launch Server – Wait for server initialization – Connect to Server • Deployment – Deploy Default Dependents – Deploy Optional Artifacts – Deploy User Dependents • Start Client Simulators – Assign Client Simulator Viewers • Start BO Trace – Start BO Trace

Executing tasks using the Task Manager view

There are several ways to execute the tasks you have selected in the Task Manager view. The following sections describe the different methods:

- "Executing selected groups of tasks"
- "Executing a group of tasks" on page 225
- "Executing a single task" on page 225

Executing selected groups of tasks

You can execute all of the tasks within several task groups at once. This makes it easy to prepare the interface for testing without manually managing it through the different stages.

A convenient approach, for instance, is to execute all of the tasks within the **Start Server**, **Start Client Simulators**, and **Start BO Trace** task groups.

Do the following to execute all of the tasks within a selection of task groups:

1. Enable the checkboxes for the tasks you want to execute.
2. Enable the checkboxes for the task groups that the tasks you want to execute belong to.
3. Do one of the following to execute the selected tasks in the selected task groups:
   - Select **Integrated Test Environment > Run** from the menu bar.

- Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Run** from the menu.
- Click the **Execute tasks** button in the title bar of the Task Manager view.

Be sure not to enable checkboxes for tasks or task groups that have conflicting results. For instance, do not enable both the **Launch Server** and **Shutdown Server** tasks.

Executing a group of tasks

To execute a single group of tasks, right-click the task group and choose **Run**. Only the tasks in the task group you right-clicked will execute. The checkbox for the task group does not have to be enabled for its tasks to execute, though the checkboxes for the tasks you want to execute within the group do have to be enabled.

You might use this approach frequently with the **Start Client Simulators** task group. You will frequently execute groups of tasks to initially prepare a test, as described in "Executing selected groups of tasks" on page 224. You might close the Client Simulator views you have open during the course of a series of tests, however, and then need to launch them and connect them again. You do not have to execute all of the tasks that have already been performed, however—you can execute only the **Start Client Simulators** task group.

Executing a single task

To execute a single task, right-click the task and choose **Run**. Only the task you right-clicked will execute. The checkbox for the task does not have to be enabled for it to execute.

You might use this approach to shut down the server. You might want to start the server again immediately afterwards, and not want to re-enable all the required checkboxes. You can just right-click the **Shutdown Server** task and execute it alone.

Stopping tasks

You can instruct Task Manager to stop its queue of currently executing tasks. It will allow whichever task is currently executing to finish, but will not execute any subsequent tasks. For example, if you choose to stop the tasks while the **Launch Server** task is executing, then Task Manager will allow the server to finish starting.

Do one of the following to stop the currently executing tasks:

- Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Stop Tasks** from the menu.
- Click the **Stop tasks** button in the title bar of the Task Manager view.

Killing tasks

You can instruct Task Manager to kill its currently executing tasks. It will terminate the currently executing task and will not execute any subsequent tasks either. For example, if you choose to kill the tasks while the **Launch Server** task is executing, then Task Manager does not allow the server to finish starting.

To kill the currently executing tasks, click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Kill Tasks** from the menu.

Pausing tasks

You can pause the currently executing tasks so that Task Manager allows the currently executing task to complete, but does not execute the remaining tasks until you unpause it. This allows you to start a series of tasks and perform an action which must be done before a later task is reached, but which you forgot about, and without stopping and restarting the remaining tasks. For example, if you want to start the server and deploy the dependents, but some of the components already exist in the repository and you do not realize that until you have already executed the tasks, you could pause the tasks before the deployment stage begins. Then you can use repos_copy to delete the repository so that when the deployment begins it does not fail.

Do one of the following to pause the currently executing tasks:

- Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Pause Tasks** from the menu.
- Click the **Pause tasks** button in the title bar of the Task Manager view.

Do one of the following to cause the tasks to resume executing:

- Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Pause Tasks** from the menu.
- Click the **Pause tasks** button in the title bar of the Task Manager view.

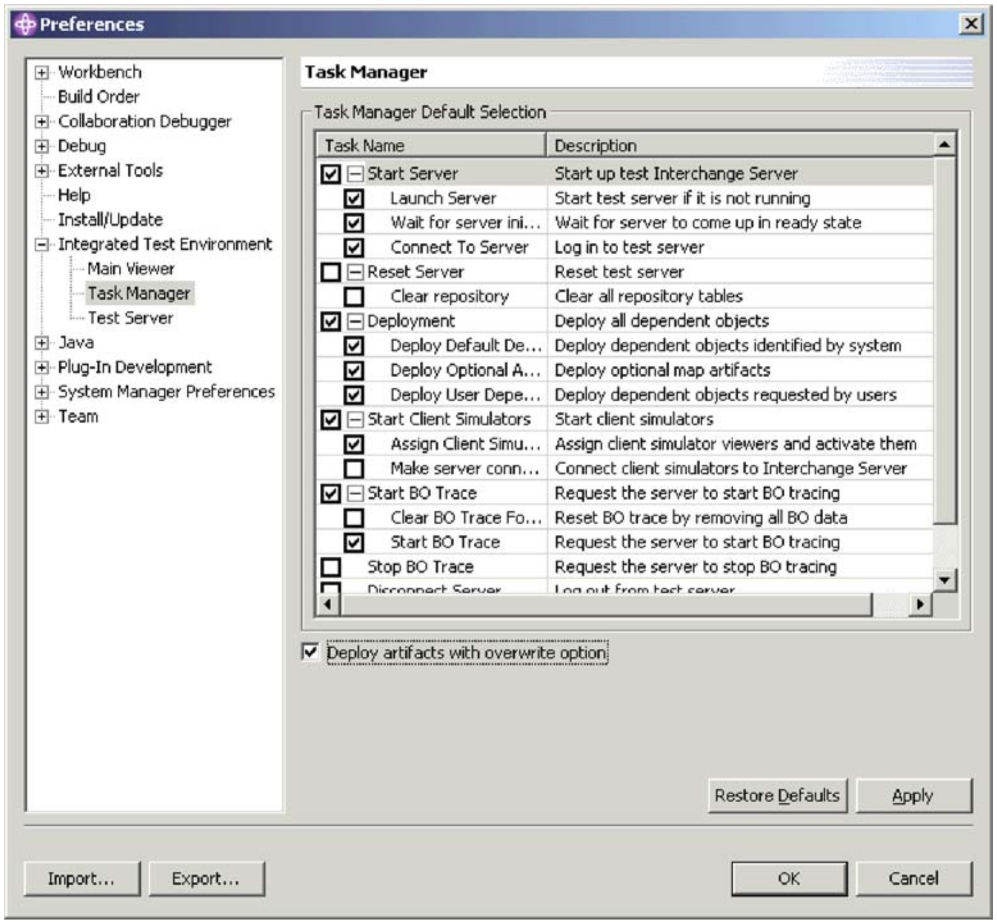
Configuring your preferred task selections

You can save selections as a preference so that you can conveniently apply the selection scheme you use most commonly.

Do the following to configure your preferred task selections:

1. Select **Window > Preferences** from the menu bar of the workbench.
2. Expand the **Integrated Test Environment** node.
3. Select **Task Manager**.
4. Expand task groups and enable the checkboxes for the individual tasks and task groups that you want selected when you apply your preferences.
5. Click **OK**.

Figure 66 on page 227 shows the Task Manager preference selection interface.


Preferences

- Workbench
- Build Order
- Collaboration Debugger
- Debug
- External Tools
- Help
- Install/Update
- Integrated Test Environment
  - Main Viewer
  - Task Manager
  - Test Server
- Java
- Plug-In Development
- System Manager Preferences
- Team

Task Manager

Task Manager Default Selection

Task Name	Description
✓ ⊟ Start Server	Start up test Interchange Server
✓ Launch Server	Start test server if it is not running
✓ Wait for server ini...	Wait for server to come up in ready state
✓ Connect To Server	Log in to test server
☐ ⊟ Reset Server	Reset test server
☐ Clear repository	Clear all repository tables
✓ ⊟ Deployment	Deploy all dependent objects
✓ Deploy Default De...	Deploy dependent objects identified by system
✓ Deploy Optional A...	Deploy optional map artifacts
✓ Deploy User Depe...	Deploy dependent objects requested by users
✓ ⊟ Start Client Simulators	Start client simulators
✓ Assign Client Simu...	Assign client simulator viewers and activate them
☐ Make server conn...	Connect client simulators to Interchange Server
✓ ⊟ Start BO Trace	Request the server to start BO tracing
☐ Clear BO Trace Fo...	Reset BO trace by removing all BO data
✓ Start BO Trace	Request the server to start BO tracing
☐ Stop BO Trace	Request the server to stop BO tracing
☐ Disconnect Server	Log out from test server

✓ Deploy artifacts with overwrite option

Restore Defaults | Apply

Import... | Export... | OK | Cancel

Figure 66. Integrated Test Environment task preferences

Configuring the default task selections

Do the following to configure the tasks and task groups that will be enabled by default each time you open a particular test unit:

1. Open the test unit for which you want to save the default selections.
2. Expand task groups and enable the checkboxes for the individual tasks and task groups that you want selected by default when you open a test unit.
3. Do one of the following to store the selections:
   - Click the down arrow in the right-hand corner of the title bar of the Task Manager view and choose **Store Selections** from the menu.
   - Click the **Store current selections** button in the title bar of the Task Manager view.

When you open the test unit the next time, the tasks you chose will be selected in the Task Manager view.

Using the Integrated Test Environment Console and InterChange Server Express Console views

As described in "Integrated Test Environment interface" on page 202, the Integrated Test Environment Console tab displays information about the execution of Integrated Test Environment and the InterChange Server Express Console displays information about the execution of InterChange Server Express.

You can use the Integrated Test Environment Console view to determine when Integrated Test Environment tasks have completed and if they were successful. For instance, messages are written when Integrated Test Environment connects successfully to InterChange Server Express, or when Integrated Test Environment launches Map Designer Express to debug a map.

If you started InterChange Server Express from within Integrated Test Environment, the InterChange Server Express Console view displays the logging and tracing output of the server, just as the MS-DOS console does. You can then use the view to troubleshoot the interfaces you are testing. It has the benefit that it is contained within the Integrated Test Environment perspective and can be conveniently searched.

The following sections describe some tasks you will frequently want to perform when using the Integrated Test Environment Console and InterChange Server Express Console views.

Finding text

You might want to search the Integrated Test Environment Console or InterChange Server Express Console views for particular phrases. For instance, you might want to confirm that the InterChange Server Express started successfully. You could search the InterChange Server Express Console view for the text "is ready" to do so.

Do the following to search the Integrated Test Environment Console or InterChange Server Express Console views for text strings:

1. Right-click in the Integrated Test Environment Console or InterChange Server Express Console view and select **Find/Replace** from the context menu.
2. Type the text string for which you want to search in the **Find** field.
3. Enable the **Forward** radio button to search the text after the cursor or enable the **Backward** radio button to search the text before the cursor.
4. Enable other options such as the **Case Sensitive** checkbox.
5. Click **Find Next**.

Copying text

You might want to copy text from the console views to show the text to other developers or Technical Support to get help with troubleshooting.

To copy text from a console view, select the desired text, then right-click it and choose **Copy** from the context menu.

Maximizing the view window

As with all workbench editors and viewers, you can maximize the Integrated Test Environment Console or InterChange Server Express Console views by double-clicking their title bar. Although this is not useful with most viewers, it can be very helpful when troubleshooting an interface by examining information in the InterChange Server Express Console view.

Clearing the view window

You might want to clear the Integrated Test Environment Console or InterChange Server Express Console views to eliminate the information displayed in them. This can be helpful when you want to view only information that is relevant to debugging a problem in the interface. You can clear the console view, reproduce the problem, and troubleshoot it without having to search through information that is not related to the problem.

Configuring Integrated Test Environment Console preferences

Do the following to configure your preferences for the Integrated Test Environment Console:

1. Select **Window > Preferences** from the menu bar of the workbench.
2. Select **Integrated Test Environment**.

   Figure 59 on page 210 shows the Integrated Test Environment preferences.
3. Click the button to the right of the “Normal text” label.
4. At the “Color” dialog, select the color you would like the Integrated Test Environment Console to use for messages that are not warnings or errors.
5. Click **OK**.
6. Repeat steps 3 through 5 for the “Warning text” and “Error text” labels to set the color for those types of messages.
7. Click **OK**.

Using the Test Unit view

As described in “Test Unit editor” on page 203, the Test Unit view provides a graphical view of the interface and provides mechanisms to manipulate the components in the interface.

Using the Server Context Overlay

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.

To enable or refresh the Server Context Overlay as described in the following sections, Integrated Test Environment must be connected to the server. For more information, see the **Connect to Server** task in Table 25 on page 221.

When the Server Context Overlay is enabled, icons are added to the icons for each component shown in the Test Unit view to indicate the state of the component. This is useful because all the components that participate in an interface must be running in order to test it.

Figure 67 on page 230 shows a Test Unit view where the Server Context Overlay is enabled. Most of the components have green arrow icons, indicating they are in a started state. The **ClarifyConnector** component has an orange circle indicating that it is in a paused state. The collaboration object **CustomerSync_ClarifyToCaesar** has a red square icon, indicating that it is in a stopped state. Table 27 on page 230 describes the different component state icons and their meanings.

For information on component states, see the *System Administration Guide*.

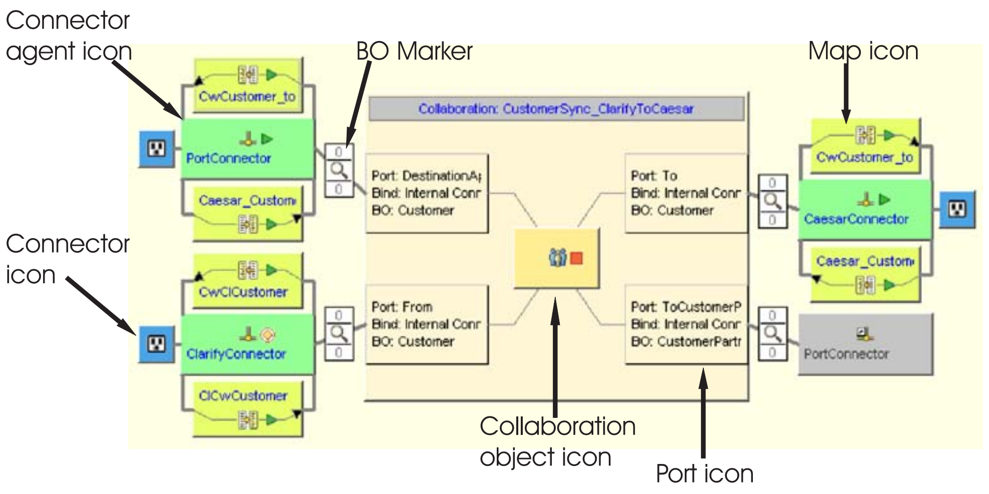


Figure 67. Test Unit view with Server Context Overlay enabled

Table 27. Component state icons and meanings

Icon	Meaning
Green arrow	The component is started.
Orange circle	The component is paused.
Red square	The component is stopped.
Question mark	The component is in an unknown state.
Red circle with white "X"	The component does not exist in the repository.
Blue checkmark	The component deployed successfully.
Red "X"	Integrated Test Environment is disconnected from the server, so the Server Context Overlay cannot display the current state of the component.

Enabling and disabling the Server Context Overlay

To enable the Server Context Overlay, right-click the background of the Test Unit view and select **Server Context Overlay** from the context menu.

When the Server Context Overlay is enabled, the component state icons appear in the Test Unit view and right-click menu items for the components are enabled.

To disable the Server Context Overlay, right-click the background of the Test Unit view and select **Server Context Overlay** from the context menu.

To have the Server Context Overlay always appear, enable the **Show Server Context Overlay always** option in the Integrated Test Environment preferences. Figure 59 on page 210 shows the Integrated Test Environment preferences.

Refreshing the Server Context Overlay

The state information in the Server Context Overlay is not dynamic, so if the state of a component changes after the Server Context Overlay is initially enabled due to actions external to Integrated Test Environment, then the state information the Server Context Overlay presents will be inaccurate. For instance, if you change the state of a component using the InterChange Server Express Component

Management view in System Manager, the Server Context Overlay will still reflect the state of the component prior to the change. You must refresh the Server Context Overlay to update it with current state information.

To refresh the Server Context Overlay, right-click the background of the Test Unit view and select **Refresh Server Context Overlay** from the context menu.

Managing the repository using the Test Unit view

All of the components required for an interface must be deployed to the server repository to test the interface. The Outline view provides a way to work with the InterChange Server Express repository to ensure that it contains the required components.

To deploy a component when it already exists in the server repository, right-click the component in the Test Unit view and choose **Deploy with overwrite** from the context menu.

To deploy a component if it does not already exist in the server repository, right-click the component in the Test Unit view and choose **Deploy** from the context menu. If the selected components already exist in the server repository and you use the **Deploy** option then the deployment will fail.

To remove a component from the server repository using the Test Unit view, right-click the component in the view and choose **Remove from server** from the context menu.

Entries are written to the Integrated Test Environment Console and InterChange Server Express Console indicating whether the deployment was successful or not.

Managing component states using the Test Unit view

You can change the state of a component using the Test Unit view. This allows you to resolve problems you discover during testing—you can deactivate the component, modify it, re-deploy it, and re-activate it without using System Manager, repos_copy, or System Monitor.

To change the state of a component, right-click its icon in the Server Context Overlay and choose the desired state operation—**Start**, **Pause**, or **Stop**. Not all component types have all state operations available.

For information on component states, see the *System Administration Guide*.

Using, showing, and closing Client Simulator views using the Test Unit view

You use Client Simulator views to emulate connectors and access clients when testing an interface. Client Simulator views provide interfaces to create business object instances with test data, send them as business object requests, and reply to them as business object responses.

To load the profile for a connector into a particular Client Simulator view, right-click the icon for the connector in the Test Unit view and select **Use Client Simulator *x***, where ***x*** is the number of the Client Simulator view position you want to use for that connector.

Table 29 on page 235 shows the default interface positions in which the different Client Simulator views are displayed.

Views in Integrated Test Environment can be concealed by other views during different stages of a test; for example, a Client Simulator view might be concealed by a BO Inspector view. To reveal a Client Simulator view that is concealed, right-click the connector icon and select **Show Client Simulator** to give focus to the Client Simulator view.

To close a Client Simulator view, right-click the connector icon and select **Close Client Simulator**.

Note: This option not only closes the Client Simulator view, but also causes the Client Simulator view to stop emulating the connector.

For more information on working with Client Simulator views, see "Using the Client Simulator view" on page 234.

Showing BO Inspector views

If you use business object tracing, Integrated Test Environment records business object data at certain points during the execution of the interface.

To show the business object data for a BO Marker into a particular BO Inspector view, right-click the BO Marker and select **Show in BO Viewer*x***, where ***x*** is the number of the BO Inspector view position you want to use for that marker. BO Marker icons are identified in Figure 67 on page 230.

Table 31 on page 249 shows the default interface positions in which the different BO Inspector views are displayed.

For more information on using business object tracing, see Table 25 on page 221.

For more information on working with BO Inspector views, see "Using the BO Inspector view" on page 247.

Refreshing BO Markers

If you are using business object tracing but have disabled auto-update of the trace information then you must explicitly refresh the BO Markers to update them.

To update BO Markers, right-click the Server Context Overlay and select **Refresh BO Markers** from the context menu.

Debugging maps using the Test Unit view

If you use business object tracing you can launch Map Designer Express in debug mode for maps that transform generic objects to application-specific objects, using the data produced during the execution of the test within Integrated Test Environment.

Do the following to debug maps using the Test Unit view:

1. Enable the Server Context Overlay as described in "Enabling and disabling the Server Context Overlay" on page 230.
2. Execute the Start BO Trace task by following the instructions in one of the following sections:
   - "Executing tasks using the Outline view" on page 218

- “Using the Task Manager view” on page 220.

3. When you have sent a business object through the interface, right-click the icon for the map you want to debug and select **Debug Map** from the context menu.
4. At the “Select a BO Instance” window select the business object instance you want to use and click **OK**.

   Map Designer Express launches in debug mode with the specified map opened and the specified test data supplied. For more information on how use the debugging features of Map Designer Express, see the *Map Development Guide*.

Note: Map Designer Express executes independently of Integrated Test Environment. Integrated Test Environment does not continue to communicate with Map Designer Express after you launch it.

Changing the Test Unit view scale

There are several options available for changing the scale of the Test Unit view.

To change the Test Unit view scale, right-click the Server Context Overlay and select the desired scale from the context menu. Table 28 lists and describes the scale options.

Table 28. Test Unit view scales and descriptions

Test Unit view scale type	**Description**
Fit Window	This scale deforms the interface illustration to fit the Test Unit view horizontally.
Fill Window	This scale deforms the interface illustration to fill the Test Unit view. Typically this results in the interface icons being stretched vertically.
View Normal Scale	This is the default scale.
View Double Scale	This scale increases the magnification of the interface illustration by two hundred percent.

Configuring Test Unit view preferences

Do the following to configure your preferences for the Test Unit view:

1. Select **Window > Preferences** from the menu bar of the workbench.
2. Expand **Integrated Test Environment**.
3. Select **Main Viewer**.

   Figure 59 on page 210 shows the Integrated Test Environment preferences.

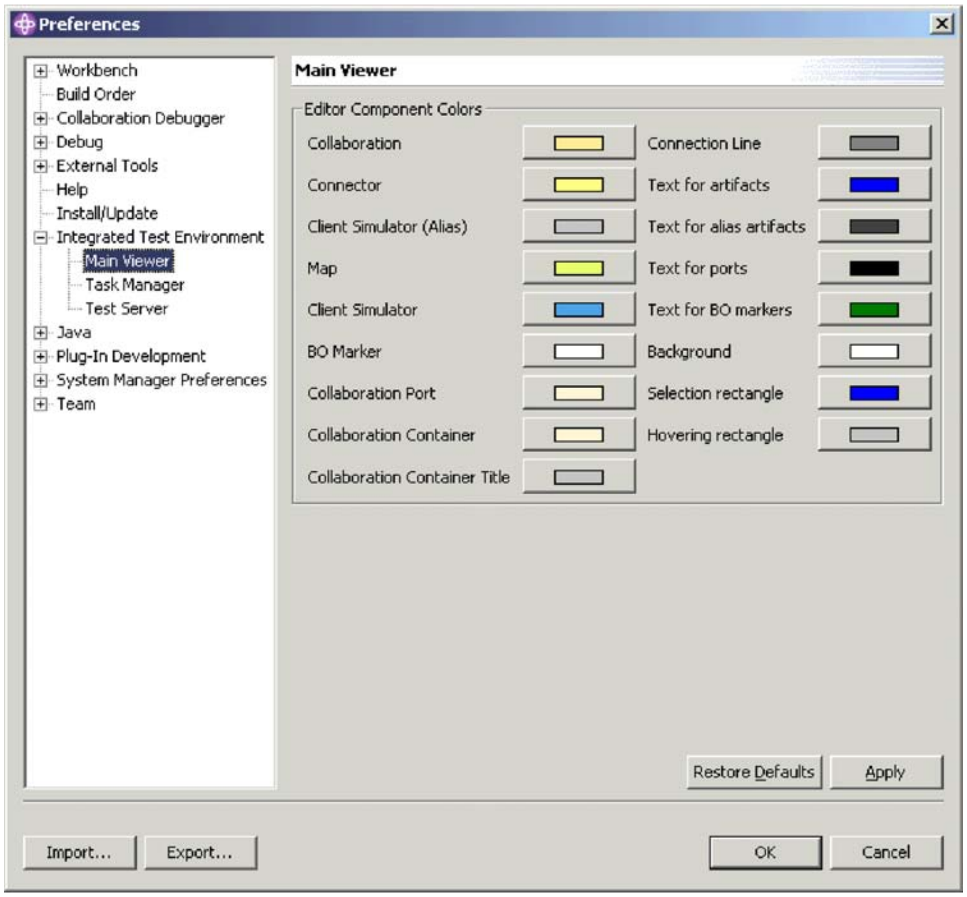


Figure 68. Test Unit view preferences

4. Click the button to the right of a component label in the "Edit Component Colors" pane.
5. At the "Color" dialog, select the color you would like the Test Unit view to use for the background of that component type.
6. Click **OK**.
7. Repeat steps 4 through 6 for the other component types.
8. Click **OK**.

Using the Client Simulator view

The Client Simulator view allows you to emulate the clients in IBM WebSphere Business Integration Server Express interfaces—connector agents and access clients. The view provides interfaces to connect the clients to the server, create and send business object requests, and reply with business object responses.

Showing and closing Client Simulator views

The following sections describe the numerous ways you can show and close Client Simulator views.

The "Default interface position" column in Table 29 on page 235 lists the views whose default positions in the interface are overlaid with the Client Simulator views in the "Client Simulator view number" column.

Table 29. Default Client Simulator view positions

Client Simulator view number	**Default interface position**
Test Connector 1	ITE Navigator
Test Connector 2	Outline
Test Connector 3	Properties
Test Connector 4	Dependency
Test Connector 5	ITE Navigator
Test Connector 6	Outline
Test Connector 7	Properties
Test Connector 8	Dependency
Test Connector 9	ITE Navigator

Showing Client Simulator views

Use one of the following techniques to show Client Simulator views for the connectors and access clients in the interface you are testing:

- “Showing and closing Client Simulator views using the Outline view” on page 219 along with Table 25 on page 221
- “Using the Task Manager view” on page 220 along with Table 25 on page 221
- “Using, showing, and closing Client Simulator views using the Test Unit view” on page 231 along with Table 25 on page 221
- Select **Integrated Test Environment > Show All Client Simulator Views** from the menu bar to show all nine Client Simulator views.
- Select **Window > Show View > *x*:Client Simulator** from the menu bar to launch a specific Client Simulator view, where *x* is the number of the Client Simulator view position you want to use for that connector.

Table 29 shows the default interface positions in which the different Client Simulator views are displayed.

Closing Client Simulator views

Use one of the following techniques to close Client Simulator views:

- “Showing and closing Client Simulator views using the Outline view” on page 219 along with Table 25 on page 221.
- “Using, showing, and closing Client Simulator views using the Test Unit view” on page 231.
- Select **Integrated Test Environment > Close All Client Simulator Views** from the menu bar to close all open Client Simulator views.
- Either click the **Close** button in the right-hand side of a Client Simulator view title bar, or right-click the Client Simulator view title bar and select **Close** from the context menu.

Connecting a Client Simulator view to the server

You must connect Client Simulator views to the server to emulate a connector as part of testing an interface.

When using the Client Simulator view, you can either emulate a connector using the definition in the repository or using a configuration file. You can also use the Client Simulator view to emulate an access client.

Note that you can also use the Task Manager view to connect agents to the server. For more information, see Table 25 on page 221.

Emulating a connector using the repository definition

Do one of the following to connect to the server using the repository definition of the connector:

- Click the **Connect** button in the toolbar.
- Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Connect**.

Note: The `DeliveryTransport` property of the connector you are emulating must be set to the value `IDL` to connect to the server using the repository definition.

Emulating a connector using a configuration file

Do the following to connect to the server using a configuration file:

1. Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Connect with *.cfg**.
2. Navigate to and open the connector configuration file.

Note: If the `DeliveryTransport` property of the connector you are emulating is not set to the value `IDL` then and you select **Server > Connect** from the menu, then Integrated Test Environment presents the dialog to specify the configuration file to be used.

Emulating an access client

To emulate an access client, click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Connect**.

Confirming that a client has connected to the server

Use the following indicators to determine if a Client Simulator has successfully connected to the server:

- In the Outline view, the node for the port through which the client communicates turns green and reads "Ready".
- If the client is a connector, the message "[AppConnector: Connector has recovered]" is written to the Status Pane of the Client Simulator view.
- If the client is an access client, the message "Ready" is written to the Status Pane of the Client Simulator view.
- The **BO Type** and **BO Instance** drop-down menus become enabled on the Input Pane of the Client Simulator view.

For more information about the Status Pane, see "Using the Status Pane" on page 238.

For more information about the Input Pane, see "Using the Input Pane" on page 237.

Disconnecting from the server

To disconnect a Client Simulator that is emulating a connector from the server, click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Disconnect**.

A Client Simulator view also disconnects from the server when you close it. The only way to disconnect a Client Simulator view that is emulating an access client from the server is to close the Client Simulator view. For more information on closing Client Simulator views, see “Closing Client Simulator views” on page 235.

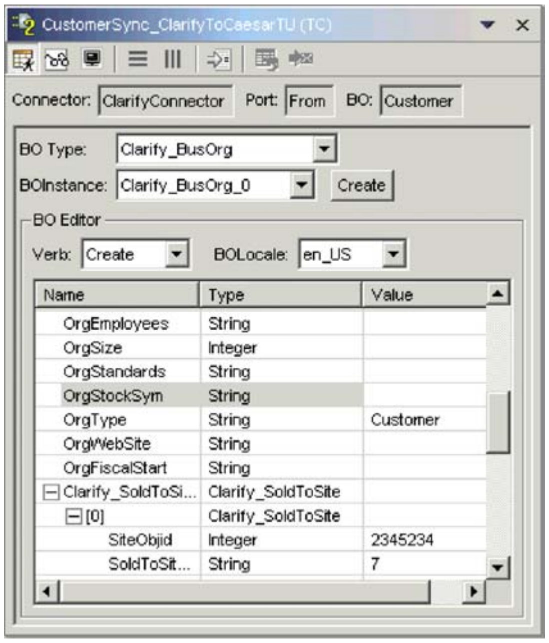
Client Simulator view panes and pane arrangements

The Client Simulator view has several panes you use to simulate the behavior of a connector or access client.

Using the Input Pane

You use the Input Pane in the Client Simulator view to work with and send business object requests. You can create business object instances, add data to their attributes, and send them as requests. You typically use the Input Pane when working with a Client Simulator view that is emulating a source client in an interface.

Figure 69 shows the Input Pane for a Client Simulator view that is emulating a source connector.



Connector: ClarifyConnector Port: From BO: Customer

BO Type: Clarify_BusOrg

BOInstance: Clarify_BusOrg_0

Verb: Create BOLocale: en_US

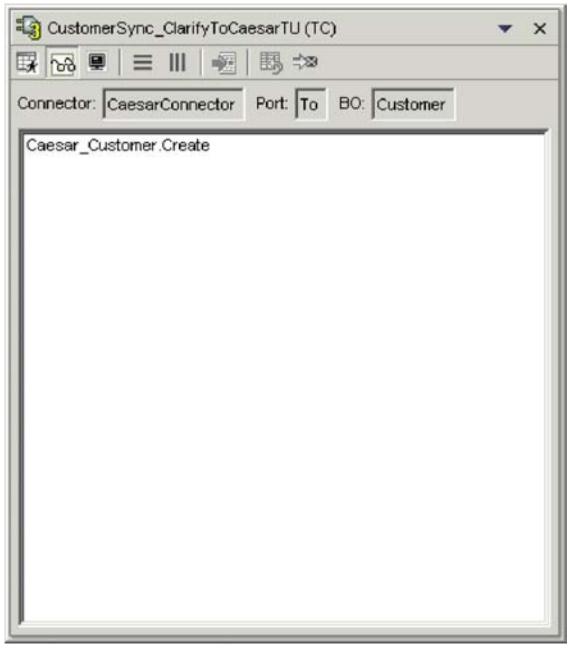
Name	Type	Value
OrgEmployees	String	
OrgSize	Integer	
OrgStandards	String	
OrgStockSym	String	
OrgType	String	Customer
OrgWebSite	String	
OrgFiscalStart	String	
⊟ Clarify_SoldToSi...	Clarify_SoldToSite	
⊟ [0]	Clarify_SoldToSite	
SiteObjid	Integer	2345234
SoldToSit...	String	7

Figure 69. Client Simulator view Input Pane

Using the Result Pane

You use the Result Pane in the Client Simulator view to work with business object requests that have been received and to send business object responses. You typically use the Result Pane when working with a Client Simulator view that is emulating a destination client in an interface.

Figure 70 on page 238 shows the Result Pane for a Client Simulator view that is emulating a destination connector.



Caesar_Customer.Create

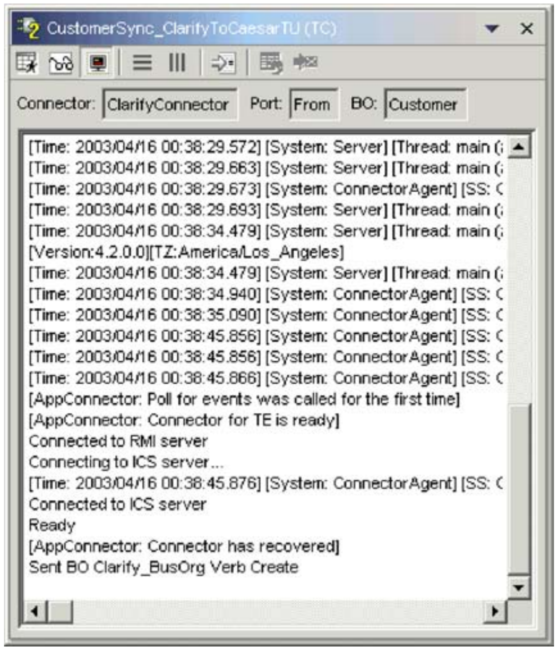
Figure 70. Client Simulator view Result Pane

Using the Status Pane

The Status Pane contains messages about the execution of the Client Simulator view. For instance, the message “[AppConnector: Connector has recovered]” is written to the Status Pane when a connector has successfully connected to the server.

You use the Status Pane primarily to confirm that Client Simulator has successfully connected to the server and to perform troubleshooting. For instance, if you send a business object request from a Client Simulator view and there is no subscription for the business object type, then a message is written to the Status Pane. This can help you to determine why a business object request does not appear in the Result Pane of a Client Simulator view emulating a destination client in an interface.

Figure 71 on page 239 shows the Status Pane for a Client Simulator view.



Connector: ClarifyConnector Port: From BO: Customer

```
[Time: 2003/04/16 00:38:29.572] [System: Server] [Thread: main (
[Time: 2003/04/16 00:38:29.663] [System: Server] [Thread: main (
[Time: 2003/04/16 00:38:29.673] [System: ConnectorAgent] [SS: C
[Time: 2003/04/16 00:38:29.693] [System: Server] [Thread: main (
[Time: 2003/04/16 00:38:34.479] [System: Server] [Thread: main (
[Version:4.2.0.0][TZ:America/Los_Angeles]
[Time: 2003/04/16 00:38:34.479] [System: Server] [Thread: main (
[Time: 2003/04/16 00:38:34.940] [System: ConnectorAgent] [SS: C
[Time: 2003/04/16 00:38:35.090] [System: ConnectorAgent] [SS: C
[Time: 2003/04/16 00:38:45.856] [System: ConnectorAgent] [SS: C
[Time: 2003/04/16 00:38:45.856] [System: ConnectorAgent] [SS: C
[Time: 2003/04/16 00:38:45.866] [System: ConnectorAgent] [SS: C
[AppConnector: Poll for events was called for the first time]
[AppConnector: Connector for TE is ready]
Connected to RMI server
Connecting to ICS server...
[Time: 2003/04/16 00:38:45.876] [System: ConnectorAgent] [SS: C
Connected to ICS server
Ready
[AppConnector: Connector has recovered]
Sent BO Clarify_BusOrg Verb Create
```

Figure 71. Client Simulator view Status Pane

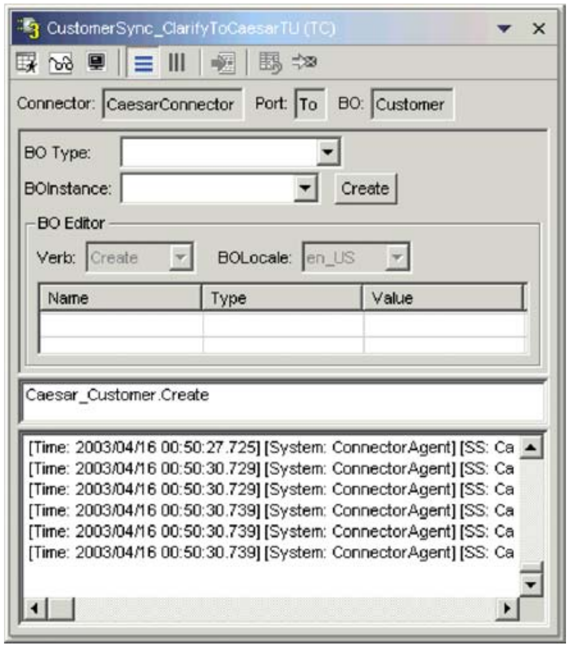
Arranging all panes vertically

You can arrange a Client Simulator view so that it displays the Input, Result, and Status panes simultaneously, stacked on top of one another. This view can be helpful when a Client Simulator view is emulating a client that has to handle both event notification and request processing responsibilities.

Do one of the following to arrange the Client Simulator view panes vertically:

- Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and choose **Window > Arrange Vertical** from the menu.
- Click the **Show all vertically** button in the title bar of the Client Simulator view.

Figure 72 on page 240 shows the Client Simulator view panes when they have been arranged vertically.


CustomerSync_ClarifyToCaesarTU (TC)
Connector: CaesarConnector Port: To BO: Customer
BO Type:
BOInstance:
Create
BO Editor
Verb: Create
BOLocale: en_US
Name
Type
Value

Caesar_Customer.Create

```
[Time: 2003/04/16 00:50:27.725] [System: ConnectorAgent] [SS: Ca
[Time: 2003/04/16 00:50:30.729] [System: ConnectorAgent] [SS: Ca
[Time: 2003/04/16 00:50:30.729] [System: ConnectorAgent] [SS: Ca
[Time: 2003/04/16 00:50:30.739] [System: ConnectorAgent] [SS: Ca
[Time: 2003/04/16 00:50:30.739] [System: ConnectorAgent] [SS: Ca
[Time: 2003/04/16 00:50:30.739] [System: ConnectorAgent] [SS: Ca
```

Figure 72. Arranging Client Simulator view panes vertically

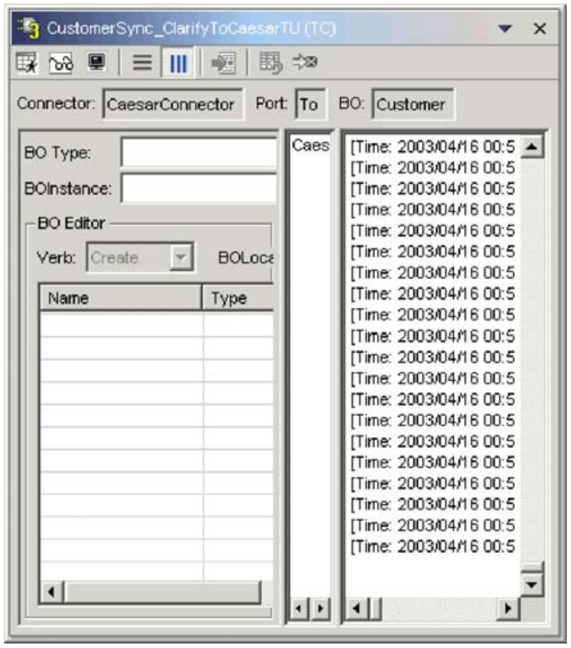
Arranging all panes horizontally

You can arrange a Client Simulator view so that it displays the Input, Result, and Status panes simultaneously from left to right. This view can be helpful when a Client Simulator view is emulating a client that has to handle both event notification and request processing responsibilities.

Do one of the following to arrange the Client Simulator view panes horizontally:

- Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and choose **Window > Arrange Horizontal** from the menu.
- Click the **Show all horizontally** button in the title bar of the Client Simulator view.

Figure 72 shows the Client Simulator view panes when they have been arranged horizontally.



```
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
[Time: 2003/04/16 00:5
```

Figure 73. Arranging Client Simulator view panes horizontally

Working with business object requests in the Client Simulator view

Request business objects are those that you send from a Client Simulator view when it is emulating a connector that is the source of the events that trigger an interface. Working with request business objects consists of creating a business object instance, populating it with data, and sending the request.

Creating request business objects

Do the following to create a new business object instance:

1. In the Input Pane, select the name of the business object you want to create from the **BO Type** drop-down menu.
2. Do one of the following:
   - Click **Create** next to the **BO Instance** field.
   - Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Edit > Create BO**.
3. When presented with the “New Instance” dialog, type a name for the instance in the **Enter Name** field.
4. Select the desired verb from the **Verb** drop-down menu.
5. Select the desired locale from the **BO Locale** drop-down menu.
6. Provide values for the simple attributes and child business objects within the top-level object, as described in “Setting values for business object attributes” on page 243.
7. Click **OK**.

Sending request business objects asynchronously

When a source client sends a request business object in asynchronous mode, it does not expect to get back a response business object. Once the request business object is dispatched, the source client’s role in the transaction is finished. The response business object is typically processed by the integration broker. The default mode for a Client Simulator view is asynchronous.

To send a business object asynchronously, do the following:

1. Create the request business object instance, as described in “Creating request business objects” on page 241.
2. Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Mode > Asynchronous**.

   Note: The Client Simulator view operates in “Asynchronous” mode by default, so you only have to perform this step if you previously were sending synchronous requests from the view. You do not have to set the mode before sending each request.
3. Do one of the following to send the request:
   - Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Send**.
   - Click the **Send BO** button in the toolbar of the Client Simulator view.

Sending business object requests synchronously

When a source client sends a request business object synchronously, it expects to get back a response business object from the integration broker after any destination applications have processed the request.

1. Create the request business object instance, as described in “Creating request business objects” on page 241.
2. Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Mode > Synchronous** to set the Client Simulator view to synchronous mode.
3. Do one of the following to send the request:
   - Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Send**.
   - Click the **Send BO** button in the toolbar of the Client Simulator view.
4. When presented with the “Select Collaboration” dialog, select the collaboration to which the business object should be sent from the **Collaboration** drop-down menu and click **OK**.

   The business object request is sent to the configured port of the collaboration object chosen for processing.

Sending business object requests in batch mode

In batch mode, the Client Simulator view lets you specify the number of instances of a particular business object you want to send, as well as one attribute in the top-level object —a primary key attribute, for example—that you want set to a unique value for each instance. The Client Simulator view copies the business object as many times as you have specified, incrementing the value of the single attribute you specified for each instance, and then sends each instance. This option allows you to create a large number of business objects quickly and easily.

If the selected attribute is a key field that participates in dynamic cross-referencing as part of an identity relationship, then you must guarantee that the initial value and all those that follow it are unique. Otherwise, the cross-referencing logic will fail, causing the request business objects to fail.

To ensure that the values are unique, you can use Relationship Manager or execute SQL statements against the table for the relationship participant as follows.

- Determine the highest current value for the participant and set the Initial Value field to an even higher value. The first business object instance in the batch and all those that follow will then be unique.
- Delete the existing table entries for the participant, thus guaranteeing that no entries have the same attribute value as any of the batch business objects.

To send business objects in batch mode, do the following:

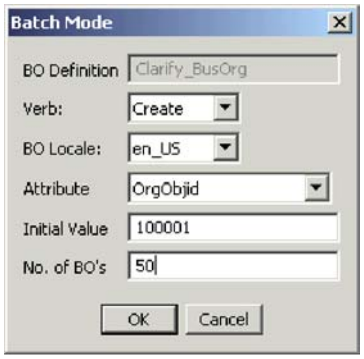
1. Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Server > Send Batch**.
2. In the "Batch Mode" window, select the desired verb from the **Verb** drop-down menu.
3. Select the desired locale from the **BO Locale** drop-down menu.
4. Select from the **Attribute** list the attribute in the top-level business object that you want incremented with each business object request in the batch.

   The selected attribute should typically be an attribute that uniquely identifies the business object, such as a primary key.
5. In the **Initial Value** field, type the starting value for the attribute to be incremented.
6. In the **No. of BO's** field, type the number of business object instances you want generated and sent.
7. Click **OK**.

   The Client Simulator view generates the number of business objects you specified, all identical with the exception of the one specified attribute, whose value is incremented for each instance.

Figure 74 shows a batch mode configuration in which:

- Fifty business objects are to be sent.
- The value of the attribute `OrgObjid` is to be incremented.
- The starting value for the attribute is 10000.



BO Definition	Clarify_BusOrg
Verb:	Create
BO Locale:	en_US
Attribute	OrgObjid
Initial Value	100001
No. of BO's	50

Figure 74. Sending business object requests in batch mode

Setting values for business object attributes

You can set values for business object attributes when creating a business object request and when editing a business object request that has been received, so that it can be returned as a response.

To set values for the attributes of a business object you intend to send as a request, you must use the Input Pane of the Client Simulator view. For more information, see "Using the Input Pane" on page 237.

To set values for the attributes of a business object you are editing to send as a response, you use the “Response BO” window. For more information, see “Editing response business objects” on page 245.

Setting values for simple attributes

To provide a value for a simple attribute, click its cell in the **Value** column of the Input Pane or the “Response BO” window and enter a value.

Adding child business objects

To add an instance of a child business object, right-click the attribute that represents the child object in the Input Pane or the “Response BO” window and select **Add Instance** from the context menu.

A plus sign (+) is added next to the attribute that represents the child business object to show that there is at least one child business object instance. If you expand the child object attribute, numbered entries are displayed for each instance. The individual instances also have plus signs (+) next to them, so you can expand them and set values for their attributes.

To add more child business object instances, right-click the attribute that represents the child object and select **Add Instance** from the context menu.

Note: If the **Card** property of the attribute that references the child business object is set to the value 1 (indicating it is of single-cardinality), then you will only be able to add one instance of the child object.

Removing child business objects

To remove an instance of a child business object, right-click the instance and select **Remove Instance** from the context menu.

To remove all instances of a child business object, right-click the attribute that represents the child business object and select **Delete All Instances** from the context menu.

Setting the verb of a child business object

You can set the verb of a child business object to test the effect that value has on the business process. This can be helpful when you are troubleshooting logic that involves the cross-referencing of child objects.

To set the verb of a child business object instance, right-click it and choose **Set Verb** from the context menu. When presented with the “Select Verb” prompt, selected the desired verb and click **OK**.

Resetting business object attributes to their default values

To reset the attributes of a business object request to their default values, click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Edit > Reset BO**.

To reset the attributes of a business object response to their default values, click the **Reset BO to default** button in the toolbar of the “Response BO” window.

Clearing business object attribute values

To clear the attributes of a business object request of their values, click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Edit > Clear BO**.

To clear the attributes of a business object response of their values, click the **Clear BO values** button in the toolbar of the “Response BO” window.

Working with response business objects

Response business objects are those that you send from a Client Simulator view when it is emulating a connector that is the recipient of business object requests in an interface. Working with request business objects consists of editing the values in the business object instance and sending the response back to the server.

If you create and send a business object request as described in “Working with business object requests in the Client Simulator view” on page 241 and it is successfully processed by the interface, then the business object appears in the Result Pane of any destination connectors in the interface. Figure 70 on page 238 shows a Result Pane in which a business object request has been received.

“Editing response business objects” describes how to edit a business object request that has been received, and “Sending response business objects” describes how to send the instance as a response.

Editing response business objects

When you receive a business object request in a Client Simulator view that is emulating a destination client in an interface, you frequently want to edit the values of the attributes. For instance, you will want to provide unique values for primary key attributes that participate in relationships, or you will want to modify the value of other attributes to test map or collaboration logic that responds differently depending on the exact values in the business object. Do the following to set the values of business object attributes:

1. Do one of the following to edit the business object request in the “Response BO” window:
   - Double-click the business object instance in the Result Pane.
   - Select the business object instance in the Result Pane, then click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Reply > Edit Response**.
2. Do one of the following to edit the attributes of the business object:
   - Use one of the techniques described in “Setting values for business object attributes” on page 243 to modify the values of the business object attributes.
   - Import business object data from a file as described in “Importing a response business object” on page 247.

     The ability to import saved data into a business object request is very useful in situations where you have to populate a response business object with data before sending it as a reply. Instead of manually typing a value for each attribute that requires response data, you can type the values once, export the business object (as described in “Exporting a response business object” on page 247), and then import the saved data on subsequent tests.

Sending response business objects

After you edit a request business object (if editing is needed), you send it back to the server as a reply.

Table 30 on page 246 lists the reply options and shows their corresponding connector return codes for both C++ and Java connectors. For more detailed information about C++ or Java Connector return codes, see the *Connector Development Guide for Java or C++*.

Table 30. Client Simulator view reply types and connector return codes.

Client Simulator view reply type	C++ connector return code	Java connector return code
Success	BON_SUCCESS	SUCCESS
Fail	BON_FAIL	FAIL
Multiple Hits	BON_MULTIPLE_HITS	MULTIPLE_HITS
Retrieve By Content Fail	BON_FAIL_RETRIEVE_BY_CONTENT	RETRIEVEBYCONTENT_FAILED
Not Found	BON_BO_DOES_NOT_EXIST	BO_DOES_NOT_EXIST
Value Duplicate	BON_VALDUPES	VALDUPES

To reply to a request business object, do the following:

1. Select the business object instance in the Request Pane.
2. Do one of the following:
   - Click the **Reply Success** or **Reply Fail** buttons in the toolbar of the Client Simulator view.
   - Click the down arrow in the right-hand corner of the title bar of the Client Simulator view, select the **Reply** submenu, and then select the desired type of reply.

Saving business objects

You can save a business object instance that you have created in the Input Pane so that it is available whenever the Client Simulator view is opened to emulate that particular connector definition. You can select the instance by name from **BO Instance** field instead of having to go through the process of creating a new request.

To save the business object instances currently listed in the **BO Instance** field, click the down arrow on the right-hand side of the title bar of the Client Simulator view and select **Edit > Save All BO's**.

The business object is saved to a file with the name of the business object instance, without any extension, in a directory named after the business object type in the `.bos` directory of the directory for the test project in which the test unit is defined. The file is saved in the same format used by Test Connector and Map Designer Express, so it can be used for testing in those tools as well.

Deleting business objects

To delete a business object instance from a Client Simulator view, click the down arrow on the right-hand side of the title bar of the Client Simulator view and select **Edit > Delete BO**.

Exporting business objects

You can export a business object instance to a file to conveniently archive data and to share with Technical Support. The test data file is saved with a `.bo` extension and can be used for testing with Test Connector and Map Designer Express as well as the Integrated Test Environment.

Exporting a request business object

Do the following to export a business object instance from the Input Pane of the Client Simulator view:

1. Select the business object instance you want to export in the **BO Instance** field.
2. Click the down arrow on the right-hand side of the title bar of the Client Simulator view and select **Edit > Export BO**.
3. Navigate to the directory in which you want to save the file and specify a name in the **File name** field.
4. Click **Save**.

Exporting a response business object

Do the following to export a business object instance from the "Response BO" window when editing a business object response in the Result Pane:

1. Click the **Save BO** button in the toolbar of the "Response BO" window.
2. Navigate to the directory in which you want to save the file and specify a name in the **File name** field.
3. Click **Save**.

Importing business objects

You can import a business object test data file that was saved from Map Designer Express or Test Connector, or exported from Integrated Test Environment.

Importing a request business object

Do the following to import a business object instance into the Input Pane of the Client Simulator view:

1. Click the down arrow on the right-hand side of the title bar of the Client Simulator view and select **Edit > Import BO**.
2. Navigate to and open the test data file.

Importing a response business object

Do the following to import a business object instance into the "Response BO" window when editing a business object response in the Result Pane:

1. Click the **Load BO** button in the toolbar of the "Response BO" window.
2. Navigate to and open the test data file.

Comparing business object instances

A Client Simulator view can compare two business objects of the same type and display the attributes that differ in value. You can use this function to view changes to a business object at different points in the execution of a transaction. For instance, you could compare a business object that has been sent to the server with the same business object after the server has processed it and returned it to the source connector. Do the following to compare two business object instances:

1. Select the request business object instance to be compared on the Input Pane.
2. Select the response business object to be compared on the Response Pane.
3. Click the down arrow in the right-hand corner of the title bar of the Client Simulator view and select **Reply > Compare BO's**.

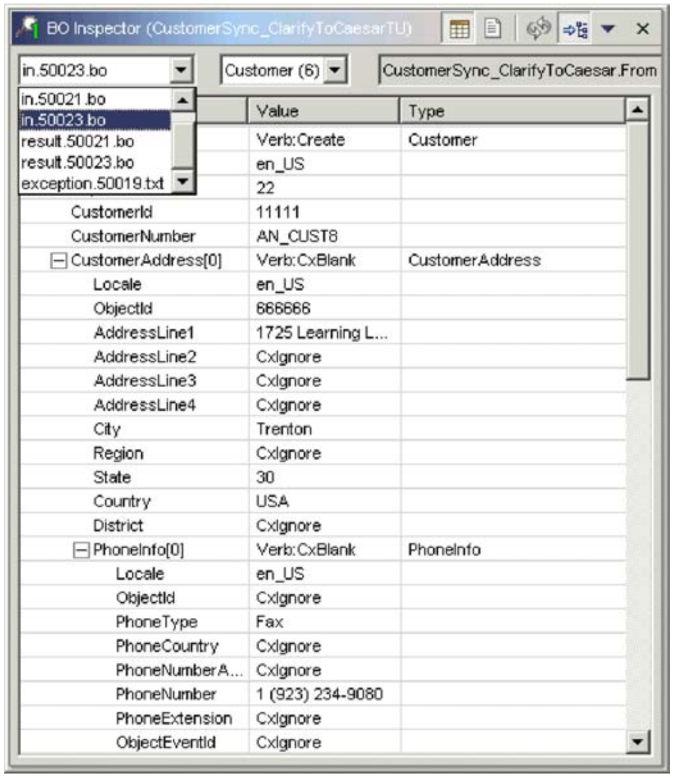
Using the BO Inspector view

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 Express.
- 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 Express.
- Exception messages related to failed flows.

Figure 75 shows a BO Inspector view that has captured three input business objects, two result business objects, and an exception message.



	Value	Type
	Verb:Create	Customer
	en_US	
	22	
CustomerId	11111	
CustomerNumber	AN_CUST8	
⊟CustomerAddress[0]	Verb:CxBlank	CustomerAddress
Locale	en_US	
ObjectId	666666	
AddressLine1	1725 Learning L...	
AddressLine2	CxIgnore	
AddressLine3	CxIgnore	
AddressLine4	CxIgnore	
City	Trenton	
Region	CxIgnore	
State	30	
Country	USA	
District	CxIgnore	
⊟PhoneInfo[0]	Verb:CxBlank	PhoneInfo
Locale	en_US	
ObjectId	CxIgnore	
PhoneType	Fax	
PhoneCountry	CxIgnore	
PhoneNumberA...	CxIgnore	
PhoneNumber	1 (923) 234-9080	
PhoneExtension	CxIgnore	
ObjectEventId	CxIgnore	

Figure 75. Business object inspector view

This section describes how to use business object tracing and BO Inspector views.

Business object trace behavior

The following statements describe the behavior related to BO Markers in the Test Unit editor and BO Inspector views as a business object is processed by an interface:

- When you send a business object request from a source connector, the BO Marker attached to the connector icon in the Test Unit editor is filled with green and the number 1 is displayed above the magnifying glass illustration in the BO Marker. This indicates that it has processed one business object.

You can show the BO Inspector view for this BO Marker at this point and see the generic business object produced by the inbound map. It is identified with a prefix of "in" in the drop-down menu in the upper-left of the BO Inspector view.

- The business object is processed by subsequent components in the interface (such as the collaboration object) and if there are no errors then it is processed by the outbound map associated with the destination connector in the interface.

  When this occurs the BO Marker attached to the destination connector icon in the Server Context Overlay is filled with green and the number 1 is displayed above the magnifying glass illustration in the BO Marker, indicating that it, too, has processed the business object.

  You can show the BO Inspector view for this BO Marker at this point and see the generic business object that was processed by the outbound map. It is identified with a prefix of "out" in the drop-down menu in the upper-left of the BO Inspector view.
- If you reply to the request business object in the destination connector at this point the interface completes its processing. The number 1 is displayed below the magnifying glass illustration in the BO Marker, indicating that the generic object produced by the inbound map of the destination connector has been processed.

  At this point you can view the generic business object in the first BO Inspector view. It is identified with a prefix of "result" in the drop-down menu in the upper-left of the BO Inspector view. Because the same BO Inspector view also contains data for the "in" business object instance, you must select the "result" instance from the drop-down menu.
- If an error occurs during the execution of the interface a red circle is drawn around the magnifying glass of the BO Marker. You can select the error—identified with a prefix of "exception"—in the BO Inspector view associated with the BO Marker. You must change the BO Inspector view to Text View in order to view the error text.

Showing and closing BO Inspector views

The following sections describe the numerous ways you can show and close BO Inspector views.

The "Default interface position" column in Table 31 lists the views whose default positions in the interface are overlaid with the BO Inspector views in the "BO Viewer number" column.

Table 31. Default BO Inspector view positions

BO Viewer number	Default interface position
BO Viewer 1	Outline
BO Viewer 2	Properties
BO Viewer 3	Dependency

Showing BO Inspector views

Use one of the following techniques to show BO Inspector views:

- Follow the instructions in "Showing and closing BO Inspector views using the Outline view" on page 219 to show a BO Inspector view in the Outline view.

- Follow the instructions in “Showing BO Inspector views” on page 232 to show a BO Inspector view in the Test Unit view.
- Select **Integrated Test Environment > Show All BO Views** from the menu bar to show all three BO Inspector views.
- Select **Perspective > Show View >** ***x*****:BO Inspector** from the menu bar to launch a specific BO Inspector view, where ***x*** is the number of the BO Inspector view instance you want to show.

Closing BO Inspector views

Use one of the following techniques to close all BO Inspector views:

- Follow the instructions in “Showing and closing BO Inspector views using the Outline view” on page 219 to close a BO Inspector view in the Outline view.
- Select **Integrated Test Environment > Close All BO Views** from the menu bar to close all open BO Inspector views.
- Either click the **Close** button in the right-hand side of the BO Inspector view title bar, or right-click the BO Inspector view title bar and select **Close** from the context menu.

Refreshing business object data

As the components of an interface process a business object, the counters of the BO Markers in the Test Unit editor get updated. This indicates that you can update the business object data in the BO Inspector view associated with that BO Marker to view the most recently produced business object.

Do one of the following to update the business object data in a BO Inspector view:

- Click the **Refresh BO data** button in the toolbar of the BO Inspector view.
- Click the down arrow in the right-hand corner of the title bar of the BO Inspector view and select **Refresh**.

After the business object data is refreshed, you can select the most current business object instance from the drop-down menu in the upper-left-hand corner of the BO Inspector view.

Refreshing business object markers

For information on refreshing BO Markers in the Server Context Overlay, see “Refreshing BO Markers” on page 232.

BO Inspector view styles

There are several view styles you can use with the BO Inspector view.

Using the Table view style

The Table view style presents the business object data in a table, as it appears in the Input Pane of the Client Simulator view or in the “Response BO” window.

Figure 75 on page 248 shows the BO Inspector view with the Table view style.

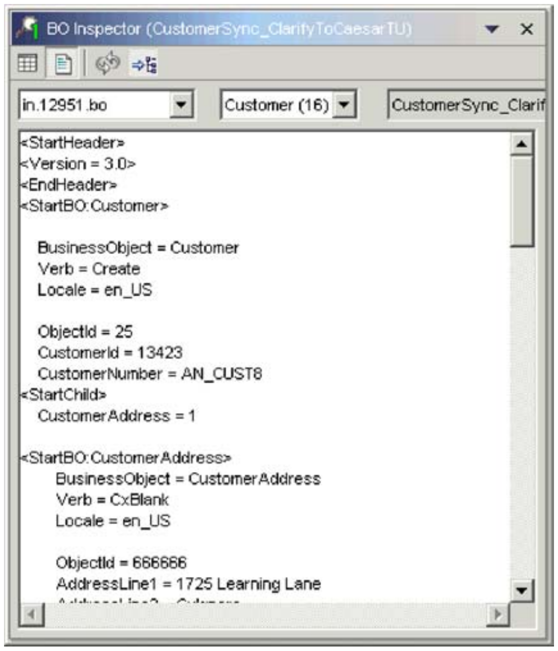
Do one of the following to use the Table view style:

- Click the **Show table view** button in the toolbar of the BO Inspector view.
- Click the down arrow in the right-hand side of the title bar of the BO Inspector view and select **Show Table View**.

Using the Text view style

The Text view style presents the business object data in the text markup format in which it is stored when exported from the Client Simulator view or “Response BO” window.

Figure 76 shows the BO Inspector view with the Text view style.

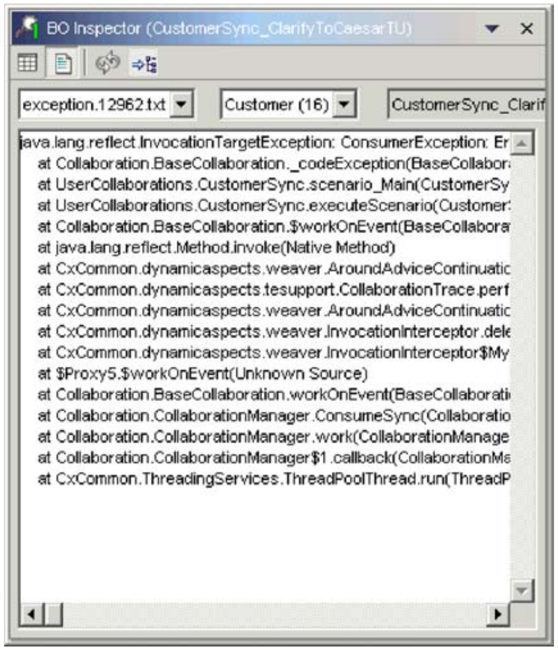


```
<StartHeader>
<Version = 3.0>
<EndHeader>
<StartBO:Customer>

   BusinessObject = Customer
   Verb = Create
   Locale = en_US

   ObjectId = 25
   CustomerId = 13423
   CustomerNumber = AN_CUST8
<StartChild>
   CustomerAddress = 1

<StartBO:CustomerAddress>
      BusinessObject = CustomerAddress
      Verb = CxBlank
      Locale = en_US

      ObjectId = 666666
      AddressLine1 = 1725 Learning Lane
```

Figure 76. BO Inspector view with business object data in text view style

To view an exception message in a BO Inspector view, you must change it to the Text view style; the BO Inspector view cannot display an exception message in the Table view style.

Figure 77 on page 252 shows an exception message in the BO Inspector view with the Text view style.



```
java.lang.reflect.InvocationTargetException: ConsumerException: Er
   at Collaboration.BaseCollaboration._codeException(BaseCollabor
   at UserCollaborations.CustomerSync.scenario_Main(CustomerSy
   at UserCollaborations.CustomerSync.executeScenario(Customer
   at Collaboration.BaseCollaboration.$workOnEvent(BaseCollabora
   at java.lang.reflect.Method.invoke(Native Method)
   at CxCommon.dynamicaspects.weaver.AroundAdviceContinuatic
   at CxCommon.dynamicaspects.tesupport.CollaborationTrace.perf
   at CxCommon.dynamicaspects.weaver.AroundAdviceContinuatic
   at CxCommon.dynamicaspects.weaver.InvocationInterceptor.dele
   at CxCommon.dynamicaspects.weaver.InvocationInterceptor$My
   at $Proxy5.$workOnEvent(Unknown Source)
   at Collaboration.BaseCollaboration.workOnEvent(BaseCollaborati
   at Collaboration.CollaborationManager.ConsumeSync(Collaboratio
   at Collaboration.CollaborationManager.work(CollaborationManage
   at Collaboration.CollaborationManager$1.callback(CollaborationMa
   at CxCommon.ThreadingServices.ThreadPoolThread.run(ThreadP
```

Figure 77. BO Inspector view with exception information in text view style

Do one of the following to use the Text view style:

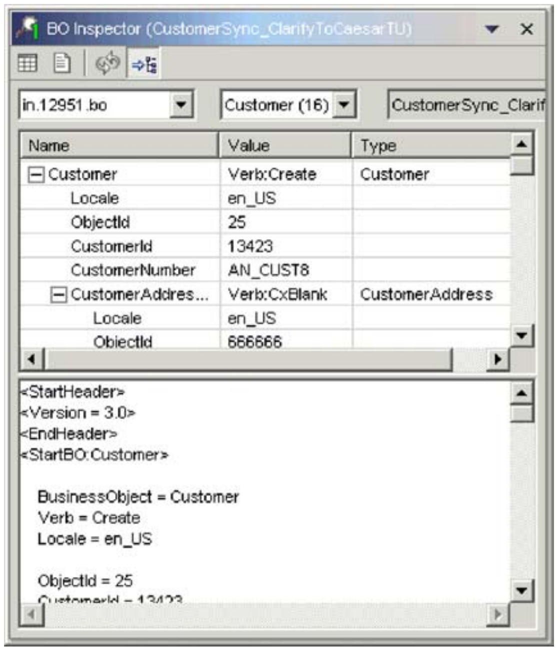
- Click the **Show text view** button in the toolbar of the BO Inspector view.
- Click the down arrow in the right-hand side of the title bar of the BO Inspector view and select **Show Text View**.

Arranging the views vertically

You can arrange a BO Inspector view so that it displays the Text view beneath the Table view at the same time.

To arrange the BO Inspector view panes vertically, click the down arrow in the right-hand corner of the title bar of the BO Inspector view and select **Arrange Vertical** from the menu.

Figure 78 on page 253 shows the BO Inspector view panes when they have been arranged vertically.



Name	Value	Type
Customer	Verb:Create	Customer
Locale	en_US	
ObjectId	25	
CustomerId	13423	
CustomerNumber	AN_CUST8	
CustomerAddres...	Verb:CxBlank	CustomerAddress
Locale	en_US	
ObjectId	666666	

```
<StartHeader>
<Version = 3.0>
<EndHeader>
<StartBO:Customer>

  BusinessObject = Customer
  Verb = Create
  Locale = en_US

  ObjectId = 25
  CustomerId = 13423
```

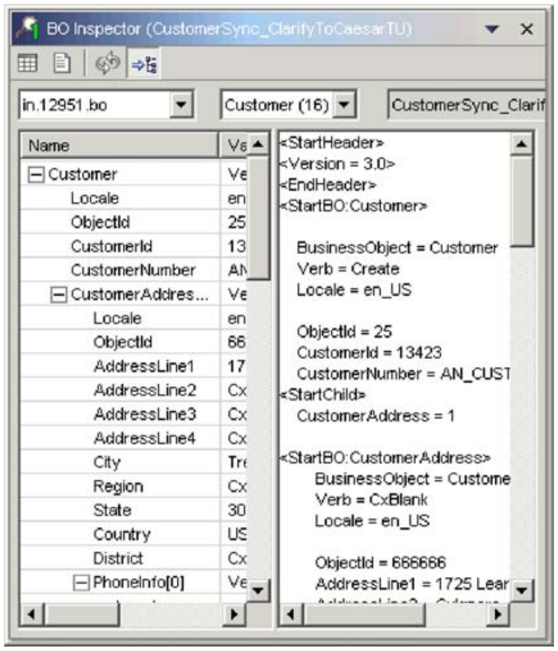
Figure 78. BO Inspector view panes arranged vertically

Arranging the views horizontally

You can arrange a BO Inspector view so that it displays the Text view to the right of the Table view at the same time.

To arrange the BO Inspector view panes horizontally, click the down arrow in the right-hand corner of the title bar of the BO Inspector view and select **Arrange Horizontal** from the menu.

Figure 79 on page 254 shows the BO Inspector view panes when they have been arranged horizontally.


BO Inspector (CustomerSync_ClarifyToCaesarTU)

in.12951.bo | Customer (16) | CustomerSync_Clarif

Name	Va
Customer	Ve
Locale	en
ObjectId	25
CustomerId	13
CustomerNumber	AN
CustomerAddres...	Ve
Locale	en
ObjectId	66
AddressLine1	17
AddressLine2	Cx
AddressLine3	Cx
AddressLine4	Cx
City	Tr
Region	Cx
State	30
Country	US
District	Cx
PhoneInfo[0]	Ve

```
<StartHeader>
<Version = 3.0>
<EndHeader>
<StartBO:Customer>

  BusinessObject = Customer
  Verb = Create
  Locale = en_US

  ObjectId = 25
  CustomerId = 13423
  CustomerNumber = AN_CUST
<StartChild>
  CustomerAddress = 1

<StartBO:CustomerAddress>
    BusinessObject = Custome
    Verb = CxBlank
    Locale = en_US

    ObjectId = 666666
    AddressLine1 = 1725 Lear
```

Figure 79. BO Inspector view panes arranged horizontally

Saving test data from a BO Inspector view

Although there is no built-in mechanism for saving test data in a BO Inspector view, you can use the following workaround:

1. Switch the BO Inspector view to the Text view style as described in “Using the Text view style” on page 251.
2. Select all of the text in the Text view pane.
3. Use the keyboard shortcut **Ctrl+C** to copy the text.
4. Paste the copied text into a text file and save it with a `.bo` extension.

   You can use the saved text file in Map Designer Express, Test Connector, and Integrated Test Environment.

Configuring business object tracing preferences

Business object tracing can be a very resource intensive task. By default it is configured to turn off after two hours, though you can change this timeout value. Furthermore, you can configure Integrated Test Environment so that it does not automatically update the counters of BO Markers as business objects are processed, which can be a resource-intensive task in high-volume transactions. Do the following to configure preferences related to business object tracing:

1. Select **Window > Preferences** from the menu bar.
2. Expand **Integrated Test Environment** and then select **Test Server**.
3. Do the following in the “BO Trace” pane to configure your business object tracing preferences:
   a. Type the number of minutes after which you want business object tracing to turn off in the **BO trace timeout** field.
   b. To have Integrated Test Environment automatically update the counters of BO Markers as business objects are processed, leave the **Enable auto update** checkbox enabled.

4. Click **OK**.

Performing a test using Integrated Test Environment

Whereas most of this chapter described subtasks you perform as part of a test or specific interface elements of the Integrated Test Environment perspective, this section describes the workflow you typically follow to perform a test. It incorporates most of the subtasks and involves most of the interface elements. It recommends the most efficient and effective techniques in situations where there are multiple ways of accomplishing a subtask. Do the following to test an interface using Integrated Test Environment:

1. Although you can deploy components to an InterChange Server Express 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.

   For information on deploying components using System Manager, see "Deploying components to a server" on page 70.

   For information on deploying components using repos_copy, see Chapter 6, "Using repos_copy," on page 109.

   For information on deploying components by using Integrated Test Environment (which is not recommended), reference one of the following sections:
   - "Managing the repository using the Outline view" on page 217
   - "Using the Task Manager view" on page 220
   - "Managing the repository using the Test Unit view" on page 231
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 in the *System Administration Guide*
   - InterChange Server Component Management view, as described in "Managing component states in the repository" on page 84
   - Outline view, as described in "Managing component states using the Outline view" on page 217
   - "Managing component states using the Test Unit view" on page 231
3. Register the InterChange Server Express you want to test with as a "Local Test Server", as described in "Registering an InterChange Server Express instance" on page 43.
4. Start the Integrated Test Environment perspective as described in "Starting Integrated Test Environment" on page 202.
5. Select the server you want to test with as described in "Selecting a server configuration" on page 204.

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. Configure RMI settings for your environment as described in "Configuring RMI settings" on page 205. If you do not do this properly, you will not be able to test interfaces using Integrated Test Environment.
7. Create a test project to contain the test unit. For more information, see "Creating a test project" on page 211.
8. Create a test unit for the interface you want to test. For more information, see "Creating a test unit within Integrated Test Environment" on page 212.
9. If you plan to deploy the components in the interface you are testing using Integrated Test Environment, do the following:
   - Add the user dependents for the interface as described in "Adding user dependents" on page 216.
   - Follow the instructions in "Configuring InterChange Server Express to start in design mode" on page 208.
10. Make sure the IBM Java Object Request Broker is started. For more information, see the *System Installation Guide for Windows or Unix*.
11. Use the Task Manager view to start the server, bind the Integrated Test Environment agent to it, and connect the Integrated Test Environment to it as described in "Using the Task Manager view" on page 220.
12. Enable the Server Context Overlay as described in "Enabling and disabling the Server Context Overlay" on page 230.
13. 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).

    You can do one of the following to show and organize Client Simulator views for the interface:
   - Execute the "Start connector views" task as described in Table 25 on page 221 and then drag-and-drop the views to the positions you prefer.
   - Follow the instructions in "Using, showing, and closing Client Simulator views using the Test Unit view" on page 231 to choose which Client Simulator view you want to use for each connector definition.
14. Connect the Client Simulator views to the server as described in "Connecting a Client Simulator view to the server" on page 235.

    Confirm that the clients connect to the server successfully as described in "Confirming that a client has connected to the server" on page 236.

    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 as described in "Using the Input Pane" on page 237 and configure the Client Simulator view for the destination connector to use the Result Pane as described in "Using the Result Pane" on page 237.
15. 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. For more information on starting the business object tracing task, see Table 25 on page 221.
16. Use the Outline view to confirm that the interface is ready for testing, as described in "Verifying test readiness using the Outline view" on page 218.

17. Do the following to create and send a business object request from the source connector:
    a. Create a business object instance to send as a request as described in "Creating request business objects" on page 241.
    b. Set values for the attributes of the business object instance as described in "Setting values for business object attributes" on page 243.
    c. Save the business object instance to a file to be used in subsequent tests as described in "Saving business objects" on page 246.
    d. Send the business object instance as a request as described in "Sending request business objects asynchronously" on page 241 or "Sending business object requests synchronously" on page 242 as appropriate.
18. Use the InterChange Server Express Console view to observe the processing of the business object as described in "Using the Integrated Test Environment Console and InterChange Server Express Console views" on page 228.
19. Do the following to examine the business object as different components finish processing it:
    - Show BO Inspector views for the BO Markers in the interface as described in "Showing BO Inspector views" on page 249.
    - Examine the business object instances in the BO Inspector views as described in "Using the Table view style" on page 250.
20. Edit the response business object in the Result Pane of the destination Client Simulator view as described in "Editing response business objects" on page 245.
21. Send the business object response as a reply as described in "Sending response business objects" on page 245.
22. Repeat steps 17 through 21 to test the interface again, or repeat steps 7 on page 256 through 21 to test another interface.

Chapter 13. Using Collaboration Debugger

Collaboration Debugger is a workbench perspective that allows you to specify points at which a collaboration process should pause while executing, and allows you to inspect the data of the flow being processed at those points.

To debug a collaboration, you must do a number of tasks such as attach Collaboration Debugger to a collaboration object, configure breakpoints, manage the progress of the flow through the collaboration, and inspect the data of the flow. The following characteristics describe the use of Collaboration Debugger:

- You only have to perform some of the tasks a single time. For instance, you typically only have to attach Collaboration Debugger to the collaboration object once.
- You typically perform some tasks multiple times. For instance, you usually manage the progress of the flow multiple times during the course of its processing.
- You might not perform some tasks during the processing of every flow.

Due to the non-linear nature of debugging a collaboration, the sections of this chapter describe the interfaces of the perspective or certain tasks that you perform in the order in which you most commonly use the interfaces or perform the tasks. "Debugging a collaboration" on page 277 describes the workflow of debugging a collaboration to provide a more linear guide to the interfaces and tasks.

This chapter contains the following sections:

Starting Collaboration Debugger

To start Collaboration Debugger, do the following:

1. Select **Start > Programs > IBM WebSphere Business Integration Express> Toolset Express> Administrative > System Manager**.
2. Select **Window > Open Perspective > Other** from the menu bar.
3. Select `Collaboration Debugger` from the list of perspectives and then click **OK**.

   The workbench starts and appears. Figure 10 on page 40 shows the Collaboration Debugger perspective and "Collaboration Debugger interface" on page 261 describes the interface and its elements.

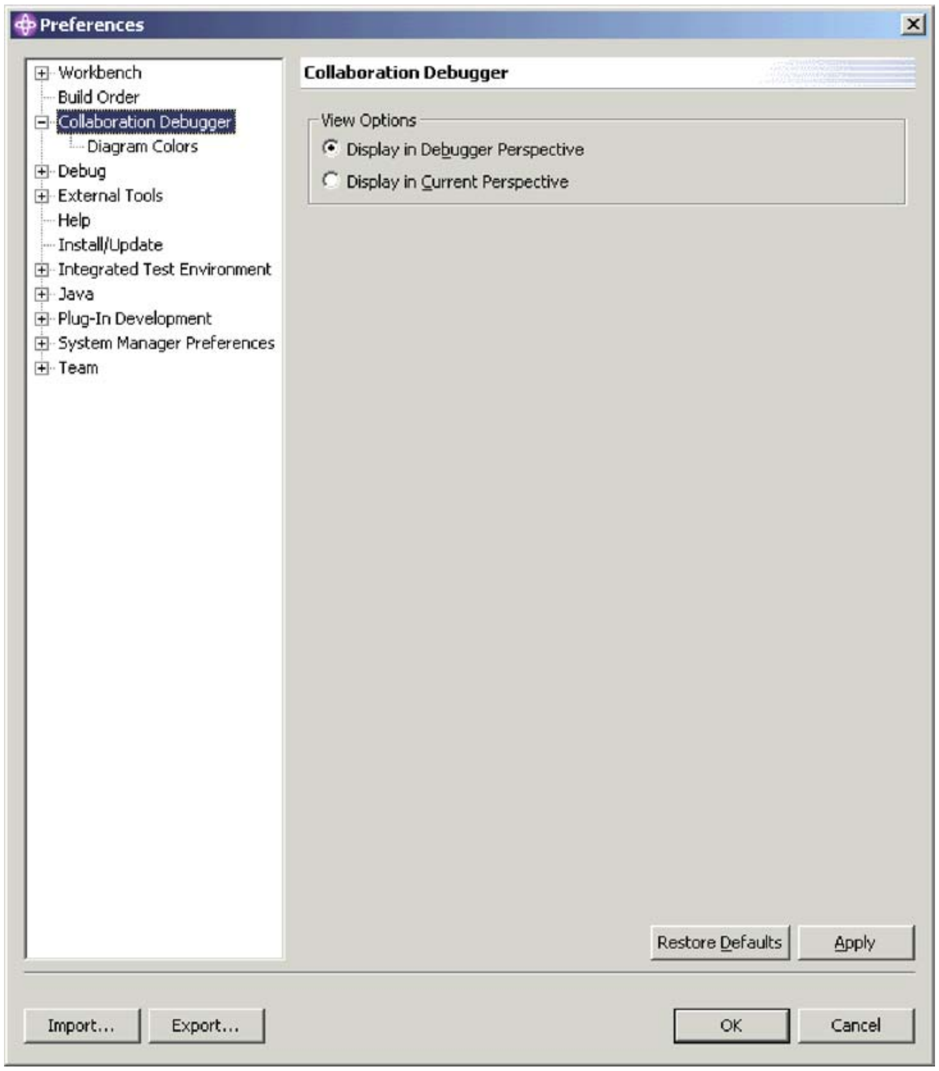
Configuring Collaboration Debugger view preferences

You can configure Collaboration Debugger so that its interfaces either open in the perspective you are currently using, or in Collaboration Debugger as designed. The benefit of having the interfaces open in the perspective you are currently using is that you do not have to navigate between perspectives to use the Collaboration Debugger views while also performing tasks like deploying components or sending events. The benefit of having the interfaces open in Collaboration Debugger is that you reduce the number of interfaces that have to share the same positions in the workbench.

Do the following to configure your preferences for the Collaboration Debugger views:

1. Select **Window > Preferences** from the menu bar of the workbench.
2. Select **Collaboration Debugger**.

   Figure 80 shows the view preferences for Collaboration Debugger.



Collaboration Debugger

View Options

[✓ Display in Debugger Perspective]

Display in Current Perspective

Figure 80. Collaboration Debugger view preferences

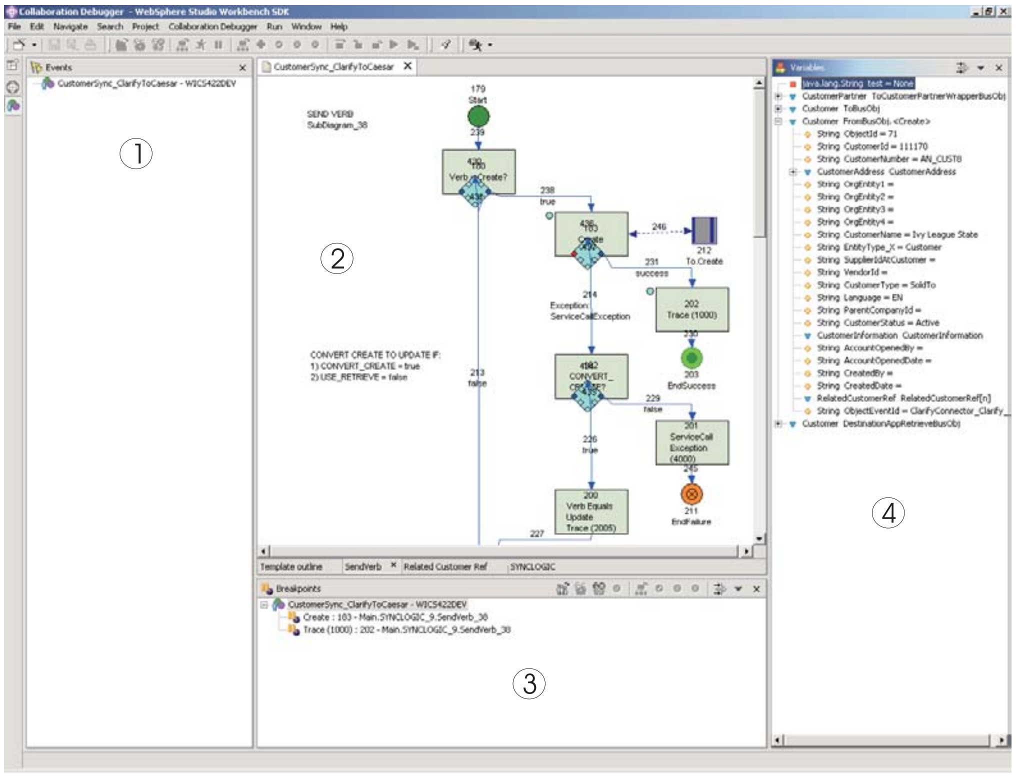
3. Enable the **Display in Debugger Perspective** radio button to have the Collaboration Debugger views open in Collaboration Debugger itself.

Enable the **Display in Current Perspective** radio button to have the Collaboration Debugger views open in the perspective you currently have active.

4. Click **OK**.

Collaboration Debugger interface

The Collaboration Debugger perspective has several views and one editor in the default configuration it opens with. Figure 81 shows the default Collaboration Debugger perspective.



Collaboration Debugger - WebSphere Studio Workbench SDK
Events
CustomerSync_ClarifyToCaesar
Breakpoints
Variables
1
2
3
4

Figure 81. Collaboration Debugger perspective

Table 32 on page 261 identifies the interface elements of the Collaboration Debugger perspective, indicated by the numbers in Figure 81.

Table 32. Collaboration Debugger perspective interface elements

Interface element number	Interface element name
1	“Events view”
2	“Collaboration template editor” on page 262
3	“Breakpoints view” on page 263
4	“Variables view” on page 264

Events view

The “Events” view lists the flows that are delivered for processing to the collaboration object you are debugging. It provides operations to affect the processing of the flow and to locate the node that is currently executing.

Figure 82 shows the “Events” view.

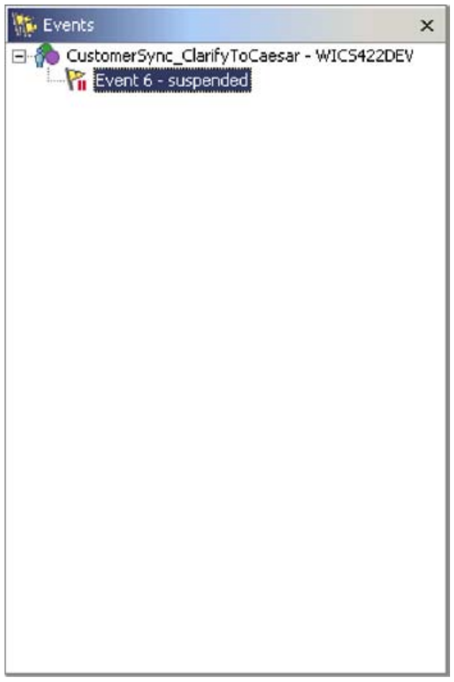


Figure 82. Events view

For more information about working with events and using the “Events” view, see “Working with events” on page 270.

Collaboration template editor

The “Collaboration template” editor is a read-only editor that displays the diagrams of the collaboration template upon which the collaboration object you are debugging is based. It allows you to view the design of the business process and perform many of the tasks associated with debugging a collaboration.

Figure 83 on page 263 shows the “Collaboration template” editor.

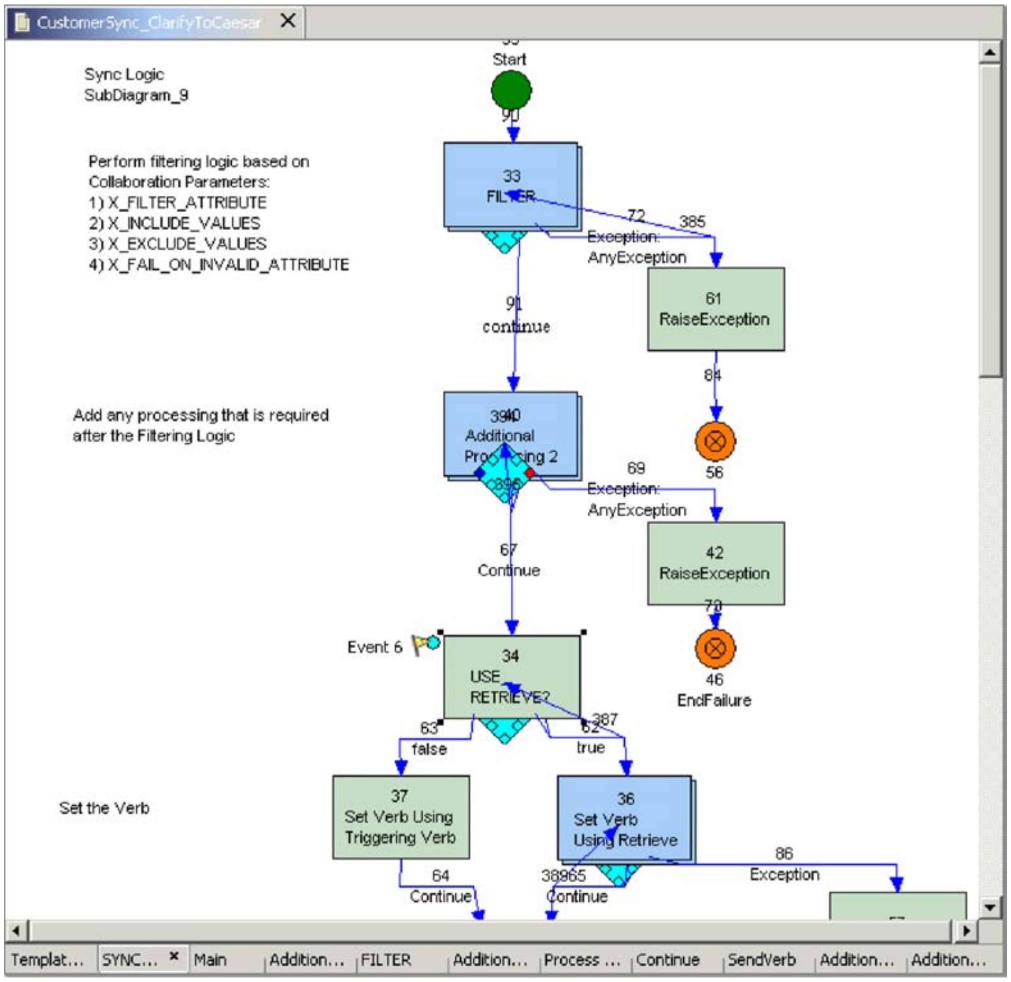

CustomerSync_ClarifyToCaesar
Sync Logic
SubDiagram_9
Perform filtering logic based on
Collaboration Parameters:
1) X_FILTER_ATTRIBUTE
2) X_INCLUDE_VALUES
3) X_EXCLUDE_VALUES
4) X_FAIL_ON_INVALID_ATTRIBUTE
Start
33 FILTER
72 385 Exception: AnyException
61 RaiseException
91 continue
Add any processing that is required after the Filtering Logic
Additional Processing 2
69 Exception: AnyException
56
42 RaiseException
67 Continue
46 EndFailure
Event 6
34 USE_RETRIEVE?
63 false
62 387 true
Set the Verb
37 Set Verb Using Triggering Verb
36 Set Verb Using Retrieve
64 Continue
38965 Continue
86 Exception
Templat... SYNC... Main Addition... FILTER Addition... Process ... Continue SendVerb Addition... Addition...

Figure 83. Collaboration template editor

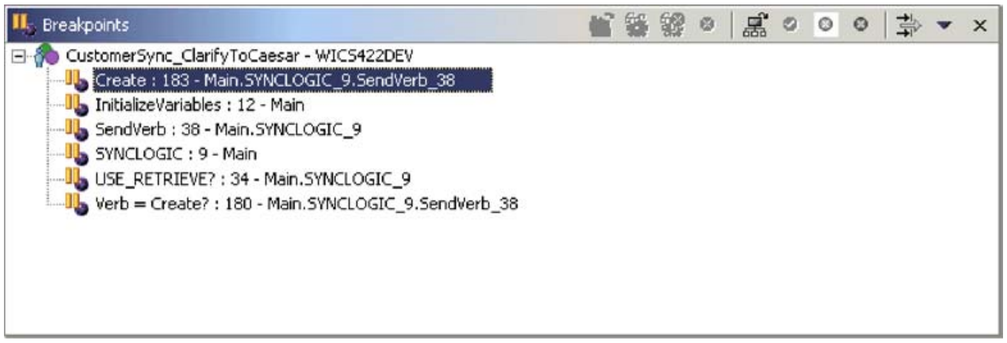
The following sections contain more information about the “Collaboration template” editor and the tasks you can perform using it:

- “Using the collaboration template editor” on page 268
- “Working with breakpoints” on page 271
- “Performing debugging operations” on page 274

Breakpoints view

The “Breakpoints” view lets you manage breakpoints in the collaboration object and manage the processing of a flow as it interacts with those breakpoints.

Figure 84 on page 264 shows the “Breakpoints” view.



CustomerSync_ClarifyToCaesar - WICS422DEV
Create : 183 - Main.SYNCLOGIC_9.SendVerb_38
InitializeVariables : 12 - Main
SendVerb : 38 - Main.SYNCLOGIC_9
SYNCLOGIC : 9 - Main
USE_RETRIEVE? : 34 - Main.SYNCLOGIC_9
Verb = Create? : 180 - Main.SYNCLOGIC_9.SendVerb_38

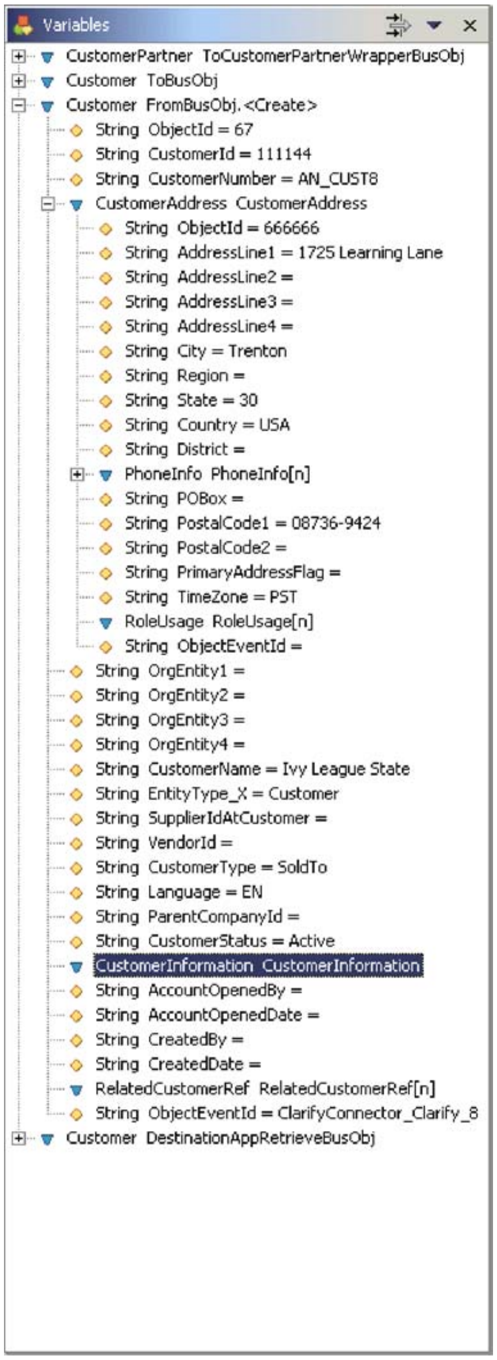
Figure 84. Breakpoints view

For more information about the tasks you typically perform using the "Breakpoints" view, see the following sections:
- "Working with breakpoints" on page 271
- "Performing debugging operations" on page 274

Variables view

The "Variables" view displays information about the flow being processed as it pauses in response to debugging operations. It lists all the variables of `BusObj` or Java primitive types that are declared in the "Declarations" section of the collaboration template. As you perform debugging operations to manage a flow through the collaboration, the "Variables" view updates the variables it lists with their values at the node being executed at the time.

Figure 85 on page 265 shows the "Variables" view.



```
Variables
+ CustomerPartner  ToCustomerPartnerWrapperBusObj
+ Customer  ToBusObj
- Customer  FromBusObj.<Create>
    String  ObjectId = 67
    String  CustomerId = 111144
    String  CustomerNumber = AN_CUST8
  - CustomerAddress  CustomerAddress
      String  ObjectId = 666666
      String  AddressLine1 = 1725 Learning Lane
      String  AddressLine2 =
      String  AddressLine3 =
      String  AddressLine4 =
      String  City = Trenton
      String  Region =
      String  State = 30
      String  Country = USA
      String  District =
    + PhoneInfo  PhoneInfo[n]
      String  POBox =
      String  PostalCode1 = 08736-9424
      String  PostalCode2 =
      String  PrimaryAddressFlag =
      String  TimeZone = PST
      RoleUsage  RoleUsage[n]
      String  ObjectEventId =
    String  OrgEntity1 =
    String  OrgEntity2 =
    String  OrgEntity3 =
    String  OrgEntity4 =
    String  CustomerName = Ivy League State
    String  EntityType_X = Customer
    String  SupplierIdAtCustomer =
    String  VendorId =
    String  CustomerType = SoldTo
    String  Language = EN
    String  ParentCompanyId =
    String  CustomerStatus = Active
    CustomerInformation  CustomerInformation
    String  AccountOpenedBy =
    String  AccountOpenedDate =
    String  CreatedBy =
    String  CreatedDate =
    RelatedCustomerRef  RelatedCustomerRef[n]
    String  ObjectEventId = ClarifyConnector_Clarify_8
+ Customer  DestinationAppRetrieveBusObj
```

Figure 85. Variables view

For more information about the "Variables" view and the tasks that affect its contents, see the following sections:

- "Working with events" on page 270
- "Performing debugging operations" on page 274
- "Working with variables" on page 276

Attaching and detaching Collaboration Debugger

You must attach Collaboration Debugger to a collaboration object to start a debugging session, set breakpoints, and manage a flow through the business process.

The collaboration object must be active to attach Collaboration Debugger to it. For more information about component states and how to change them, see the *System Administration Guide*.

When you are finished debugging a collaboration object, detach Collaboration Debugger to resume the normal processing of flows. If you stop a collaboration object then Collaboration Debugger automatically detaches from it.

Attaching Collaboration Debugger

You can attach Collaboration Debugger to a collaboration object in either System Manager or Integrated Test Environment, as described in the following sections:

- "Attaching Collaboration Debugger in System Manager"
- "Attaching Collaboration Debugger in Integrated Test Environment"

Attaching Collaboration Debugger in System Manager

Do the following to attach Collaboration Debugger to a collaboration object in System Manager:

1. In the InterChange Server Express Component Management view, expand the server instance in which the collaboration object exists, then expand the Collaboration Objects folder.
2. Select the collaboration objects that you want to attach Collaboration Debugger to. You can use standard Windows selection techniques to select multiple user projects for deployment at the same time, such as the following:
   - Hold down **Shift** to select contiguous items.
   - Hold down **Ctrl** to select non-contiguous items.
3. Right-click one of the selected collaboration objects and choose **Attach Debugger** from the context menu

Attaching Collaboration Debugger in Integrated Test Environment

To attach Collaboration Debugger to a collaboration object in Integrated Test Environment, right-click the center icon for the collaboration object in the Test Unit view and choose **Attach Debugger** from the context menu.

For more information about the Test Unit view, see "Using the Test Unit view" on page 229.

Attaching Collaboration Debugger in Collaboration Debugger

Do one of the following to attach Collaboration Debugger to a collaboration object in Collaboration Debugger itself:

- Right-click anywhere in the "Collaboration template" editor and choose **Attach Debugger** from the context menu
- Right-click the collaboration object in the "Breakpoints" view or the "Events" view and choose **Attach Debugger** from the context menu
- Select the collaboration object in the "Breakpoints" view or the "Events" view and select **Collaboration Debugger > Attach Debugger** from the menu bar

- Select the collaboration object in the “Breakpoints” view and click **Attach Debugger** in the toolbar of the “Breakpoints” view
- Select the collaboration object in the “Breakpoints” view, click the drop-down arrow in the upper-right-hand corner of the title bar, and choose **Attach Debugger**

Detaching Collaboration Debugger

You can detach Collaboration Debugger from a collaboration object in either System Manager or Integrated Test Environment, as described in the following sections:

- “Detaching Collaboration Debugger in System Manager”
- “Detaching Collaboration Debugger in Integrated Test Environment”

Detaching Collaboration Debugger in System Manager

Do the following to detach Collaboration Debugger from a collaboration object in System Manager:

1. In the InterChange Server Express Component Management view, expand the server instance in which the collaboration object exists, then expand the Collaboration Objects folder.
2. Select the collaboration objects that you want to detach Collaboration Debugger from. You can use standard Windows selection techniques to select multiple user projects for deployment at the same time, such as the following:
   - Hold down **Shift** to select contiguous items.
   - Hold down **Ctrl** to select non-contiguous items.
3. Right-click one of the selected collaboration objects and choose **Detach Debugger** from the context menu

Detaching Collaboration Debugger in Integrated Test Environment

To detach Collaboration Debugger from a collaboration object in Integrated Test Environment, right-click the center icon for the collaboration object in the Test Unit view and choose **Detach Debugger** from the context menu.

For more information about the Test Unit view, see “Using the Test Unit view” on page 229.

Detaching Collaboration Debugger in Collaboration Debugger

Do one of the following to detach Collaboration Debugger from a collaboration object in Collaboration Debugger itself:

- Right-click anywhere in the “Collaboration template” editor and choose **Detach Debugger** from the context menu
- Right-click the collaboration object in the “Breakpoints” view or the “Events” view and choose **Detach Debugger** from the context menu
- Select the collaboration object in the “Breakpoints” view or the “Events” view and select **Collaboration Debugger > Detach Debugger** from the menu bar
- Select the collaboration object in the “Breakpoints” view and click **Detach Debugger** in the toolbar of the “Breakpoints” view
- Select the collaboration object in the “Breakpoints” view, click the drop-down arrow in the upper-right-hand corner of the title bar, and choose **Detach Debugger**

Removing detached collaboration objects from Collaboration Debugger

Do one of the following to remove all collaboration objects you have detached from Collaboration Debugger so that they are not displayed in the views:

- Right-click anywhere in the “Events” or “Breakpoints” views and choose **Remove All Detached** from the context menu
- Click **Remove All Detached** in the toolbar of the “Breakpoints” view
- Click the drop-down arrow in the upper-right-hand corner of the title bar of the “Breakpoints” view, and choose **Remove All Detached**

Using the collaboration template editor

The “Collaboration template” editor is a read-only editor that displays the diagrams of the collaboration template upon which the collaboration object you are debugging is based. It allows you to view the design of the business process and perform many of the tasks associated with debugging a collaboration.

The “Collaboration template” editor allows you to work with diagrams as described in “Opening and closing diagrams” and node labels and IDs as described in “Showing and hiding node IDs and labels.”

In addition to these sections, which describe tasks that are specific to the “Collaboration template” editor, the following sections describe tasks you can perform using multiple interfaces:

- “Using the collaboration template editor”
- “Working with breakpoints” on page 271
- “Performing debugging operations” on page 274

Opening and closing diagrams

To manage breakpoints for nodes that belong to subdiagram and iterator nodes in a collaboration template, you must open the diagram for the node.

Do one of the following to open the diagram for a subdiagram or iterator node:

- Select the node and press **Enter**
- Double-click the node
- Right-click the node and choose **Open Diagram** from the context menu.

To close the diagram for a subdiagram or iterator node, click the **X** in the upper-right-hand corner of the tab for the diagram.

Showing and hiding node IDs and labels

The unique IDs and labels assigned to the nodes and links in a collaboration template diagram make it easy to identify them and to understand the business process defined by the template. Having them displayed in the Collaboration Debugger can therefore facilitate your troubleshooting experience.

To show the unique identifiers in a diagram, right-click anywhere in the editor and select **Show uniqueIDs** from the context menu.

To hide the unique identifiers in a diagram, right-click anywhere in the editor and select **Hide uniqueIDs** from the context menu.

To show the labels in a diagram, right-click anywhere in the editor and select **Show labels** from the context menu.

To show the unique identifiers in a diagram, right-click anywhere in the editor and select **Show labels** from the context menu.

Configuring Collaboration template editor preferences

Do the following to configure your preferences for the Collaboration template editor:

1. Select **Window > Preferences** from the menu bar of the workbench.
2. Expand **Collaboration Debugger**.
3. Select **Diagram Colors**.

   Figure 86 shows the Collaboration template diagram color preferences.

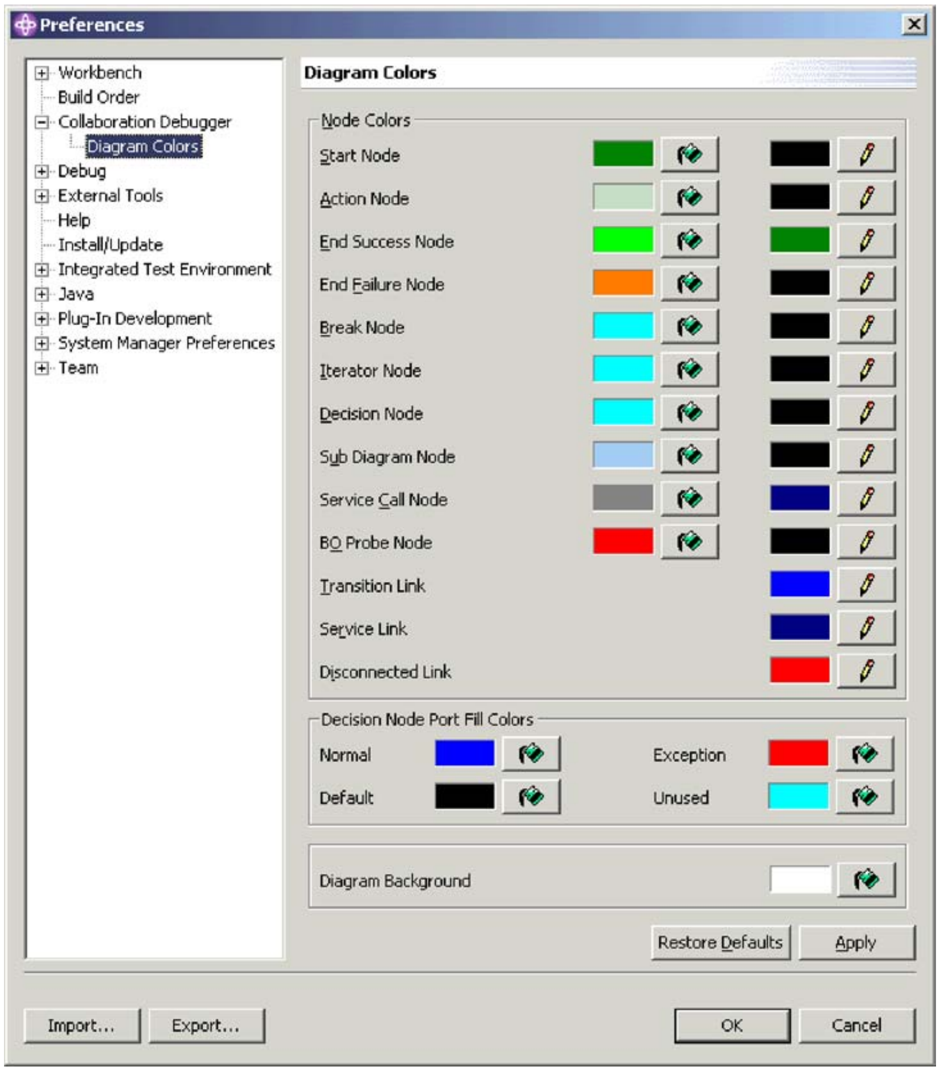


Diagram Colors

Node Colors

Start Node	🟩	⬛
Action Node	🟩	⬛
End Success Node	🟩	🟩
End Failure Node	🟧	⬛
Break Node	🟦	⬛
Iterator Node	🟦	⬛
Decision Node	🟦	⬛
Sub Diagram Node	🟦	⬛
Service Call Node	⬜	🟦
BO Probe Node	🟥	⬛
Transition Link		🟦
Service Link		🟦
Disconnected Link		🟥

Decision Node Port Fill Colors

Normal	🟦	Exception	🟥
Default	⬛	Unused	🟦

Diagram Background	⬜

Figure 86. Collaboration template diagram color preferences

4. Click the paint bucket icons associated with the different interface elements to set the desired fill color for those elements.

Click the pencil icons associated with the different interface elements to set the desired line color for those elements.

At the "Color" dialog, select the color you would like the Test Unit view to use for the background of that component type, then click **OK**.

5. Click **OK**.

Working with events

Events are the flows that trigger the execution of a collaboration object. Collaborations execute business process logic to manage event data, manipulating it and coordinating it among enterprise software resources. You use Collaboration Debugger primarily to pause the processing of an event so that you can inspect its data at significant points in the business logic.

Displaying an event

When a new flow is delivered for processing to a collaboration object that you are debugging it is assigned a number and is listed beneath the collaboration object in the "Events" view.

Figure 87 shows an event that has been received by a collaboration object and is displayed in the "Events" view.

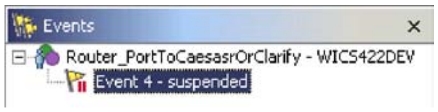


Figure 87. An event displayed in the Events view

When you display an event, Collaboration Debugger gives focus to the node where processing is paused in the "Collaboration template" editor; black squares appear in each corner of the node, indicating that it is selected, and a flag icon labeled with the event instance is displayed in the upper-left-hand corner of the node. Figure 88 shows a node where processing of the event is paused in the Collaboration template editor.

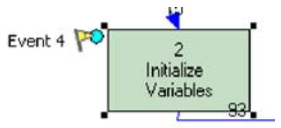


Figure 88. An event displayed in the Collaboration template editor

If there are active breakpoints set in the collaboration, the flow suspends processing while it waits for you to perform a debugging operation, and a flag icon is drawn on the node where processing is broken to represent the event as well.

You display an event to locate it in the collaboration diagram and to populate the "Variables" view with the current data in the flow.

The "Variables" view dynamically updates as you perform debugging operations, so you typically only need to display an event in the following situations:

- When the collaboration object first receives the flow for processing

- If you have navigated elsewhere in Collaboration Debugger (to display a breakpoint, for example) and need to locate the event again

Do one of the following to display an event:
- Right-click the event in the “Events” view and choose **Display** from the context menu
- Select the event in the “Events” view and select **Collaboration Debugger > Display** from the menu bar

Suspending an event

When you run an event, as described in “Running an event” on page 276, the flow of execution proceeds as it would normally, without pausing on any breakpoints. If you decide that you want to resume debugging the flow, you can suspend the event.

Once you have instructed Collaboration Debugger to run an event, it typically processes too quickly for you to suspend it, though service calls provide this opportunity. When a business object is sent out a service call, the business process blocks until a response is received; this provides enough time to suspend the flow prior to replying to the request, so that debugging resumes when the response is received by the collaboration.

Do one of the following to suspend an event:
- Right-click the event in the “Events” view and choose **Suspend** from the context menu
- Select the event in the “Events” view and select **Collaboration Debugger > Suspend** from the menu bar

Running an event

For more information on running an event, see “Running an event” on page 276.

Working with breakpoints

Breakpoints are instructions that Collaboration Debugger uses to pause the execution of a collaboration object. You set breakpoints at node in the business process where you want to inspect the data of a flow to observe its transformation and troubleshoot problems.

You can set breakpoints on the following types of nodes:
- Action node
- Subdiagram node
- Iterator node
- End success node
- End failure node
- Iterator node
- Break node

You can not set a breakpoint on a decision node.

For more information about the different types of nodes in a collaboration template, see the *Collaboration Development Guide*.

The following sections describe how you can arrange breakpoints in a collaboration you are debugging:

- "Setting breakpoints"
- "Displaying breakpoints"
- "Removing breakpoints"
- "Disabling breakpoints" on page 273
- "Enabling breakpoints" on page 273

After you have arranged breakpoints in the collaboration object, use the instructions in "Performing debugging operations" on page 274 to manage the processing of the flow.

Setting breakpoints

Do one of the following to set a breakpoint on a supported node:

- Right-click the node and choose **Set breakpoint(s)** from the context menu
- Select the node and click **Set breakpoint(s)** in the toolbar of Collaboration Debugger
- Select the node and select **Collaboration Debugger > Set breakpoint(s)** from the menu bar
- Double-click a node that does not have a breakpoint set on it already

Displaying breakpoints

Do one of the following to select and give focus to a node on which a breakpoint has been set:

- Right-click the breakpoint in the "Breakpoints" view and choose **Display** from the context menu
- Double-click the breakpoint in the "Breakpoints" view
- Select the breakpoint in the "Breakpoints" view and press **Enter**
- Select the breakpoint in the "Breakpoints" view and click **Display** in the toolbar of the "Breakpoints" view
- Select the breakpoint in the "Breakpoints" view, click the drop-down arrow in the upper-right-hand corner of the title bar, and choose **Display**
- Select the breakpoint in the "Breakpoints" view and click **Display** in the toolbar of Collaboration Debugger
- Select the breakpoint in the "Breakpoints" view and select **Collaboration Debugger > Display breakpoint(s)** from the menu bar

Removing breakpoints

Do one of the following to remove a breakpoint:

- Right-click the node on which the breakpoint is set in the "Collaboration template" editor and choose **Remove breakpoint(s)** from the context menu
- Right-click the breakpoint in the "Breakpoints" view and choose **Remove** from the context menu
- Select the breakpoint in the "Breakpoints" view and click **Remove** in the toolbar of the "Breakpoints" view
- Select the breakpoint in the "Breakpoints" view, click the drop-down arrow in the upper-right-hand corner of the title bar, and choose **Remove**
- Select the node on which the breakpoint is set in the "Collaboration template" editor and click **Remove breakpoint(s)** in the toolbar of Collaboration Debugger

- Select the node on which the breakpoint is set in the “Collaboration template” editor and select **Collaboration Debugger > Remove breakpoint(s)** from the menu bar
- Double-click a node that has a breakpoint set on it

Disabling breakpoints

Do one of the following to disable a breakpoint so that execution no longer pauses at the specified node, but so that you can re-activate it if necessary:

- Right-click the node on which the breakpoint is set in the “Collaboration template” editor and choose **Disable breakpoint(s)** from the context menu
- Right-click the breakpoint in the “Breakpoints” view and choose **Disable** from the context menu
- Select the breakpoint in the “Breakpoints” view and click **Disable** in the toolbar of the “Breakpoints” view
- Select the breakpoint in the “Breakpoints” view, click the drop-down arrow in the upper-right-hand corner of the title bar, and choose **Disable**
- Select the node on which the breakpoint is set in the “Collaboration template” editor and click **Disable breakpoint(s)** in the toolbar of Collaboration Debugger
- Select the node on which the breakpoint is set in the “Collaboration template” editor and select **Collaboration Debugger > Disable breakpoint(s)** from the menu bar

Enabling breakpoints

Do one of the following to enable a breakpoint that has been disabled:

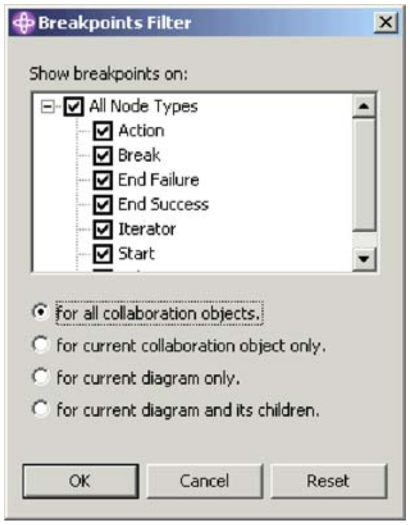
- Right-click the node on which the breakpoint is set in the “Collaboration template” editor and choose **Enable breakpoint(s)** from the context menu
- Right-click the breakpoint in the “Breakpoints” view and choose **Enable** from the context menu
- Select the breakpoint in the “Breakpoints” view and click **Enable** in the toolbar of the “Breakpoints” view
- Select the breakpoint in the “Breakpoints” view, click the drop-down arrow in the upper-right-hand corner of the title bar, and choose **Enable**
- Select the node on which the breakpoint is set in the “Collaboration template” editor and click **Enable breakpoint(s)** in the toolbar of Collaboration Debugger
- Select the node on which the breakpoint is set in the “Collaboration template” editor and select **Collaboration Debugger > Enable breakpoint(s)** from the menu bar

Filtering breakpoints

You can filter the “Breakpoints” view to only display breakpoints set on certain types of nodes. Do the following to filter the “Breakpoints” view:

1. Do one of the following to display the “Breakpoints Filter” dialog:
   - Click **Filter** in the title bar of the “Breakpoints” view
   - Click the down-facing arrow in the upper-right-hand corner of the title bar of the “Breakpoints” view and select **Filter** from the drop-down menu

   Collaboration Debugger displays the “Breakpoints Filter” dialog, as shown in Figure 90 on page 277:


Breakpoints Filter

Show breakpoints on:

- ✓ All Node Types
  - ✓ Action
  - ✓ Break
  - ✓ End Failure
  - ✓ End Success
  - ✓ Iterator
  - ✓ Start

● for all collaboration objects.
○ for current collaboration object only.
○ for current diagram only.
○ for current diagram and its children.

Figure 89. Breakpoints filter

2. Use the following techniques to configure your filter settings:
   - Enable the **All Node Types** checkbox to display all supported node types
   - Enable only the checkboxes for specific node types to display only those types
   - Click **Reset** to apply the default filter settings
3. Enable one of the following radio buttons to specify how broadly the filter should be applied:
   - **for all collaboration objects**
   - **for current collaboration object only**
   - **for current diagram only**
   - **for current diagram and its children**
4. Click **OK**.

Performing debugging operations

Once you have arranged breakpoints in the collaboration object you are debugging, you perform debugging operations to manage the flow of the event through the breakpoints. The following operations are available:

- "Stepping over"
- "Stepping into" on page 275
- "Stepping out" on page 275
- "Running" on page 275
- "Running to a specific node" on page 276
- "Running an event" on page 276

Stepping over

When you perform a "step over" operation, InterChange Server Express resumes the paused flow so that the node on which the breakpoint is set executes, and then pauses the flow at the next node.

Do one of the following to step over a node where execution is broken:

- Right-click anywhere in the “Collaboration template” editor and choose **Step Over** from the context menu
- Give focus to the “Collaboration template” editor and click **Step Over** in the toolbar
- Give focus to the “Collaboration template” editor and select **Collaboration Debugger > Step Over** from the menu bar
- Give focus to the “Collaboration template” editor and use the keyboard shortcut **F6**

Stepping into

When you perform a “step into” operation on a subdiagram or iterator node, InterChange Server Express resumes the paused flow. Execution proceeds until the flow reaches the start node in the subdiagram or iterator and then pauses.

Do one of the following to step into a subdiagram or iterator where execution is broken:
- Right-click anywhere in the “Collaboration template” editor and choose **Step Into** from the context menu
- Give focus to the “Collaboration template” editor and click **Step Into** in the toolbar
- Give focus to the “Collaboration template” editor and select **Collaboration Debugger > Step Into** from the menu bar
- Give focus to the “Collaboration template” editor and use the keyboard shortcut **F5**

Stepping out

When you perform a “step out” operation on a node within a subdiagram or iterator, InterChange Server Express resumes the paused flow. Execution proceeds until the subdiagram or iterator completes, or until the flow reaches an active breakpoint within the subdiagram or iterator.

Do one of the following to step out of a subdiagram or iterator where execution is broken:
- Right-click anywhere in the “Collaboration template” editor and choose **Step Out** from the context menu
- Give focus to the “Collaboration template” editor and click **Step Out** in the toolbar
- Give focus to the “Collaboration template” editor and select **Collaboration Debugger > Step Out** from the menu bar
- Give focus to the “Collaboration template” editor and use the keyboard shortcut **F7**

Running

When you perform a “run” operation, InterChange Server Express resumes the paused flow and allows processing to continue either until completion of the business process or to a node with an active breakpoint set on it.

Do one of the following to run a paused flow:
- Right-click anywhere in the “Collaboration template” editor and choose **Run** from the context menu
- Give focus to the “Collaboration template” editor and click **Run** in the toolbar

- Give focus to the "Collaboration template" editor and select **Collaboration Debugger > Run** from the menu bar
- Give focus to the "Collaboration template" editor and use the keyboard shortcut **F8**

Running to a specific node

When you perform a "run to this node" operation on a flow, InterChange Server Express resumes the paused flow and allows processing to continue either to the node on which you performed the operation, or to a node with an active breakpoint set on it.

Do one of the following to run a paused flow to a specific node:

- Right-click the node to which you want execution to run and choose **Run to this node** from the context menu
- Select the node to which you want execution to run and click **Run to** in the toolbar
- Select the node to which you want execution to run and select **Collaboration Debugger > Run to** from the menu bar

Running an event

When you perform a "run event" operation, InterChange Server Express resumes the paused flow and allows processing to continue, bypassing any breakpoints in the business process even if the breakpoints are active.

Do one of the following to run an event:

- Right-click the event in the "Events" view and choose **Run** from the context menu
- Select the event in the "Events" view and select **Collaboration Debugger > Run Event(s)** from the context menu
- Select the event in the "Events" view and click **Run Event(s)** in the toolbar

Working with variables

The "Variables" view displays the value of any variable that is declared in the "Declarations" tab of the collaboration template definition. This includes the business objects that are automatically declared by Process Designer Express to be associated with each port that you create on the "Ports and Triggering Events" tab, as well as any variables you add to the "Declarations" tab yourself.

When you display an event as described in "Displaying an event" on page 270 or manage a flow as described in "Performing debugging operations" on page 274, the "Variables" view updates the variable information it shows. You can expand any variables of `BusObj` type to view their attributes.

There is currently no way to display the `triggeringBusObj` variable. This is a system-declared variable that references the original flow that triggered execution of the collaboration object, and is typically very significant. If you want to maintain a record of the values of the `triggeringBusObj` variable, you can declare a new `BusObj` variable in the "Declarations" section of the collaboration template, then initialize it to the value of the `triggeringBusObj` variable in the first node of the scenario.

Furthermore, many delivered collaboration templates manipulate a variable named `processingBusObj` and leave the `triggeringBusObj` variable untouched. The `processingBusObj` variable is typically declared at the scenario level rather than the template level, though, so it is not displayed in the “Variables” view. To have the `processingBusObj` variable available in the “Variables” view, move the declaration for the variable from the scenario definition to the template definition.

You can filter the “Variables” view to only display variables of certain types. Do the following to filter the “Variables” view:

1. Do one of the following to display the “Variables Filter” dialog:
   - Click **Filter** in the title bar of the “Variables” view
   - Click the down-facing arrow in the upper-right-hand corner of the title bar of the “Variables” view and select **Filter** from the drop-down menu

   Collaboration Debugger displays the “Variables Filter” dialog, as shown in Figure 90:

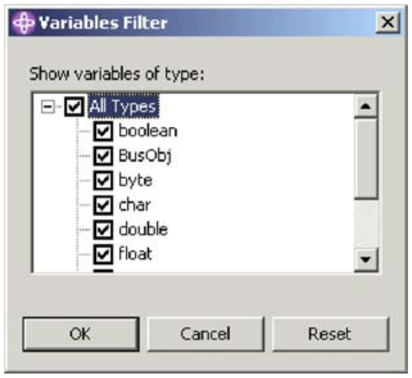


Figure 90. Variables filter

2. Use the following techniques to configure your filter settings:
   - Enable the **All Types** checkbox to display all supported variable types
   - Enable only the checkboxes for specific variable types to display only those types
   - Click **Reset** to apply the default filter settings
3. Click **OK**.

Debugging a collaboration

Use the following workflow as a guideline to efficiently debug a collaboration object:

1. Create the collaboration template, create a collaboration object based on it, deploy both to the InterChange Server Express instance, and make sure that the collaboration object is started.
2. Start the Collaboration Debugger perspective as described in “Starting Collaboration Debugger” on page 259.
3. Attach Collaboration Debugger to the collaboration object as described in “Attaching Collaboration Debugger” on page 266.
4. Set breakpoints in desired locations, as described in “Working with breakpoints” on page 271.

You will usually find it most useful to set breakpoints on nodes where the flow is being transformed, where the flow is being sent out of the collaboration, and where problematic pieces of logic are located.

5. Create and send an event such that the collaboration object will be triggered, using either Test Connector or Integrated Test Environment. For more information on Test Connector, see Chapter 11, "Using Test Connector," on page 189.

   For more information on Integrated Test Environment, see Chapter 12, "Using Integrated Test Environment," on page 201.
6. Display the submitted event as described in "Displaying an event" on page 270.
7. Manage the flow across the breakpoints you have configured by using the techniques described in "Performing debugging operations" on page 274.
8. Use the "Variables" view to watch the data of the event change as it is processed by the collaboration, as described in "Working with variables" on page 276.
9. Repeat steps 4 on page 277 through 8 to modify the breakpoints and process flows with different data.

Chapter 14. Performance tuning

This chapter describes how to implement various techniques and configurations to improve system performance. It contains the following sections:

Implementing concurrent processing of event-triggered flows

You can configure collaboration objects and connector controllers to process multiple event-triggered flows at the same time. This can significantly improve the performance for event-triggered interfaces.

Concurrent event-triggered flow processing in Collaborations

Collaborations can be configured to process multiple event-triggered flows concurrently. System throughput and response time to event processing improve when concurrent event processing is properly used (see tip below). By default, a collaboration processes one event-triggered flow at a time.

When a collaboration is processing concurrent event-triggered flows, the collaboration identifies dependencies among those flows and processes them in the order that they were sent from the connector controller. Concurrent processing of event-triggered flows is performed on flows that do not have data conflicts, while flows with data conflicts are processed in the order received.

To configure a collaboration to handle multiple event-triggered flows, see "Maximum number of concurrent events" on page 167.

Tip: Concurrent processing of event-triggered flows in collaborations requires additional system resources. To maximize performance, ensure that system resources used to handle concurrent events are not idle. For example, do not set the value for the **Maximum number of concurrent events** to 10 if the number of events in the collaboration queue never reaches 10.

If a collaboration's inbound ports are bound only to receive external calls through the Server Access Interface, and are not bound to any connectors, you can improve performance by setting the value of **Maximum number of concurrent events** to zero. But do not set this value to zero if the collaboration is being used for bidirectional exchanges with connectors.

Concurrent event-triggered flow processing in collaboration-object groups

Each collaboration in a collaboration-object group can be configured independently for processing a number of concurrent event-triggered flows. It is recommended that you set the same value for the number of concurrent event-triggered flows for all of the collaborations in a group, so that a collaboration with a low concurrency rate does not become a bottleneck.

Concurrent event-triggered flow processing in connector controllers

Connector controllers can be configured to process multiple event-triggered flows concurrently.

Event-flow processing performance improves when a connector is configured to process triggered event flows concurrently. This is because multiple business objects can be transformed in mapping at the same time.

To configure concurrent event-triggered flow processing for connector controllers, you set the **ConcurrentEventTriggeredFlows** property to the maximum number of flows you want processed at the same time. For more information on this property, see “ConcurrentEventTriggeredFlows” on page 137. For more information on using Connector Configurator to set connector properties, see in Chapter 7, “Configuring connectors,” on page 121 general.

Tip: If the **ConcurrentEventTriggeredFlows** property is configured to a value greater than 1, the connector controller maintains the same order of events that it received from the application. Concurrent processing of event-triggered flows in connectors requires additional system resources. To maximize performance, ensure that the system resources that handle concurrent events are not idle. For example, do not set the **ConcurrentEventTriggeredFlows** property to a value of 10 if the number of events arriving at InterChange Server Express for the connector never reaches 10. Use the server statistics window to determine the number of events that are in connector’s queue by monitoring the MQ queue depth for the connector. Monitoring this statistic can help you set the value for the **ConcurrentEventTriggeredFlows** property.

If a connector is being used only as a destination, you can improve performance by setting the value of **ConcurrentEventTriggeredFlows** to zero. But do not set this value to zero if the connector is being used in bidirectional exchanges with a collaboration.

Distributing connector agents

You can improve the performance of the business integration system by distributing the connector agents. The reasons connector agent distribution improves performance include the following:

- Each InterChange Server Express and connector agent process runs in its own Java Virtual Machine (JVM). A computer can only have so many JVM instances running on it before performance begins to degrade. The threshold is dependent upon the computer specifications, but distributing a connector agent decreases the load on the machine hosting InterChange Server Express by one JVM.
- The performance of an individual connector generally improves the closer it is to the application with which it communicates, with respect to the network topology. Ideally the connector agent should be installed on the same computer that hosts the application server itself, though this might not be possible depending on the environment. If the application host computer and the InterChange Server Express host computer are on different subnets, however, the connector agent will typically still perform better if installed on another computer in the same subnet as the one in which the application host computer exists.

For instructions about setting up distributed connector agents, see the *System Administration Guide*.

Caching static relationships

You can cache static relationships so that the participant data they contain is loaded into memory and any queries for the data are issued against the in-memory data, rather than in the data stored in the database. This can make the relationship lookups much faster than if they had to retrieve the information from the database for each flow.

For more information on configuring static relationships to be cached, see the *Map Development Guide*.

Using database connection pools

It is common to have to retrieve or insert information into a database table during the processing of a flow. Performance is impacted if the interface must establish a connection to the database for each flow, particularly because the log in process must be performed for each connection.

Database connection pools are integration components that establish and maintain a pool of connections to a database resource. The connections are established initially and are available to other components to issue queries against the database resource. When a component is done using a connection, the connection is released and returned to the pool so that another component may use it. This eliminates the need to log in each time a connection is required.

For information on how to create database connection pools, see Chapter 8, "Configuring database connection pools," on page 145. For information on how to use database connection pools in maps, see the *Map Development Guide*. For information on how to use database connection pools in collaboration templates, see the *Collaboration Development Guide*.

Using the memory checker thread

InterChange Server Express features a memory checker thread that you can use to regulate how events are processed by the system and thereby regulate the consumption of system memory. This can mitigate the risk of system crashes due to lack of memory.

The memory checker thread periodically measures the amount of memory used by InterChange Server Express and evaluates if it is within an acceptable configured range. If memory usage is not within an acceptable range then the memory checker thread manages system components to reduce the memory usage.

If the memory usage exceeds a lower threshold specified by the lower threshold then the memory checker thread causes the event listener threads of the connector controllers in the system to sleep for a configurable maximum amount of time. The amount of time varies depending on how much the memory usage exceeds the lower threshold, but does not exceed the amount of time specified by the **Connector pause time at threshold** property. By causing the event listener threads to sleep in this way, the memory checker thread slows down the rate at which events are delivered to InterChange Server Express and reduces the risk of the upper memory threshold being exceeded.

If the memory usage exceeds the upper threshold specified by the **Memory lower threshold percent** property then the memory checker thread pauses the connectors in the system. When connectors are paused they continue to process business objects that are currently queued, but they do not poll new events. The connectors therefore reduce the number of queued business objects and thereby reduce the amount of memory in use, without using more memory to process new business objects. The connectors remain paused for a minimum amount of time specified by **Connector pause time at threshold** property. When the interval specified by the Connector pause time at threshold property passes, the memory checker thread examines memory usage again. If the memory usage does not exceed the upper threshold anymore then the memory checker thread starts the connectors again. If the memory usage still exceeds the upper threshold, however, the memory checker thread allows the connectors to remain paused.

The memory checker thread sleeps in between its examination and state management operations for the amount of time specified by the Memory checker sleep time property.

To use the memory checker thread, you must follow the instructions in the following sections:

1. "Setting the CW_MEMORY_MAX parameter"
2. "Configuring the memory checker thread properties"
3. "Further considerations" on page 284

Setting the CW_MEMORY_MAX parameter

You must configure the Java parameter `CW_MEMORY_MAX` in the InterChange Server Express startup script to set the maximum heap memory that InterChange Server Express can use. The memory checker thread takes action based on the values of configurable properties that specify percentages of this maximum heap memory amount.

You should set the `CW_MEMORY_MAX` parameter to the same value specified for the `-mx` parameter, which specifies the maximum heap size for InterChange Server Express. By default the `CW_MEMORY_MAX` parameter is set to same value as the `-mx` parameter, but if you modify the `-mx` parameter then you must modify the `CW_MEMORY_MAX` parameter accordingly.

The following example is from an unmodified `start_server.bat` startup script for InterChange Server Express on a Windows computer:

```
REM This is the  -mx param value for the InterChange Server Express's memory heap
set CW_MEM_HEAP=512

REM Start the InterChange Server Express
%CWJAVA% -Djava.ext.dirs=%JRE_EXT_DIRS%;"%MQ_LIB%";"%DB2_LIB%"
-Duser.home="%CROSSWORLDS%" -mx%CW_MEM_HEAP%m -DTEAgent=1200
-DCW_MEMORY_MAX=%CW_MEM_HEAP% %ORB_PROPERTY% -classpath %JCLASSES%
ServerWrapper -s%SERVERNAME% %2 %3
```

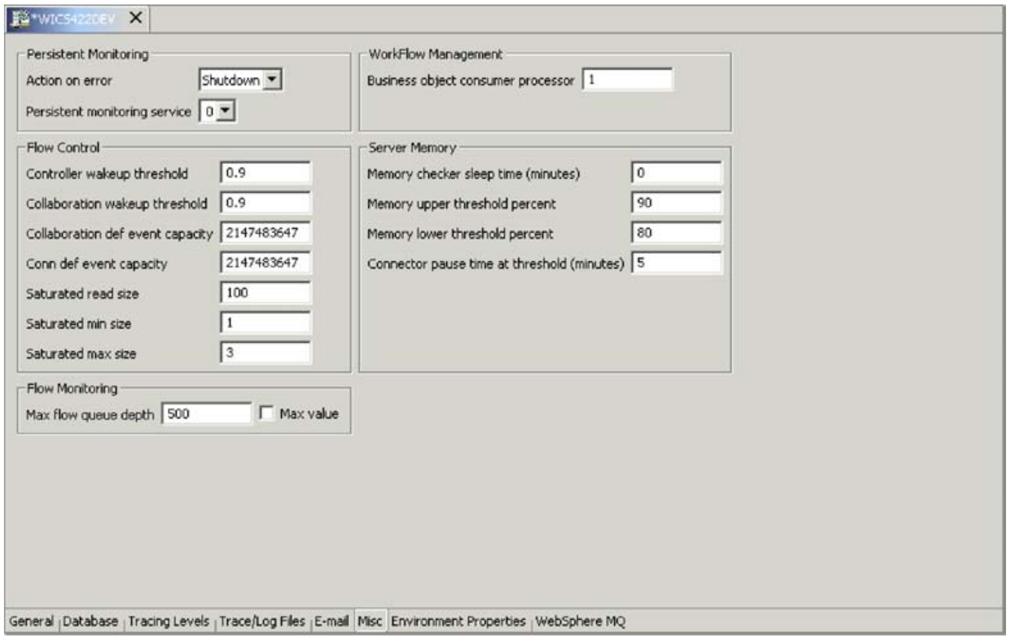
The `CW_MEM_HEAP` variable is set to the value 512, and is used to set the `-mx` parameter, so 512 megabytes of memory are reserved for the Java heap. The `CW_MEMORY_MAX` parameter is set to the same value by also using the `CW_MEM_HEAP` variable. Note that the `CW_MEMORY_MAX` parameter must be proceeded by `-D`.

Configuring the memory checker thread properties

You must edit the InterChange Server Express configuration file to set the properties for the memory checker thread. These properties are exposed on the "Misc" tab of the InterChange Server Express configuration interface. For more information on configuring InterChange Server Express, see "Configuring miscellaneous properties using System Manager" on page 106. Do the following in the "Server Memory" pane of the "Misc" tab to configure the memory checker thread:

1. Set the **Memory checker sleep time** field to the interval of time for which the memory checker thread should sleep in between each operation. Set the **Memory checker sleep time** field to the value `0` (the default value) to disable the memory checker thread.
2. Set the **Memory upper threshold percent** field to the percentage of the total memory available to InterChange Server Express (specified by the `CW_MEMORY_MAX` parameter) which, when exceeded, should cause the memory checker thread to pause the connectors so that the connectors do not poll any new events until the memory usage drops beneath the upper threshold again. The default value is `90`, meaning that the upper threshold is 90 percent of the total memory specified by the `CW_MEMORY_MAX` parameter. For example, if the `CW_MEMORY_MAX` parameter is set to 512 megabytes, then the upper threshold would be approximately 460 megabytes.
3. Set the **Memory lower threshold percent** field to the percentage of the total memory available to InterChange Server Express (specified by the `CW_MEMORY_MAX` parameter) which, when exceeded, should cause the memory checker thread to reduce the speed at which connectors deliver events to InterChange Server Express. The default value is `80`, meaning that the lower threshold is 80 percent of the total memory specified by the `CW_MEMORY_MAX` parameter. For example, if the `CW_MEMORY_MAX` parameter is set to 512 megabytes, then the lower threshold would be approximately 410 megabytes.
4. Set the **Connector pause time at threshold** field to an amount of time (in minutes) that both specifies the maximum time for which the rate at which connectors deliver events to InterChange Server Express is reduced when the lower memory threshold is exceeded, and the minimum of time for which the connectors will remain paused when the upper memory threshold is exceeded. The default value is `5`.

Figure 91 on page 284 shows the "Misc" tab and the "Server Memory" properties used to configure the memory checker thread.


*WICS422DEV

Persistent Monitoring
- Action on error: Shutdown
- Persistent monitoring service: 0

WorkFlow Management
- Business object consumer processor: 1

Flow Control
- Controller wakeup threshold: 0.9
- Collaboration wakeup threshold: 0.9
- Collaboration def event capacity: 2147483647
- Conn def event capacity: 2147483647
- Saturated read size: 100
- Saturated min size: 1
- Saturated max size: 3

Server Memory
- Memory checker sleep time (minutes): 0
- Memory upper threshold percent: 90
- Memory lower threshold percent: 80
- Connector pause time at threshold (minutes): 5

Flow Monitoring
- Max flow queue depth: 500
- Max value: unchecked

Figure 91. Configuring memory checker thread properties

Further considerations

Consider the following information when using the memory checker thread:

- You should not use the memory checker thread if you are running an InterChange Server Express instance that processes few transactions. For instance, if you have the `-mx` parameter set to less than 512 megabytes, the Java Virtual Machine garbage collection thread will not be able to free memory effectively.
- Use the following techniques to determine the best configuration for the memory checker thread properties:
  - Initially, set the **Memory lower threshold percent** and **Memory upper threshold percent** properties close to one another, test the behavior of the system, then increase the difference between the values and test the system again. Repeat this process to determine the optimal difference.
  - Set the **Connector pause time at threshold** property between one and five minutes.
  - If InterChange Server Express must handle a heavy workload, set the Memory checker sleep time property to a small number so that it takes more frequent measurements and regulatory actions.

Notices

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

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

Printed in USA