WebSphere Business Integration Server
Express and Express Plus

IBM

Adapter for e-Mail User Guide

Version 4.3.1

WebSphere Business Integration Server
Express and Express Plus

IBM

Adapter for e-Mail User Guide

Version 4.3.1

> **Note!**
> Before using this information and the product it supports, read the information in "Notices" on page 67.

30July2004

This edition of this document applies to IBM WebSphere Business Integration Server Express, version 4.3.1, IBM Websphere Business Integration Server Express Plus, version 4.3.1, and to all subsequent releases and modifications until otherwise indicated in new editions.

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

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

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 Toolset Express 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.

This document describes installation, configuration, business object development, and troubleshooting for the Adapter for e_Mail.

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.

Audience

This document is for IBM consultants and customers. Users of this document should be familiar with the WebSphere business integration systems and e-mail technologies.

IBM(R) WebSphere(R) Business Integration Server Express Adapters supply integration connectivity for leading e-business technologies and enterprise applications.

This document describes the installation, configuration, and business object development for the adapter for e-Mail.

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, http://www.ibm.com/software/integration/websphere/support/. Select the component area of interest and browse the Technotes and Flashes sections.

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 cross-reference or variable name.
blue text	Blue text, which is visible only when you view the manual online, indicates a cross-reference hyperlink. Click any blue text 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 for Windows. OS/400 and Linux use forward slashes (/) for directory paths. All WebSphere Business Integration Server Express system product pathnames are relative to the directory where the product is installed on your system.
`%text%` and `$text`	Text within percent (%) signs indicates the value of the Windows `text` system variable or user variable.
ProductDir	Represents the directory where the IBM WebSphere Business Integration Server Express for Adapters product is installed. The defaults for each platform are as follows: Windows: IBM\WebSphereServer OS/400: /QIBM/ProdData/WBIServer43/product Linux: /home/${username}/IBM/WebSphereServer

New in this release

New in release 4.3.1

The Adapter for e-Mail is supported on the following operating systems:

- Microsoft Windows 2000
- Microsoft Windows 2003
- IBM OS/400 V5R2, V5R3
- Red Hat Enterprise Linux AS 3.0
- SuSE Linux Enterprise Server 8.1 with SP3

New in Release 4.3

This is the first release of this guide.

Chapter 1. Overview of the adapter

This chapter describes the e-Mail adapter and its components. The e-Mail adapter enables the integration broker, InterChange Server Express, to exchange business objects, files, or messages with e-mail applications. The adapter also enables InterChange Server Express to send e-mail to specified users when Error or Fatal Error messages occur. It contains the following sections:

- "Features of the e-Mail adapter"
- "Adapter components"
- "How the e-Mail adapter works" on page 3
- "Example scenario for the e-Mail adapter" on page 8
- "Processing locale-dependent data" on page 10

Features of the e-Mail adapter

The e-Mail adapter has the following functionality:

- **Business object processing** – The e-Mail adapter enables InterChange Server Express to send e-mail messages containing business objects or files to e-mail applications.

  The adapter can also poll for new e-mail messages on one or more mail servers, and convert the messages into business objects that it sends to InterChange Server Express. Encapsulating business objects in e-mail messages enables enterprises to integrate business processes and synchronize data across the Internet.

- **Error notification** – The e-Mail adapter enables IBM WebSphere Business Integration Server Express to use e-mail to deliver error messages. InterChange Server Express sends an e-mail message to specified users when Error or Fatal Error messages occur.

  The e-Mail adapter works with the e-Mail collaboration, which is automatically created by InterChange Server Express. When an error occurs and e-mail notification is configured, InterChange Server Express creates an e-Mail Notification business object containing the message information and passes it to the e-Mail adapter. The adapter extracts the message information from the business object, creates an e-mail message, and sends the message.

  Note: Business object processing and error notification are two distinct roles for the adapter. In order to enable both behaviors, you must configure the adapter for both. For more information, see Chapter 2, "Installing and configuring the adapter," on page 11.

Adapter components

Adapters consist of two parts: the adapter framework and the application-specific component. The adapter framework, whose code is common to all adapters, acts as an intermediary between InterChange Server Express and the application-specific component. The application-specific component contains code tailored to a particular application or technology. The adapter framework provides the following services between InterChange Server Express and the application-specific component:

- Receives and sends business objects

- Manages the exchange of startup and administrative messages

This document contains information about the adapter framework and the application-specific component, which it refers to as the adapter. The WebSphere Business Integration Server Express adapter for e-Mail also uses a data handler, which converts a business object to a specific data format, and converts e-mail attachments in a specific format into business objects. The e-Mail adapter can use one of the IBM-delivered data handlers or a custom data handler. For information on data handlers, see Chapter 3, "Planning an e-Mail adapter implementation," on page 23 and the *Data Handler Guide*.

Figure 1 illustrates the architecture of the e-Mail adapter. The integration broker is InterChange Server Express.
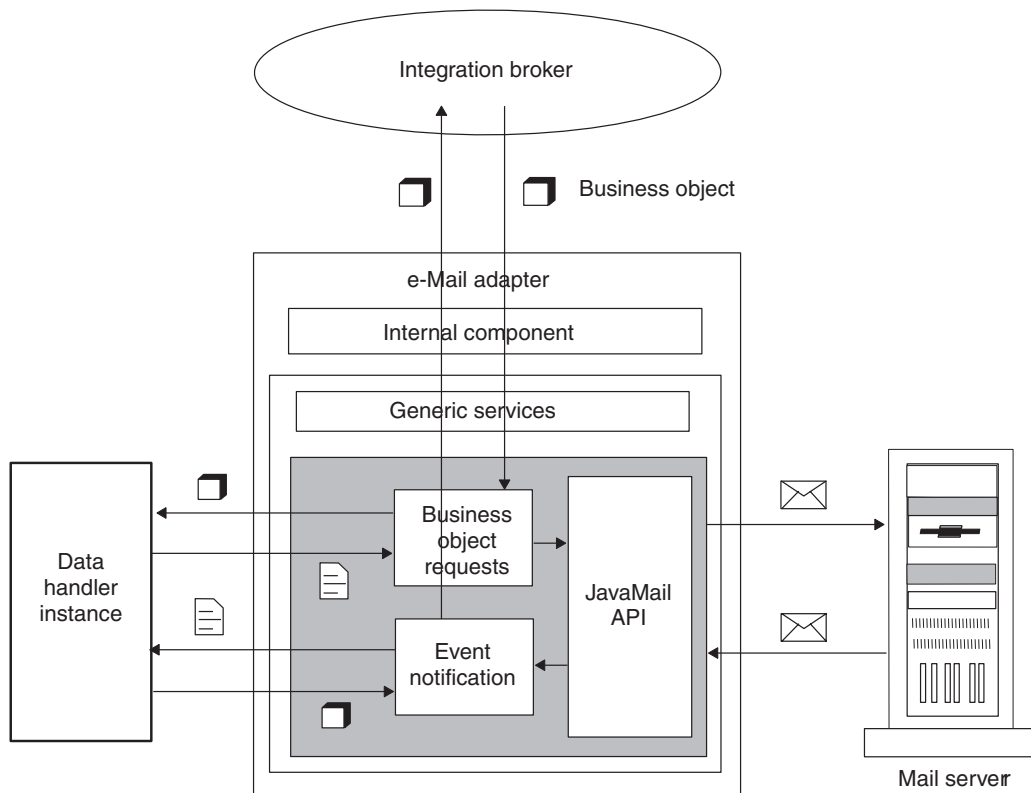
Figure 1. e-Mail adapter architecture

Transport protocols

The adapter interacts with mail servers using JavaMail. The adapter supplies the parameters that JavaMail requires to communicate with mail servers, and JavaMail handles the underlying SMTP (Simple Mail Transport Protocol) and IMAP (Internet Message Access Protocol) protocols.

SMTP is the Internet mail transport protocol. The adapter uses the SMTP protocol for the mail transport.

For polling, the e-Mail adapter supports the IMAP protocol via JavaMail for online access of e-mail messages and remote manipulation of the mailbox. IMAP is used for storing and manipulating messages on a mail server, thus enabling access of e-mail from anywhere. IMAP supports online query of messages, selective retrieval

of message contents, and server-side searches for messages. Note that the e-Mail adapter does not currently support the POP protocol via JavaMail.

How the e-Mail adapter works

The following sections describe how the e-Mail adapter processes business object requests and handles events.

Business object processing

When InterChange Server Express sends a business object to the e-Mail adapter, the adapter processes the business object and generates an e-mail.

An e-Mail business object must contain e-mail routing information (from address, to address, subject, and content), the mime type of the attachments, and the business objects to deliver. The business object can also specify the complete path of files to include in the e-mail as attachments. A single business object can result in an e-mail message containing multiple business objects and multiple files as attachments.

To process a business object request, the adapter:

1. Extracts the routing information from the top-level business object and composes an e-mail message.
2. Processes each contained business object or file as follows:
   - If there are contained business objects, the adapter calls a data handler to convert each business object into a business object string that is included as an attachment to the e-mail. The mime type of the attachment is defined in the top-level business object. There is only one business object per attachment.
   - If there are files specified in the top-level business object, the adapter includes the files as attachments to the e-mail. Each file must be located at its specified location.
3. When the e-mail is complete, the adapter delivers it using JavaMail.

Figure 2 illustrates business object processing with the e-Mail adapter. For information on creating business objects for the e-Mail adapter, see Chapter 4, "Developing business objects for the adapter," on page 25. The integration broker is InterChange Server Express.
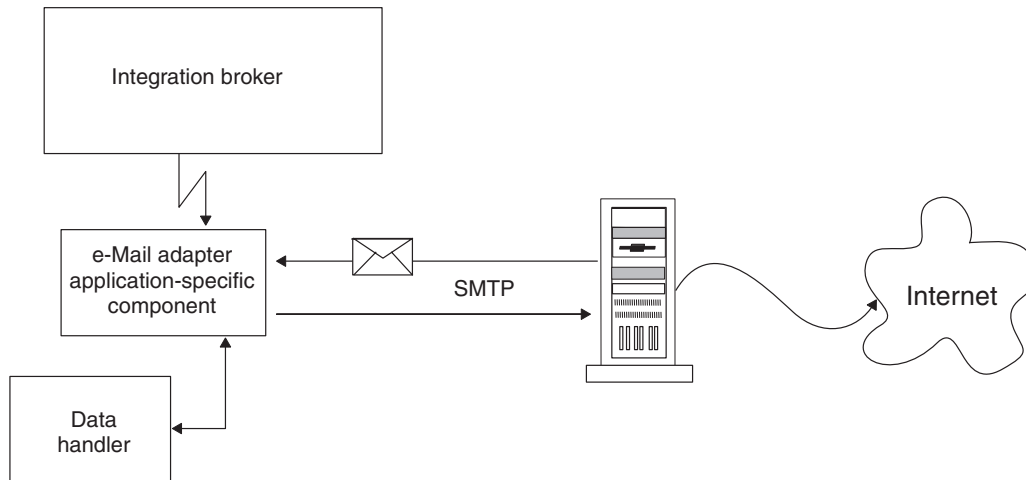
Figure 2. e-Mail adapter business object processing

Event notification

In event notification, the adapter polls for e-mail messages in specified folders on the mail server. The adapter checks the folders, retrieves new messages, converts the messages to business objects using the data handler, and sends the business objects to InterChange Server Express.

The adapter can poll in multiple locations. The following folders are required per poll location:

- Poll folder – The folder in which the adapter polls.
- Archive folder – The folder in which the adapter archives successful events.
- Failed event folder – The folder in which the adapter archives failed events.

Although polling locations can be located on different IMAP stores, these three folders, if specified for a poll location, should be in the same mailbox.

The sections that follow describe the event mechanism in more detail.

Event triggering

A triggering event for the e-Mail adapter is a new e-mail that arrives in a poll folder. The adapter checks each poll folder for new e-mails at the interval defined by the PollFrequency configuration property.

Event detection

Each IMAP mail server has a message store called an IMAP store. The IMAP store has a mailbox for each of the users of the mail server. A mailbox can have multiple folders, and a folder can contain multiple e-mail messages or folders.

Each poll folder must have a corresponding archive folder and failed event folder. The poll folder, archive folder, and failed event folder are together referred to as a **poll location**. An administrator must create the folders and specify the poll locations in the meta-object for the adapter.

The e-Mail adapter polls for new messages in the poll folder of an IMAP store. When the adapter sees a new message in a poll folder, it retrieves the message. The adapter determines that a message is a new message by checking the status of the IMAP flag for the message.

Event status: IMAP supports flags that describe the state of e-mail messages in a mailbox. The event detection mechanism uses the flags to determine which e-mail messages to retrieve. The status of an event is specified both by the folder in which the message is located and by the message flag.

The event mechanism uses the IMAP flag SEEN to mark events that are "in progress." In polling, the adapter checks the messages in a poll folder and picks messages that are not marked with the SEEN flag. These events are "ready to be polled." When the adapter picks a message from the poll folder for processing, it sets the SEEN flag for the message. Events marked as SEEN are "in progress" events.

Note: The adapter does not poll messages marked as SEEN.

When processing is complete, and all the attachments of a message are successfully converted to business objects and sent to InterChange Server Express, the e-mail is marked as FLAGGED and then moved to the archive folder. Messages that have attachments that cannot be converted to business objects, or messages that have unsubscribed business objects or business objects that could not be delivered to InterChange Server Express are moved to the failed event folder. Table 1 summarizes the event states for the e-Mail adapter.

Table 1. Event states

Event state	e-Mail message state
ready to be polled	All messages in the poll folder with the SEEN flag not set.
in progress	All messages in the poll folder with the SEEN flag set.
sent to integration broker	All messages in the archive folder. Any message in the PollFolder that is marked with the FLAGGED flag set. **Note:** The adapter changes the flag of an IN_PROGRESS message to FLAGGED once it has been sent to InterChange Server Express and is set to be archived (moved to the archive folder).
unsubscribed business objects and errors	All messages in the failed event folder.

Event retrieval

To retrieve events, the adapter polls each poll folder in turn and picks one e-mail from each mailbox. This polling strategy ensures that e-mails in all poll folders get processed.

The adapter processes each of the e-mails retrieved from a poll location in the following way:

1. From the poll folder, the adapter picks an e-mail with the SEEN flag not set. The adapter then sets the SEEN flag on this e-mail.
2. The adapter fetches the e-mail from the poll folder. The e-mail can have any number of attachments, and the adapter processes each attachment. There must be only one business object per attachment.

3. For each attachment, the adapter uses the mime type of the attachment to determine which data handler to use to convert the attachment to a business object. The adapter retrieves the contents of the attachment and passes it to the data handler. Attachments in an e-mail can be in different formats. If the attachment has no mime type, however, the adapter does not process the attachment.

4. The data handler generates a business object and returns it to the adapter.

5. If the business object is subscribed, the adapter delivers it to InterChange Server Express; otherwise, the adapter considers this e-mail to be a failed event.

The adapter processes all message attachments in the same way. An e-mail message is considered to be a successful event if all the attachments were converted to business objects and delivered to InterChange Server Express. In the cases of successful e-mails, the adapter marks the e-mail messages as FLAGGED, and then archives the successfully processed e-mails to the archive folder for this polling location. The adapter does not archive an e-mail until all of the attachments of the e-mail have been processed.

If one or more attachments cannot be converted to business objects or the business object provided by the data handler is unsubscribed, the event is a failed event. In this case, the adapter archives the e-mail in the failed event folder for this polling location.

The adapter delivers a number of events per folder that is less than or equal to the value configured for the PollQuantity adapter property. The adapter does not retrieve and process new e-mail until it has finished processing the previous e-mail.

Failed events: An e-mail is considered to be a failed event in the following situations:

- An attachment could not be retrieved.
- The adapter was not able to determine which data handler to use for an attachment.
- The data handler was not able to convert an attachment to a business object.
- The business object returned by a data handler was not subscribed to.

The adapter processes all the attachments of a message and delivers business objects obtained from a message to InterChange Server Express, even if the message is a failed event. As an example, assume that an e-mail has five attachments. The adapter processes attachments one and two, fails to process attachment three, and then succeeds in processing attachments four and five. Although the adapter delivers business objects for four of the attachments, the entire e-mail message is treated as failed event and archived in the failed event folder because attachment three could not be processed. The adapter logs the details of each attachment that it fails to convert to a business object.

Event archiving

To archive events, the adapter requires an archive folder and a failed event folder. Events are archived into these folders as follows:

- If an e-mail is a successful event, the adapter marks the e-mail message as FLAGGED, then, moves the e-mail from the poll folder to the archive folder. If an archive folder is not specified, the adapter deletes the e-mail from the poll folder, and the message is lost.

- If the e-mail is a failed event, the adapter moves the e-mail from the poll folder to the failed event folder. If a failed event folder is not specified, the adapter deletes the event from the poll folder, and the message is lost.

Recovery mechanism

An administrator can configure the recovery mechanism of the adapter for each polling location specified by the PollConfigMO configuration property. The PollConfigMO property identifies a meta object that has an `InDoubtEvents` attribute for each polling location. This attribute can have the values `Reprocess`, `Ignore`, `LogError`, or `FailOnStartUp`.

If the adapter crashes between poll calls, the poll folders may have messages that are in-progress. These messages have a SEEN flag set and are in-doubt events for the adapter. The next time the adapter starts, the initialization procedure scans all the poll folders for the in-doubt transactions as follows:

- If `InDoubtEvents` for the poll location is set to `Reprocess`, the adapter resets the SEEN flag of the in-doubt events in that poll folder. The adapter picks these messages in the subsequent poll calls.
- If the InDoubtEvents setting for the poll location is FailOnStartUp, the adapter terminates after displaying error messages.
- If the InDoubtEvents setting for the poll location is LogError, the adapter continues after displaying error messages.
- If `InDoubtEvents` is Ignore, the adapter continues without processing the in-doubt transactions.

Before restarting the adapter, the administrator can view each of the poll folders using an e-mail client program, and determine how to handle in-doubt events. The administrator can either leave the in-doubt messages in the poll folder, reset the SEEN flag of in-doubt events, or delete the messages. When the adapter next starts, if there are in-doubt events, the adapter processes the in-doubt transactions in the initialization method as explained below.

Adapter initialization

When the e-Mail adapter starts, the following occurs:

1. The adapter gets the subscription list for the business objects it supports.
2. Using the information in the adapter's meta-object, the adapter creates a list of poll locations.
3. The adapter tries to connect to each poll location. The adapter does not start if it fails to connect to any of the poll locations.
4. The adapter makes sure that all the folders for each poll location exist. The adapter will not start if any folder does not exist.
5. The adapter retrieves the value of the SMTP_MAILHOST property from the adapter properties. Using this value, the adapter connects to the mail host. If the attempt to connect to the mail host is not successful, the adapter does not start.
6. The adapter looks for in-doubt transactions in each of the poll folders. If the adapter finds in-doubt transactions, it performs recovery based on the value of the `InDoubtEvents` setting for the poll location. If the `InDoubtEvents` flag is `Reprocess`, the adapter resets the SEEN flag for the message. If it is `FailOnStartUp`, the adapter terminates. If it is `LogError`, the adapter logs the error and continues. If the `InDoubtEvents` flag is `Ignore`, the adapter continues and ignores the in-doubt transactions.

Example scenario for the e-Mail adapter

Figure 3 illustrates a hypothetical scenario for the e-Mail adapter. The illustration shows a cross-enterprise solution that enables trading partners to exchange business data via e-mail. The trading partners have chosen to exchange business data via e-mail rather than communicate across firewalls or send data via FTP (File Transfer Protocol).

Trading partner A has an enterprise application generating business data. A WebSphere Business Integration Server Express adapter polls for events in the application, generates business objects, and sends the business objects to InterChange Server Express. InterChange Server Express is configured to forward the business objects to the e-Mail adapter.

The e-Mail adapter uses a data handler to convert business objects to a specified data format. The adapter composes an e-mail message, attaches files containing the converted data, and sends the message across the Internet to mail host MailServer1.

Concurrently, a legacy application generates e-mail messages containing business data formatted as text strings, and sends the messages to mail host MailServer2.

Trading partner B configures the e-Mail adapter to poll both MailServer1 and MailServer2 for new e-mails. When new e-mails arrive, the adapter checks the mime type of each attachment, calls the appropriate data handler to convert the attachments to business objects, and sends the business objects to InterChange Server Express, the integration broker.

Figure 3. Example scenario for the e-Mail adapter

As Figure 3 shows, trading partners can create new mailboxes at host mail servers and use the e-Mail adapter to execute business processes over the Internet. For enterprises with firewalls between the Internet and local networks, or between internal department networks, the use of e-mail enables data exchange without requiring tunnels through firewalls. The e-Mail adapter is also useful when applications are locked for direct access but can exchange data by means of e-mail.

Processing locale-dependent data

The adapter has been internationalized so that it can support double-byte character sets, and deliver message text in the specified language. When the adapter transfers data from a location that uses one character code set to a location that uses a different code set, it performs character conversion to preserve the meaning of the data.

The Java runtime environment within the Java Virtual Machine (JVM) represents data in the Unicode character code set. Unicode contains encodings for characters in most known character code sets (both single-byte and multibyte). Most components in the WebSphere business integration system are written in Java. Therefore, when data is transferred between most WebSphere business integration system components, there is no need for character conversion.

To log error and informational messages in the appropriate language and for the appropriate country or territory, configure the Locale standard configuration property for your environment. For more information on these properties, see "Appendix A. Standard configuration properties" on page 35

Characteristics of e-Mail adapter locale processing

The e-Mail adapter encodes the e-mails it exchanges according to character set and encoding information configured at the header, content, and business object levels. The following traits characterize the encoding activity:

- In the header, only the subject line is encoded; fields that contain information about the sender and receiver, for instance, are not encoded.
- Business objects contained in an e-mail can either be processed as parts of the message or as attachment files; for more information on this, see "Business object structures" on page 25.

Locale configuration order of precedence

There is an order of precedence to how the adapter attempts to apply encoding to the data it processes:

- To encode a business object that is contained in an e-mail, the adapter first checks if the mime type specified for the business object is "text"; if it is not, then the adapter uses "UTF-8" as the mime character set. Note that this does not apply to the message header or content, but only to a business object.
- The adapter searches for encoding information in the top-level wrapper business object for data it processes, which contains information about the content and routing requirements of an e-mail; if the necessary attributes have been defined and configured then the adapter uses these specifications. For more information about the structure of top-level objects, see "Business object structures" on page 25.

  If encoding information is not specified at the business object level, then the adapter checks if the optional properties are configured in the adapter definition; if they are then the adapter uses those properties to drive its behavior. For more information about adapter-specific properties, see "Adapter-specific properties" on page 16.
- If encoding information is not specified at either the business object level or in the adapter definition, then the adapter uses the default system locale information.

Chapter 2. Installing and configuring the adapter

This chapter describes how to install and configure the e-Mail adapter. It contains the following sections:

- "Determining installation and configuration tasks"
- "Installing the e-Mail adapter and related files" on page 13
- "Adapter configuration tasks" on page 15
- Chapter 2, "Installing and configuring the adapter"
- "Starting the connector" on page 20

Compatibility

The adapter framework used must be compatible with the version of the integration broker with which the adapter is communicating. The 4.3.1 version of the adapter for e-Mail is supported on the following adapter framework and integration brokers:

Adapter framework: Adapter Framework, version 4.3.1

Integration brokers: InterChange Server Express, version 4.3.1

See the Release Notes for any exceptions.

Prerequisites

The connector runs on:

- Microsoft Windows 2000
- Microsoft Windows 2003
- IBM OS/400 V5R2, V5R3
- Red Hat Enterprise Linux AS 3.0
- SuSE Linux Enterprise Server 8.1 with SP3

Determining installation and configuration tasks

The installation and configuration tasks for the e-Mail adapter depend on the roles it must fill in the WebSphere business integration system. Read the sections below to determine what tasks you must perform based on how you will be using the e-Mail adapter.

Using the adapter for event notification

If the adapter must poll for new e-Mail messages in an application and send them to InterChange Server Express for processing, perform the following steps:

1. Install the adapter as described in "Installing the e-Mail adapter and related files" on page 13.
2. Configure the mail server and mailboxes for poll locations. For information, see "Configuring mail servers for polling" on page 15.
3. Create the business objects required to support the interface. For more information, see Chapter 4, "Developing business objects for the adapter," on page 25.

4. Create or modify a meta-object for the data handler that the adapter uses to process the data. For more information, see "Data handler meta-objects" on page 29.

5. Create and configure a poll meta-object to contain a child meta-object for each poll location. For more information, see "Using e-Mail adapter meta-objects" on page 30.

6. Configure the mail servers that the adapter will be polling as described in "Configuring mail servers for polling" on page 15.

7. Configure the adapter definition as described in "Configuring the adapter definition" on page 15. Keep the following points in mind:
   - You must add the business objects developed for the interface in 3 as supported business objects for the adapter.
   - You must add the poll meta-object created in 5 as a supported object for the adapter.
   - You must add the data handler meta-object created in 4 as a supported object for the adapter.
   - You must configure the properties of the adapter; because the adapter will be polling, you must be sure to set the value of the PollConfigMO property to the name of the poll meta-object created in 5.

8. Run the adapter as described in "Starting the connector" on page 20.

Using the adapter for business object processing

If the adapter must receive business objects from InterChange Server Express and process them, perform the following steps:

1. Install the adapter as described in "Installing the e-Mail adapter and related files" on page 13.

2. Create the business objects required to support the interface. For more information, see Chapter 4, "Developing business objects for the adapter," on page 25. These business objects will include a top-level wrapper business object that contains routing information for the adapter. For more information, see "Using e-Mail adapter business objects" on page 25.

3. Create or modify a meta-object for the data handler that the adapter uses to process the data. For more information, see "Data handler meta-objects" on page 29.

4. Configure the adapter definition as described in "Configuring the adapter definition" on page 15. Keep the following points in mind:
   - You must add the business objects developed for the interface in 3 as supported business objects for the adapter.
   - You must add the data handler meta-object created in 4 as a supported object for the adapter.
   - You must configure the properties of the adapter.

5. Run the adapter as described in "Starting the connector" on page 20.

Using the adapter for error notification

Your integration broker is InterChange Server Express. If the adapter will be used for e-mail notification of errors, perform the following steps:

1. The e-Mail adapter is automatically installed for InterChange Server Express because it plays a significant role in error notification. You do not have to perform installation tasks. Proceed to configure the adapter properties as described in "Configuring the adapter definition" on page 15.

2. Configure the Email Notification business object as described in "EmailNotification business object" on page 32.

3. You must configure other InterChange Server Express components for e-mail notification. For information on setting up e-mail notification of errors, refer to the *System Administration Guide*.

4. Run the adapter as described in "Starting the connector" on page 20.

Installing the e-Mail adapter and related files

For information on installing the adapter, refer to the WebSphere Business Integration Server Express installation guide for Windows, for OS/400, or for Linux located in the WebSphere Business Integration Server Express Infocenter at the following site:

http://www.ibm.com/websphere/wbiserverexpress/infocenter

Installed file structure

The following subsections describe the installed file structure of the adapter on Windows, OS/400, and Linux platforms.

Table 2 describes the Windows file structures, Table 3 on page 14 describes the OS/400 file structures, and Table 4 on page 14 describes the Linux file structures used by the connector.

For more information on installing the connector component, refer to the following guides:
• *Quick Start Guide*
• *Installation Guide for IBM WebSphere Business Integration Server Express and Express Plus*

Installed file structure for Windows

Table 2. Installed Windows file structure for the connector

Subdirectory of %ProductDir%	Description
connectors\EMail	Contains the start up script start_Email.bat and start_Email_service.bat. It also contains the CWEMail.jar file.
connectors\EMail\JavaMail-Sun	Contains the activation.jar and mail.jar files.
connectors\messages	Contains the EMailConnector.txt file as well as the EMailConnector_ll_TT.txt files (message files specific to a language (ll) and a country or territory (TT)).
repository	Contains the EMail.jar and CN_EMail.txt files.
connectors\EMail\Samples	Has a directory WebSphereICS (InterChange Server Express) that contains EmailSample.jar and PurchaseOrder.bo files.
\lib	Contains the WBIA. jar file.
\bin	Contains the CWConnEnv.bat file.

For Windows, the installer adds an icon for the connector file viewable by selecting Programs > IBM WebSphere Business Integration Express > Adapters > Connectors. For a fast way to start the connector, create a shortcut to this file on the desktop.

Installed file structure for OS/400

Table 3. Installed OS/400 file structure for the connector

Subdirectory of %ProductDir%	Description
connectors/EMail	Contains the connector CWEMail.jar and the start_EMail.sh files.
connectors\EMail\JavaMail-Sun	Contains the activation.jar and mail.jar files.
connectors/messages	Contains the EMailConnector.txt file as well as the EMailConnector_ll_TT.txt files (message files specific to a language (ll) and a country or territory (TT)).
repository	Contains the EMail.jar and CN_EMail.txt files.
connectors\EMail\Samples	Has a directory WebSphereICS (InterChange Server Express) that contains EmailSample.jar and PurchaseOrder.bo files.
/lib	Contains the WBIA. jar file.
/bin	Contains the CWConnEnv.sh file.

For OS/400, a fast way to start the connector is to use the Console function. For information about the Console, refer to the online help provided with the Console.

Installed file structure for Linux

Table 4. Installed Linux file structure for the connector

Subdirectory of %ProductDir%	Description
connectors/EMail	Contains the connector CWEMail.jar and the start_EMail.sh.files.
connectors/EMail/JavaMail-Sun	Contains the activation.jar and mail.jar files. .
connectors/messages	Contains the EMailConnector.txt file as well as the EMailConnector_ll_TT.txt files (message files specific to a language (ll) and a country or territory (TT)).
repository	Contains the EMail.jar and CN_EMail.txt files.
connectors/EMail/Samples/WebSphereICS	Contains the EmailSample.jar and PurchaseOrder.bo files.
/lib	Contains the WBIA. jar file.
/bin	Contains the CWConnEnv.sh file.

For Linux, you have to start the connector using the "connector_manager" command.

Adapter configuration tasks

There are many configuration tasks for the e-Mail adapter, though only some of them are required, depending on how the adapter is being used. Refer to "Determining installation and configuration tasks" on page 11 to determine which of the following configuration tasks are required for your situation, and then refer to the sections below for specific task information.

This section describes:
- "Configuring mail servers for polling" on page 15
- "Configuring the adapter definition"
- "Standard configuration properties" on page 16
- "Adapter-specific properties" on page 16

Configuring mail servers for polling

If the e-Mail adapter must poll for e-mail messages to send to InterChange Server Express, you must configure the mail servers to provide the mailboxes and poll locations that the e-Mail adapter uses for polling. To configure mail servers, do the following:

1. Create user accounts that can access the poll locations. Users can be located on a single mail server or on multiple mail servers.

2. Configure the poll locations for each user by creating a poll folder, archive folder, and failed event folder for each poll location. Although users may have multiple poll locations, and poll locations can be located on different mail servers, the three folders for a poll location must be located in the same mailbox. The adapter connects to these folders using the user name and password of the mailbox owner.

Note that the adapter does not use these mailboxes when sending business object requests. Instead, it uses the mail server name specified by the SMTP_MailHost adapter configuration properties.

Configuring the adapter definition

You must configure the e-Mail adapter before running it. To configure the adapter:
- Set the standard and adapter-specific configuration properties.
- If the adapter must poll for events, then you must configure the required meta-objects for the e-Mail adapter. For information on configuring meta-objects, see "Using e-Mail adapter meta-objects" on page 30.

After configuring the meta-objects, you must add support for them to the adapter definition. To configure an adapter definition, use Connector Configurator Express (which you launch from System Manager for InterChange Server Express).

An adapter obtains its configuration values at startup. During a runtime session, you may want to change the values of one or more adapter properties. Changes to some adapter configuration properties may take effect immediately or may require a restart of the adapter or of the whole integration system, depending on the property and the integration system. To determine when changes to the adapter configuration take effect, refer to the Update Method column on the relevant properties tab in the adapter configuration tool.

Standard configuration properties

Standard configuration properties provide information that all adapters use. See "Appendix A. Standard configuration properties" on page 35 for documentation of these properties.

Table 5 provides information specific to this adapter about configuration properties in the appendix.

Table 5. Property information specific to this adapter

Property	Note
CharacterEncoding	This adapter does not use this property.
ApplicationName	You must provide a value for this configuration property before running the adapter.
Locale	Because this adapter has been internationalized, you can change the value of this property. See release notes for the adapter to determine currently supported locales.

Adapter-specific properties

Adapter-specific configuration properties provide information for the adapter that is specific to the application or technology that the adapter integrates. Adapter-specific properties enable you to change adapter behavior without having to recode and rebuild the application-specific component.

Table 6 lists the adapter-specific configuration properties for the e-Mail adapter. See the sections that follow for explanations of the properties.

Table 6. Adapter-specific configuration properties

Name	Possible values	Default value	Required
ApplicationPassword			No
ApplicationUserName			No
BusinessObjectMimeCharset	a valid character set value		No
BusinessObjectMimeEncoding	a valid data encoding value		No
DataHandlerConfigMO	*Data handler meta-object name*	MO_DataHandler_Default	Yes
DebugMode	true or false	false	No
DefaultVerb	*a verb value that is valid for the business objects handled by the adapter*	Create	No
MailsPerMailBox	*Any positive integer*	1	No
MessageContentMimeCharset	a valid character set value		No
MessageContentMimeEncoding	a valid data encoding value		No
MessageHeaderMimeCharset	a valid character set value		No
MessageHeaderMimeEncoding	a valid data encoding value		No
PollConfigMO	*Meta-object name*		No
PollQuantity	*Any positive integer*	1	No
SMTP_MailHost	*Mail host name*		Yes

ApplicationPassword

Not used.

ApplicationUserName

Not used.

BusinessObjectMimeCharset

The character set for business objects contained in the e-mail. Reference the e-mail RFC documents to determine the proper value for the environment. An example is iso-8859-1, which is the Latin 1 character set of Western European languages.

BusinessObjectMimeEncoding

The encoding for business objects contained in the e-mail. Reference the e-mail RFC documents to determine the proper value for the environment. As an example, you specify Q for "Quoted-Printable" and B for "Base64." If this property is set to a value then business objects are sent as message parts; if it is not specified then business objects are sent as attached files.

DataHandlerConfigMO

Name of the meta-object that the Data Handler uses to set configuration properties. Also used by the DataHandler base class to determine which DataHandler class to use for a particular content type. For information on this meta-object, see "Data handler meta-objects" on page 29.

DebugMode

Specifies whether JavaMail debug messages are sent to STDOUT.

DefaultVerb

Specifies the value that is inserted in the Verb attribute for top-level business objects created by the adapter during event notification if the Verb attribute does not contain a value. In some situations the adapter may poll an event and create a business object, but the original data source (such as an XML document) might not have information in it that directly corresponds to the Verb attribute of the business object definition. If business objects are delivered to InterChange Server Express without a valid verb then they are regarded as not being subscribed to, and do not get processed. You can specify a valid verb (such as `Create`) for this property to populate the Verb attribute of the top-level business objects handled by the adapter in the case that the verb is blank or null.

The default value is `Create`.

MailsPerMailBox

Specifies the number of messages processed in each mailbox before the adapter proceeds to process the next mailbox.

MessageContentMimeCharset

The character set for the message content. Reference the e-mail RFC documents to determine the proper value for the environment. An example is iso-8859-1, which is the Latin 1 character set of Western European languages.

MessageContentMimeEncoding

The encoding for the message content. Reference the e-mail RFC documents to determine the proper value for the environment. As an example, you specify Q for "Quoted-Printable" and B for "Base64".

MessageHeaderMimeCharset

The character set for the message header. Reference the e-mail RFC documents to determine the proper value for the environment. An example is iso-8859-1, which is the Latin 1 character set of Western European languages.

MessageHeaderMimeEncoding
The encoding for the message header. Reference the e-mail RFC documents to determine the proper value for the environment. As an example, you specify Q for "Quoted-Printable" and B for "Base64".

PollConfigMO
The name of the meta-object that the e-Mail adapter uses for polling. For information on the e-Mail adapter meta-object, see "Using e-Mail adapter meta-objects" on page 30.

PollQuantity
Specifies the maximum number of events polled during a single poll call.

SMTP_MailHost
The mail host to be contacted to send e-mail.

Creating multiple connector instances

Note: When you create an additional instance of this adapter (or any adapter that is supplied with WebSphere Business Integration Server Express or Express Plus), that instance of the adapter will be counted as a separate adapter by the licensing function that limits the total number of adapters that can be deployed.

You can set your system up to create and run multiple instances of a connector by following the steps below. You must:
- Create a new directory for the connector instance
- Make sure you have the requisite business object definitions
- Create a new connector definition file
- Create a new start-up script

Create a new directory
You must create a connector directory for each connector instance. This connector directory should be named:
- For Windows platforms,

  ProductDir\connectors\connectorInstance

  If the connector has any connector-specific meta-objects, you must create a meta-object for the connector instance. If you save the meta-object as a file, create this directory and store the file here:

  ProductDir\repository\connectorInstance

  You can specify the ICS servername as a parameter of startup.bat; an example is start_Email.bat connName serverName.
- For OS/400 platforms,

  /QIBM/UserData/WBIServer43/WebSphereICSName/connectors/connectorInstance

  where connectorInstance uniquely identifies the connector instance and where WebSphereICSName is the name of the InterChange Server Express instance with which the connector runs

  If the connector has any connector-specific meta-objects, you must create a meta-object for the connector instance. If you save the meta-object as a file, create this directory and store the file here:
  /QIBM/UserData/WBIServer43/WebSphereICSName/repository/connectorInstance

where WebSphereICSName is the name of the InterChange Server Express instance with which the connector runs

- For Linux platforms,

    ProductDir/connectors/connectorInstance

    where connectorInstance uniquely identifies the connector instance If the connector has any connector-specific meta-objects, you must create a meta-object for the connector instance. If you save the meta-object as a file, create this directory and store the file here: *ProductDir/repository/connectorInstance*

    You can specify the ICS servername as a parameter of connector_manager; an example is connector_manager -start connName WebSphereICSName [-cConfigFile].

Create business object definitions

If the business object definitions for each connector instance do not already exist within the project, you must create them.

1. If you need to modify business object definitions that are associated with the initial connector, copy the appropriate files and use Business Object Designer Express to import them. You can copy any of the files for the initial connector. Just rename them if you make changes to them.

2. Files for the initial connector should reside in the appropriate directory:
   - Windows:

     \ProductDir\repository\initialConnectorInstance

     Any additional files you create should be in the appropriate connector instance subdirectory of \ProductDir\repository

   - OS/400: QIBM/UserData/WBIServer43/WebSphereICSName/repository /initialConnectorInstance where WebSphereICSName is the name of the InterChange Server Express server instance with which the connector runs

     Any additional files you create should be in the appropriate connectorInstance subdirectory of: /QIBM/UserData/WBIServer43/WebSphereICSName/repository

   - Linux:

     /ProductDir/repository/initialConnectorInstance

     Any additional files you create should be in the appropriate connectorInstance subdirectory of /ProductDir/repository

Create a connector definition

You create a configuration file (connector definition) for the connector instance in Connector Configurator Express. To do so:

1. Copy the initial connector's configuration file (connector definition) and rename it.

2. Make sure each connector instance correctly lists its supported business objects (and any associated meta-objects).

3. Customize any connector properties as appropriate.

Create a start-up script

To create a startup script:

1. Copy the initial connector's startup script and name it to include the name of the connector directory:

   dirname

(For Linux only.)You must change the start-up script CONJAR from CONJAR=${CONDIR}/CW${CONNAME}.jar to "CONJAR=${CONDIR}/CWEmail.jar"

2. Put this startup script in the connector directory you created in "Create a new directory" on page 18.

3. (For Windows only.) Create a startup script shortcut.

4. (For Windows only.) Copy the initial connector's shortcut text and change the name of the initial connector (in the command line) to match the name of the new connector instance.

5. (For OS/400 only.) Create a job description for the connector using the information below: CRTDUPOBJ OBJ(QWBIEMAILC) FROMLIB(QWBISVR43) OBJTYPE(*JOBD) TOLIB(QWBISVR43) NEWOBJ(newemailname)

   where *newemailname* is a 10-character name that you use for the job description for your new email connector

6. (For OS/400 only.) Add the new connector to the Console. For information about the Console, refer to the online help provided with the Console.

You can now run both instances of the connector on your integration server at the same time.

Starting the connector

A connector must be explicitly started using its **connector start-up script**. The startup script should reside in the connector's runtime directory. For example, for Windows use:

ProductDir\connectors*connName*

where *connName* identifies the connector. The name of the startup script depends on the operating-system platform, as Table 7 shows.

Table 7. Startup scripts for a connector

Operating system	Startup script
Windows	start_*connName*.bat
OS/400	start_connName.sh
Linux	connector_manager -start connName [-cConfigFile] sets environmental variables and starts the start_connName.sh start-up script automatically. You do not have to run the start-up script manually.

For more information on how to start a connector, including the command line start-up options, refer to the *System Administration Guide*.

Invoking the start-up script on Windows

On Windows platforms, you can invoke the connector start-up script in the following ways:

• From System Monitor

  You can load, activate, deactivate, pause, shutdown or delete a connector using this tool.

• From the **Start** menu

  – Select Program>IBM WebSphere Business Integration Express>Adapters>Connectors>your_connector_name

By default, the program name is "IBM WebSphere Business Integration Express." However, it can be customized. Alternatively, you can create a desktop shortcut to your connector.

– On Windows systems, you can configure the connector to start as a Windows service. In this case, the connector starts when the Windows system boots (for an Auto service) or when you start the service through the Windows Services window (for a Manual service).

- From the command line

```
start_connName connName WebSphereICSName [-cconfigFile ]
```

where connName is the name of the connector and WebSphereICSName is the name of the InterChange Server Express instance. By default, the name of the InterChange Server Express instance is WebSphereICS.

Invoking the start-up script on OS/400

On OS/400 platforms, you can invoke the connector start-up script in the following ways:

- From System Monitor

  You can load, activate, deactivate, pause, shutdown or delete a connector using this tool.

- From Windows

  Where the WebSphere Business Integration Server Express Console is installed, select Programs>IBM WebSphere Business Integration Console >Console. Then, specifiy the OS/400 system name or IP address and a user profile and password that has *JOBCTL special authority.Select the connName adapter from the list of adapters and select the Start Adapter button.

- From the OS/400 command line

  In Batch mode: You need to run CL Command QSH and from the QSHELL environment run /QIBM/ProdData/WBIServer43/bin/submit_adapter.sh connName WebSphereICSName pathToConnNameStartScript jobDescriptionName where *connName* is the connector name, *WebSphereICSName* is the InterChange Server Express server name (default is QWBIDFT), *pathToConnNameStartScript* is the full path to the connector start script, *jobDescriptionName* is the name of the job description to use in the QWBISVR43 library.

  In interactive mode: You need to run CL command QSH and from the QSHELL environment, run /QIBM/UserData/WBIServer43/WebSphereICSName/connectors /connName/start_connName.sh connName WebSphereICSName [-cConfigFile] where connName is the name of your connector and WebSphereICSName is the name of the Interchange Server Express instance.

Note: To start with TCP/IP servers, use this command: /QIBM/ProdData/WBIServer43/bin/add_autostart_adapter.sh with the following parameters in this order: connName, WebSphereICSName, pathToConnNameStartScript, and jobDescriptionName as explained on the previous page.

Invoking the start-up script on Linux

On Linux platforms, you can invoke the connector start-up script in the following way:connector_manager -start connName *WebSphereICSName* [-cConfigFile]

where connName is the name of the connector and WebSphereICSName is
WebSphereICS

Stopping the connector

The way to stop a connector depends on the way that the connector was started.

Stopping the connector from Windows

On Windows platforms, you can stop the connector in the following ways:

- From System Monitor

  You can load, activate, deactivate, pause, shutdown or delete a connector using
  this tool.

- Activate the connector window, and type "q" and then press Enter.

- If the connector started as a Windows service, you can stop it through Control
  Panel (Control panel>Administrative
  Tools>Services>CWConnectorWBIEMailAdapter).

Stopping the connector from OS/400

On OS/400 platforms, you can stop the connector in the following ways:

- From System Monitor

  You can load, activate, deactivate, pause, shutdown, or delete a connector using
  this tool.

- From the Console or command line

  If you started the connector using the Console or using the "submit_adapter.sh"
  script in QSHELL, from the OS/400 command entry, use the CL Command
  WRKACTJOB SBS(QWBISVR43) to display the jobs to the Server Express
  product. Scroll the list to find the job with a jobname that matches the job
  description for the connector. For example, for the e-Mail connector the jobname
  is QWBIEMAILC

  Select option 4 on this job, and press F4 to get the prompt for the ENDJOB
  command. Then specify *IMMED for the Option parameter and press enter.

- If you used the start_connName.sh script to start the adapter from QSHELL,
  press F3 to end the connector.

Stopping the connector from Linux

On Linux systems, connectors run in the background so they have no separate
window. Instead, run the following command to stop the connector:

```
connector_manager connName -stop serverName
```

where *connName is the name of the connector and serverName is the
InterChange Server Express instance*

Chapter 3. Planning an e-Mail adapter implementation

The e-Mail adapter provides for the integration of applications within and across enterprise firewalls using e-mail. This chapter provides information on planning an implementation of the e-Mail adapter. It contains the following sections:

- "Overview of e-Mail adapter implementation"
- "How the e-Mail adapter and data handlers work"
- "Sample e-Mail adapter files" on page 24

Overview of e-Mail adapter implementation

The process of implementing the e-Mail adapter includes a variety of tasks. Here is an overview of the tasks that you must complete before using the e-Mail adapter to send and receive business objects:

1. Determine the format of the data that will be sent and received in e-mail attachments.
2. When you understand the data formats, you can determine what data handlers the adapter will need to convert business objects to e-mail messages and messages to business objects. For information on data handlers, see "How the e-Mail adapter and data handlers work" on page 23.
3. If the adapter must handle business object requests, create the business object definitions that the e-Mail adapter will use. For information, see Chapter 4, "Developing business objects for the adapter," on page 25.
4. Configure the e-Mail adapter with the other components of InterChange Server Express to send or receive the desired business objects via e-mail.
5. Because your integration broker is InterChange Server Express, you may need to modify the data handler top-level meta-object. For adapters, the default top-level meta-object is named MO_DataHandler_Default. To modify this meta-object, use Business Object Designer Express. For information on data handler meta-objects, see "Data handler meta-objects" on page 29.
6. If polling must be implemented, determine what mail servers the adapter will connect with, and create the mailboxes for the poll locations. For information, see "Configuring mail servers for polling" on page 15.
7. For polling, configure the e-Mail adapter meta-object. This meta-object defines the poll locations for the e-Mail adapter. For more information, see "Using e-Mail adapter meta-objects" on page 30.
8. Configure the adapter by setting adapter configuration properties. For example, specify the name of the top-level data handler meta-object in the e-Mail adapter configuration attribute DataHandlerConfigMO. For information on adapter configuration properties, see "Configuring the adapter definition" on page 15.

How the e-Mail adapter and data handlers work

The e-Mail adapter is designed to use data handlers to convert business objects to specified formats and convert data in specified formats to business objects. A data handler is not part of the adapter but is a separate module that can be customized or replaced as needed.

You can either use data handlers provided by IBM or write your own data handlers. If the data is in a proprietary format or the IBM-delivered data handlers

do not support your application's data format, you must write a custom data handler. For example, IBM provides a data handler that converts between business objects and XML data. If your application generates HTML documents rather than XML documents, you would need to implement a custom data handler that would convert between HTML data and business objects.

Before you begin to work with e-Mail adapter, take time to analyze the data formats that your implementation will need to support. Then use the information in Table 8 to determine whether you can use the data handlers provided by IBM.

Table 8. WebSphere business integration system-delivered data handlers

IBM-delivered data handlers	Description
XML data handler	Converts business objects to and from XML documents. Supports XML 1.0.
NameValue data handler	Parses text data based on named fields, for example, fields that identify the business object type.
FixedWidth data handler	Parses text data using fixed-length fields. The field lengths are specified by the MaxLength property of each business object attribute.
Delimited data handler	Parses text data based on a specified delimiter that separates the individual fields of a business object's data.

For information on data handlers, see the *Data Handler Guide.* If you determine that you must write a custom data handler, see the *Data Handler Guide* for information on how to do this.

Sample e-Mail adapter files

You may want to load and run the e-Mail adapter sample files for an example of the use of the e-Mail adapter. The sample presents a scenario in which the e-mail is used to integrate applications across a firewall.

The files include business objects, collaborations, and a `readme` file containing instructions on how to configure and run the sample. The sample files are located in `connector\Email\samples`. To run the sample, you need to configure a mail server and a client e-mail program.

Chapter 4. Developing business objects for the adapter

This chapter describes the structure of the business objects required by the e-Mail adapter. It contains the following sections:

- "Using e-Mail adapter business objects"
- "Data handler meta-objects" on page 29
- "Using e-Mail adapter meta-objects" on page 30
- "EmailNotification business object" on page 32
- "Specifying valid e-mail addresses" on page 34

Using e-Mail adapter business objects

Using the e-Mail adapter, InterChange Server Express can e-mail a message and include business objects or files as attachments to the e-mail. The business objects or files are wrapped in a top-level hierarchical business object that contains routing information for the adapter. The business objects to be sent are defined as children of the top-level business object. If the e-Mail adapter will be used for business object request processing, you must define these wrapper business object definitions.

Note that wrapper business objects need only be designed for request processing. No wrapper business objects are needed for polling. In polling, the adapter looks at the MIME type of each of the e-mail message attachments, instantiates the appropriate data handler, and passes the attachment to the data handler. The data handler determines the name of the business object from the attachment, finds the corresponding business object definition in the list of supported business objects for the adapter, and converts the contents of the attachment into a business object.

Business object structures

The e-Mail adapter top-level business object must contain a set of attributes that describes the content and routing information for an e-mail message. The adapter composes the e-mail message using these attributes. These attributes are listed in Table 9.

Table 9. Required attributes in a top-level e-mail business object

Name	Description
RecipientName	Recipient e-mail address or addresses. For more information, see "Specifying valid e-mail addresses" on page 34.
MessageSubject	Description of the e-mail message. This value is set in the business object by InterChange Server Express.
MessageContent	Content of the e-mail message. This value is set in the business object by InterChange Server Express.
FromAddress	Sender's e-mail address. The sender receives any undelivered messages. For more information, see "Specifying valid e-mail addresses" on page 34.

Table 9. Required attributes in a top-level e-mail business object (continued)

Name	Description
MimeType	Mime type of the attachments. If not specified, the adapter does not process child business objects.
AttachmentExtension	Set the Default Value property of this attribute to the extension that should be given to files that represent business objects contained in the e-mail. For instance, if a business object is sent as an attachment in an e-mail and the Default Value property of this attribute is set to the value txt, then the attachment that represents the business object will have an extension of .txt.
BusinessObjectMimeCharset	The character set for business objects contained in the e-mail. For more information about this property, see the description for the identical property at the level of the adapter definition in "BusinessObjectMimeCharset" on page 17.
BusinessObjectMimeEncoding	The encoding for business objects contained in the e-mail. For more information about this property, see the description for the identical property at the level of the adapter definition in "BusinessObjectMimeEncoding" on page 17.
MessageContentMimeCharset	The character set for the message content. For more information about this property, see the description for the identical property at the level of the adapter definition in "MessageContentMimeCharset" on page 17.
MessageContentMimeEncoding	The encoding for the message content. For more information about this property, see the description for the identical property at the level of the adapter definition in "MessageContentMimeEncoding" on page 17.
MessageHeaderMimeCharset	The character set for the message header. For more information about this property, see the description for the identical property at the level of the adapter definition in "MessageHeaderMimeCharset" on page 17.
MessageHeaderMimeEncoding	The encoding for the message header. For more information about this property, see the description for the identical property at the level of the adapter definition in "MessageHeaderMimeEncoding" on page 18.

In addition to the attributes listed in Table 9, the top-level business object can contain the optional attributes listed in Table 10.

Table 10. Optional attributes in a top-Level e-mail business object

Name	Description
PriorityLevel	Valid values include Normal, High, and Low. The adapter uses this value to set the priority of outgoing mail.

Name	Description
attributes that represent hierarchical business objects	The adapter converts the child business objects into attachments in the e-mail message.
attributes that specify the names of files	The adapter includes the files as attachments in the e-mail message.

Figure 4 illustrates the basic structure of a top-level business object named Email_TLO_Customer_Wrapper for the e-Mail adapter. This business object contains a hierarchical Customer business object as a child and includes an attribute pointing to a file. The Customer business object and the file are added to the e-mail as attachments.
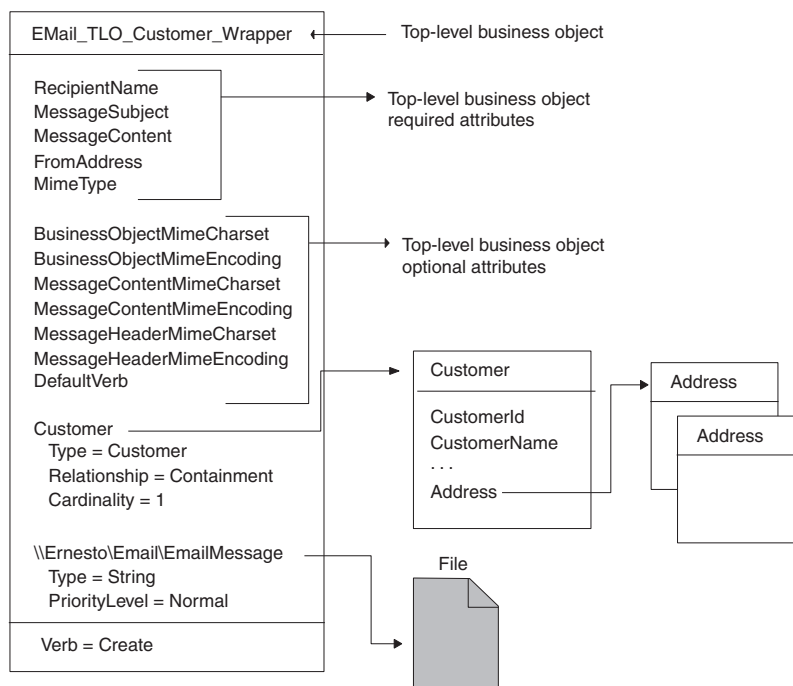
Figure 4. Example e-Mail adapter business object

Attributes specifying business objects

Any business object can be included in the top-level Email business object, as long as the contained business object conforms to the requirements imposed by the configured data handler. If an attribute is a business object, its cardinality must be 1.

When creating business object definitions for the e-Mail adapter, keep in mind that a data handler for the e-Mail adapter must place each business object contained in a top-level business object for the e-Mail adapter in a single attachment. This contained business object may be a large hierarchical business object containing many child business objects of its own, and these children are typically included in the serialized business object in the attachment.

There may be instances when you want to place multiple hierarchical business objects in a single attachment in the top-level e-Mail adapter business object. To do

this, wrap the business objects in a parent business object, and define a complex attribute for that parent in the e-Mail adapter top-level business object.

Attributes specifying files

To include a file attachment with an e-mail message, specify the complete filename including the path name. If the file resides on another machine, specify the file name using the UNC naming convention. For example, if the file resides on a machine named `Ernesto`, specify the path name as:
`\\Ernesto\Email\EmailMessage.txt`. On Windows platforms, you can also map a network drive to the machine and specify the file name as
`F:\Email\EmailMessage.txt`.

How business objects and files are processed

When the adapter has processed the basic set of attributes in the top-level business object, it looks for additional attributes. These can be either type String or type business object. If an attribute is of any other type, the adapter ignores it. The adapter processes each additional non-null attribute as follows:

* If the type of an attribute is string, the adapter treats this as a complete file name. The adapter attaches the file as an attachment to the e-mail message. If the file is not at the location specified in the file name, the adapter returns `BON_FAIL`.

* If the type is a business object, the adapter gets the MimeType attribute of the top-level business object to determine which data handler to use to convert the business object to a string. If the adapter is not able to determine which data handler to use based on the MIME type, or if the MimeType attribute is null, the adapter returns `BON_FAIL`. When a data handler has been instantiated, the adapter passes the business object to the data handler, and the data handler returns a business object string. The adapter puts the business object string in an attachment in the e-mail message. The name of the attachment is the name of the business object, and the MIME type of the attachment is specified by the MimeType attribute of the top-level business object.

After processing all the non-null attributes of the top-level business object successfully, the adapter sends the e-mail.

Business object conformance with data handler requirements

Although you can include any business object in the top-level wrapper business object for the e-Mail adapter, the contained business objects must deliver data in a form that conforms with the requirements of the data handler used to convert the data.

For example, for the BySize data handler, a business object definition must specify a value for the MaxLength attribute property for each business object attribute. For the XML data handler, the business object definition must include application-specific text that enables the data handler to generate an XML document.

A good practice, therefore, is to create your own business object definitions for each type of data to be processed. In the business object definition, provide only the data required by the application and the information required by the data handler. You can then include these business objects in the top-level e-Mail adapter business object.

See the *Data Handler Guide* for information specific to each data handler.

Business object verb processing

When processing business object requests, the e-Mail adapter processes only the Create verb. It returns failure for any other verb. The adapter retains the verb of the child business objects.

For event notification, each e-mail can result in multiple business objects. The application sending the e-mail message is responsible for setting the verb of each business object. The data handler does not process the verbs of these business objects, but sets them in the business objects that it generates.

Business object attribute properties

Business object attributes have properties that can affect how the adapter and integration broker treat those attributes. Table 11 describes how the e-Mail adapter uses these properties for the attributes in the top-level business object.

Table 11. Business object attribute properties

Attribute property	Description
Required	For business object requests, the e-Mail adapter checks whether the Required property is set to True.
Default Values	If the business object does not provide a value for a Required attribute and a default value is specified, the adapter uses the default value.
Max Length	Not used
Type	Not used
Key	Not used
Foreign Key	Not used
Application-Specific Information	Not used

For event notification, the e-Mail adapter does not use business object attribute properties.

Data handler meta-objects

If the e-Mail adapter will be processing business object requests or performing event notification, you must set up a data handler meta-object that specifies what data handlers the adapter must use to convert data.

The top-level meta-object for a data handler is a hierarchical business object that can contain any number of child objects. Each child object is a flat business object that represents a specific data handler instance. Child meta-objects have attributes that provide configuration values that enable a data handler instance to do its work. Different types of data handlers require different configuration properties, so the child meta-objects that support specific handlers have different attributes.

To configure a data handler for the e-Mail adapter, do the following:

- Set up the top-level data handler meta-object to have an attribute for each MIME type that the e-Mail adapter must support. The attribute name should be the name of the MIME type. The attribute represents a child meta-object for a data handler instance. The meta-object typically used for adapters is named MO_DataHandler_Default.

  Note that for attachments received in event notifications, the e-Mail adapter converts the MIME type of the attachment into lowercase before instantiating the

data handler. Therefore, when specifying the MIME type in the data handler meta-object, make sure that the MIME type is in lower case font. Otherwise, the e-Mail adapter cannot instantiate the data handler.

- Set the default attribute values for each child meta-object. The attributes for IBM-delivered data handlers are described in the *Data Handler Guide*.

For detailed information on setting up meta-objects for individual data handlers, see the *Data Handler Guide*.

Note: For the e-Mail adapter to instantiate a data handler, the data handler top-level meta-object must be configured in the list of business objects supported by the adapter.

Using e-Mail adapter meta-objects

Meta-objects are business objects that are designed to contain configuration information for adapters. A meta-object is required to configure the e-Mail adapter for event notification.

The e-Mail adapter gets the name of its top-level meta-object from the adapter property PollConfigMO. You must create the definition of this meta-object to match the structure described in "Structure of the e-Mail adapter meta-object" on page 30. Then specify its name in the PollConfigMO property. For information on creating a business object definition, see the *Business Object Development Guide*.

Important: For the e-Mail adapter to use the configuration information in the meta-object, you must create the meta-object's definition and add it to the list of business objects that the adapter supports.

Structure of the e-Mail adapter meta-object

The meta-object for the e-Mail adapter is a hierarchical business object whose top-level object can contain any number of child objects. Each child object represents one poll location. To configure the adapter to check one or more poll locations, you must set up the meta-object with a corresponding number of child meta-objects. For an illustration of the meta-object definition, see Figure 5 on page 32.

Top-level meta-object attributes
All attributes of the top-level meta-object are container attributes of cardinality 1. For example, if a adapter has two poll locations, Location1 and Location2, the attribute types might be MO_PollLocation1 and MO_PollLocation2.

Child meta-object attributes
Each child meta-object is a flat object with the attributes listed in Table 12.

Table 12. Child meta-object attributes

Name	Description
PollHostName	Name of the host machine running the mail server. The mailbox is located on this server. JavaMail requires this name to connect to the mail store.
UserName	The name of the user. The adapter polls in the mailbox of this user. JavaMail requires this to authenticate the user.

Table 12. Child meta-object attributes (continued)

Name	Description
Password	The password of the user. JavaMail requires this to authenticate the user.
PollFolder	Name of the folder in the mailbox. The adapter polls in this folder. This folder should exist in the mailbox of the user. The default value is PollFolder.
ArchiveFolder	Name of the folder in the mailbox. The adapter archives the successful messages (successful events) in this folder. This folder should exist in the mailbox of the user. If a value is not specified for this attribute, the adapter deletes successful messages after processing them.
FailFolder	Name of the folder in the mailbox. The adapter archives the failed messages (failed events) in this folder. If a value is not specified for this attribute, the adapter deletes failed messages.
InDoubtEvents	Valid values for this attribute are `Reprocess`, `FailOnStartUp`, `LogError`, and `Ignore`. If there are in-doubt events in this poll location, during initialization the adapter processes them based on the default value set for this attribute. If the value is set to `Reprocess`, the adapter resets the SEEN flag of the in-doubt events in the poll folder, and picks up these messages in the subsequent poll calls. If it is set to `FailOnStartUp` and there are in-doubt transactions in this poll folder, the adapter fails. If it is set to `LogError` and there are in-doubt transactions in this poll folder, the adapter logs the error and continues. If it is set to `Ignore` and there are in-doubt transactions in the poll folder, the adapter simply ignores them. The default value is `Reprocess`.
Reconnect	Values for this attribute are True or False. This value is used to determine whether a reconnect attempt should be made for each poll cycle in the event of a connection loss.

Meta-object example

Figure 5 shows an example of a top-level meta-object named `MO_Email_Default`. The e-Mail adapter uses this meta-object to poll two locations, which are specified in the two child meta-objects, `MO_PollLocation1` and `MO_PollLocation2`.

In the example, the value of the EventRecovery attribute in `MO_PollLocation1` is set to `Resubmit`, and the value of this attribute in `MO_PollLocation2` is set to `Ignore`. The value of the EventRecovery attribute informs the adapter administrator to resubmit in-doubt events for Location1 and ignore in-doubt events for Location2.
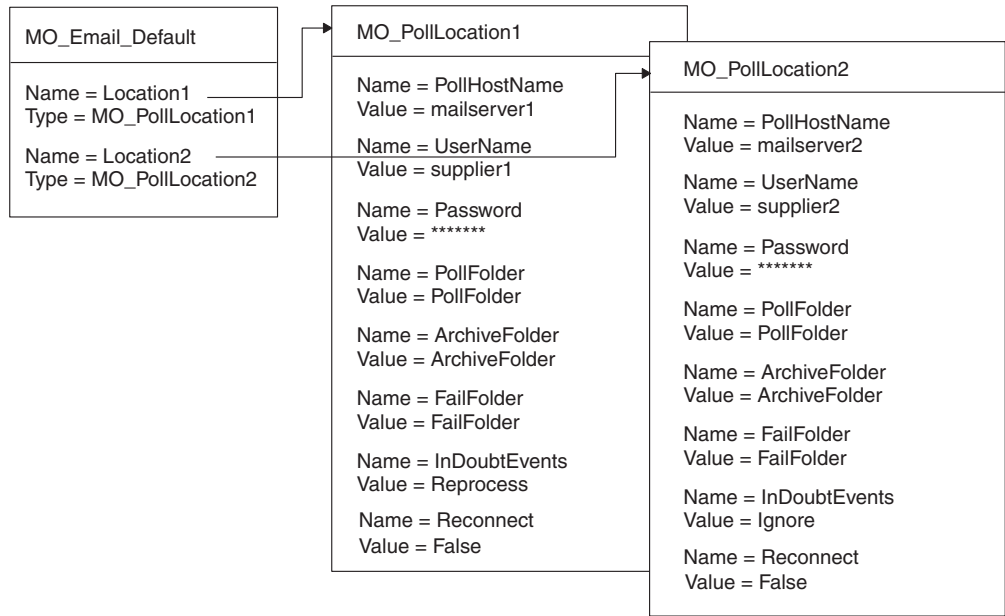
MO_Email_Default

Name = Location1
Type = MO_PollLocation1

Name = Location2
Type = MO_PollLocation2

MO_PollLocation1

Name = PollHostName
Value = mailserver1

Name = UserName
Value = supplier1

Name = Password
Value = *******

Name = PollFolder
Value = PollFolder

Name = ArchiveFolder
Value = ArchiveFolder

Name = FailFolder
Value = FailFolder

Name = InDoubtEvents
Value = Reprocess

Name = Reconnect
Value = False

MO_PollLocation2

Name = PollHostName
Value = mailserver2

Name = UserName
Value = supplier2

Name = Password
Value = *******

Name = PollFolder
Value = PollFolder

Name = ArchiveFolder
Value = ArchiveFolder

Name = FailFolder
Value = FailFolder

Name = InDoubtEvents
Value = Ignore

Name = Reconnect
Value = False

Figure 5. e-Mail adapter meta-object example

EmailNotification business object

Because your integration broker is InterChange Server Express, a business object named EmailNotification is automatically created by the system to support error notification. When an error occurs, the EmailNotification business object is instantiated by the system and sent to the e-Mail adapter. The adapter extracts the message information from the business object, creates the e-mail message, and sends the message.

The EmailNotification business object contains the attributes listed in Table 13. The only supported verb for this business object is Create.

Table 13. EmailNotification business object attributes

Name	Description	Required?
RecipientName	Message recipient. For more information, see "Specifying valid e-mail addresses" on page 34.	Yes
MessageSubject	Description of e-mail content. Value set in business object by collaboration.	Yes. If empty, the adapter uses the default value.
MessageContent	Content of the e-mail message. Value set in business object by collaboration.	Yes. If empty, the adapter uses the default value.
MessageAttachment	File to be attached with the message.	Optional. If empty, no attachment is sent with the message. To include a file attachment with an e-mail message, set this attribute to the complete filename including the path.

Table 13. EmailNotification business object attributes (continued)

Name	Description	Required?
FromAddress	Recipient of undelivered messages. Value defined in business object as part of configuration. For more information, see "Specifying valid e-mail addresses" on page 34.	Optional. If empty, no undelivered messages are returned. To have undelivered messages delivered to an administrator, specify the desired e-mail address in this attribute and enable its Required attribute property.
BusinessObjectMimeCharset	The character set for business objects contained in the e-mail. For more information about this property, see the description for the identical property at the level of the adapter definition in "BusinessObjectMimeCharset" on page 17.	No
BusinessObjectMimeEncoding	The encoding for business objects contained in the e-mail. For more information about this property, see the description for the identical property at the level of the adapter definition in "BusinessObjectMimeEncoding" on page 17.	No
MessageContentMimeCharset	The character set for the message content. For more information about this property, see the description for the identical property at the level of the adapter definition in "MessageContentMimeCharset" on page 17.	No
MessageContentMimeEncoding	The encoding for the message content. For more information about this property, see the description for the identical property at the level of the adapter definition in "MessageContentMimeEncoding" on page 17.	No
MessageHeaderMimeCharset	The character set for the message header. For more information about this property, see the description for the identical property at the level of the adapter definition in "MessageHeaderMimeCharset" on page 17.	No
MessageHeaderMimeEncoding	The encoding for the message header. For more information about this property, see the description for the identical property at the level of the adapter definition in "MessageHeaderMimeEncoding" on page 18.	No

Note: No e-mail is sent if an attachment filename is incorrect, an attachment file does not exist, or an attachment is not readable. If the file resides on another machine, specify the file name using the UNC naming convention. For example, if the file resides on a machine named Ernesto, the path name should be specified as: `\\Ernesto\Email\EmailMessage`. On Windows platforms, you can also map a network drive to the machine and specify the file name as `F:\Email\EmailMessage`.

At runtime, the adapter extracts the attribute values from the business object and inserts the values into an e-mail message as follows:

- Inserts one or more of the message recipients as defined in the RecipientName attribute into the TO field.
- Inserts the value of the MessageSubject attribute into the SUBJECT field.
- Inserts the value of the MessageContent attribute into the CONTENT field.

- Inserts the value of the MessageAttachment attribute as an attachment to the mail.
- Inserts the value of the FromAddress attribute into the FROM field.

Specifying valid e-mail addresses

A valid e-mail address entry can be one or more fully qualified Internet addresses separated by commas. For example, a valid entry for two recipients is:

```
JohnDoe@company.com,FredSmith@company.com
```

The adapter does not resolve personal address aliases, such as the EngineeringDepartment alias defined in a personal address book. However, a valid address can be an alias defined in a mail server, such as EngineeringDepartment or EngineeringDepartment@company.com. In this case, the mail server will decode the alias and send e-mail to all members of the alias.

The e-Mail adapter assumes that an e-mail address is correct and does not parse each individual address. Examples of valid address entries are:

- name@company.com
- EngineeringDepartment or EngineeringDepartment@company.com, where EngineeringDepartment is an alias defined in the mail server. The mail server decodes the alias and sends e-mail to all members of the alias.
- person1@some_company.com, person2@another_company.com. Multiple address can be entered and must be separated by commas.

If an e-mail address requires an at (@) sign, it must be included in the address because the adapter will not add characters to an e-mail address.

Appendix A. Standard configuration properties

This appendix describes the standard configuration properties for the connector component of WebSphere Business Integration adapters. The information covers connectors running on the following integration brokers:

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator, WebSphere MQ Integrator Broker, and WebSphere Business Integration Message Broker, collectively referred to as the WebSphere Message Brokers (WMQI).
- WebSphere Application Server (WAS)

Not every connector makes use of all these standard properties. When you select an integration broker from Connector Configurator, you will see a list of the standard properties that you need to configure for your adapter running with that broker.

For information about properties specific to the connector, see the relevant adapter user guide.

Note: In this document, backslashes (\) are used as the convention for directory paths. For UNIX installations, substitute slashes (/) for backslashes and follow the conventions for each operating system.

New and deleted properties

These standard properties have been added in this release.

New properties

- XMLNameSpaceFormat

Deleted properties

- RestartCount

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

You configure connector properties from Connector Configurator, which you access from System Manager. For more information on using Connector Configurator, refer to the Connector Configurator appendix.

Note: Connector Configurator and System Manager run only on the Windows system. If you are running the connector on a UNIX system, you must have a Windows machine with these tools installed. To set connector properties

for a connector that runs on UNIX, you must start up System Manager on the Windows machine, connect to the UNIX integration broker, and bring up Connector Configurator for the connector.

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 (only if WebSphere InterChange Server is the integration broker)
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. If the connector is working in stand-alone mode (independently of System Manager), for example with one of the WebSphere message brokers, you can only change properties through the configuration file. In this case, a dynamic update is not possible.

- 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 (ICS only)
  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 window, or see the Update Method column in the Property Summary table below.

Summary of standard properties

Table 14 on page 37 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 14. 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 MQ or IDL: Repository directory is <REMOTE>
AgentTraceLevel	0-5	0	Dynamic	
ApplicationName	Application name	Value specified for the connector application name	Component restart	
BrokerType	ICS, WMQI, WAS			
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	True	Dynamic	Repository directory is <REMOTE>
ControllerTraceLevel	0-5	0	Dynamic	Repository directory is <REMOTE>
DeliveryQueue		*CONNECTORNAME*/DELIVERYQUEUE	Component restart	JMS transport only
DeliveryTransport	MQ, IDL, or JMS	JMS	Component restart	If Repository directory is local, then value is JMS only
DuplicateEventElimination	True or False	False	Component restart	JMS transport only: Container Managed Events must be <NONE>
FaultQueue		*CONNECTORNAME*/FAULTQUEUE	Component restart	JMS transport only

Table 14. Summary of standard configuration properties (continued)

Property name	Possible values	Default value	Update method	Notes
jms.FactoryClassName	CxCommon.Messaging.jms .IBMMQSeriesFactory or CxCommon.Messaging .jms.SonicMQFactory or any Java class name	CxCommon.Messaging. jms.IBMMQSeriesFactory	Component restart	JMS transport only
jms.MessageBrokerName	If FactoryClassName is IBM, use crossworlds.queue. manager. If FactoryClassName is Sonic, use localhost:2506.	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>
ListenerConcurrency	1- 100	1	Component restart	Delivery Transport must be MQ
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	Repository Directory must be <REMOTE>
MaxEventCapacity	1-2147483647	2147483647	Dynamic	Repository Directory must be <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 must be <REMOTE>

Table 14. Summary of standard configuration properties (continued)

Property name	Possible values	Default value	Update method	Notes
OADMaxNumRetry	A positive number	1000	Dynamic	Repository Directory must be <REMOTE>
OADRetryTimeInterval	A positive number in minutes	10	Dynamic	Repository Directory must be <REMOTE>
PollEndTime	HH:MM	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	For ICS: set to <REMOTE> For WebSphere MQ message brokers and WAS: set to C:\crossworlds\ repository
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: required only if Repository directory is <REMOTE>
RestartRetryCount	0-99	3	Dynamic	
RestartRetryInterval	A sensible positive value in minutes: 1 - 2147483547	1	Dynamic	
RHF2MessageDomain	mrm, xml	mrm	Component restart	Only if Delivery Transport is JMS and WireFormat is CwXML.

Table 14. Summary of standard configuration properties (continued)

Property name	Possible values	Default value	Update method	Notes
SourceQueue	Valid WebSphere MQ name	*CONNECTORNAME*/SOURCEQUEUE	Agent restart	Only if Delivery Transport is JMS and Container Managed Events is specified
SynchronousRequestQueue		*CONNECTORNAME*/ SYNCHRONOUSREQUESTQUEUE	Component restart	Delivery Transport is JMS
SynchronousRequestTimeout	0 - any number (millisecs)	0	Component restart	Delivery Transport is JMS
SynchronousResponseQueue		*CONNECTORNAME*/ SYNCHRONOUSRESPONSEQUEUE	Component restart	Delivery Transport is JMS
WireFormat	CwXML, CwBO	CwXML	Agent restart	CwXML if Repository Directory is not <REMOTE>: CwBO if Repository Directory is <REMOTE>
WsifSynchronousRequest Timeout	0 - any number (millisecs)	0	Component restart	WAS only
XMLNameSpaceFormat	short, long	short	Agent restart	WebSphere MQ message brokers and WAS only

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

Applicable only if RepositoryDirectory is <REMOTE>.

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 type that you are using. The options are ICS, WebSphere message brokers (WMQI, WMQIB or WBIMB) or WAS.

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 Connector Configurator.

ConcurrentEventTriggeredFlows

Applicable only if `RepositoryDirectory` is <REMOTE>.

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.

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. 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.

This property only appears if the `DeliveryTransport` property is set to the value JMS.

ControllerStoreAndForwardMode

Applicable only if `RepositoryDirectory` is <REMOTE>.

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 ICS, 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

Applicable only if `RepositoryDirectory` is <REMOTE>.

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 the integration broker.

The default value is `CONNECTORNAME/DELIVERYQUEUE`.

DeliveryTransport

Specifies the transport mechanism for the delivery of events. Possible values are `MQ` for WebSphere MQ, `IDL` for CORBA IIOP, or `JMS` for Java Messaging Service.
- If ICS is the broker type, the value of the `DeliveryTransport` property can be MQ, IDL, or JMS, and the default is IDL.
- If the `RepositoryDirectory` is a local directory, the value may only be JMS.

The connector sends service call requests and administrative messages over CORBA IIOP if the value configured for the `DeliveryTransport` property is `MQ` or `IDL`.

WebSphere MQ and IDL

Use WebSphere MQ rather than IDL for event delivery transport, unless you must have only one product. WebSphere MQ offers the following advantages over IDL:
- Asynchronous communication:
  WebSphere MQ allows the application-specific component to poll and persistently store events even when the server is not available.
- Server side performance:
  WebSphere MQ provides faster performance on the server side. In optimized mode, WebSphere MQ stores only the pointer to an event in the repository database, while the actual event remains in the WebSphere MQ queue. This saves having to write potentially large events to the repository database.
- Agent side performance:
  WebSphere MQ provides faster performance on the application-specific component side. Using WebSphere MQ, the connector's polling thread picks up an event, places it in the connector's queue, then picks up the next event. This is faster than IDL, which requires the connector's polling thread to pick up an event, go over the network into the server process, store the event persistently in the repository database, then pick up the next event.

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. 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 in the following environment:
- AIX 5.0
- WebSphere MQ 5.3.0.1
- When ICS is the integration broker

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. If your installation uses less than 768M of process heap size, IBM recommends that you set:

- The `LDR_CNTRL` environment variable in the `CWSharedEnv.sh` script.

  This script resides in the `\bin` directory below the product directory. With a text editor, add the following line as the first line in the `CWSharedEnv.sh` script:

  `export LDR_CNTRL=MAXDATA=0x30000000`

  This line restricts heap memory usage to a maximum of 768 MB (3 segments * 256 MB). If the process memory grows more than this limit, page swapping can occur, which can adversely affect the performance of your system.

- The `IPCCBaseAddress` property to a value of 11 or 12. For more information on this property, see the *System Installation Guide for UNIX*.

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.

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). This property is applicable only if the `RepositoryDirectory` value is <REMOTE>.

The default value is 128m.

JvmMaxNativeStackSize

The maximum native stack size for the agent (in kilobytes). This property is applicable only if the `RepositoryDirectory` value is <REMOTE>.

The default value is 128k.

JvmMinHeapSize

The minimum heap size for the agent (in megabytes). This property is applicable only if the `RepositoryDirectory` value is <REMOTE>.

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 (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.

ListenerConcurrency

This property supports multi-threading in MQ Listener when ICS is the integration broker. It enables batch writing of multiple events to the database, thus improving system performance. The default value is 1.

This property applies only to connectors using MQ transport. The `DeliveryTransport` property must be set to MQ.

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 Connector Configurator.

The default value is en_US. If the connector has not been globalized, the only valid value for this property is en_US. To determine whether a specific connector has been globalized, see the connector version list on these websites:

http://www.ibm.com/software/websphere/wbiadapters/infocenter, or
http://www.ibm.com/websphere/integration/wicserver/infocenter

LogAtInterchangeEnd

Applicable only if RespositoryDirectory is <REMOTE>.

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 and is applicable only if the value of the RepositoryDirectory property is <REMOTE>.

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

Valid only when the `RepositoryDirectory` is <REMOTE>.

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* or *for UNIX*.

The default value is `false`.

OADMaxNumRetry

Valid only when the `RepositoryDirectory` is <REMOTE>.

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

Valid only when the `RepositoryDirectory` is <REMOTE>.

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 the integration broker 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.

When the integration broker is ICS, this value must be set to <REMOTE> because the connector obtains this information from the InterChange Server repository.

When the integration broker is a WebSphere message broker or WAS, this value must be set to *<local directory>*.

ResponseQueue

Applicable only if DeliveryTransport is JMS and required only if RepositoryDirectory is <REMOTE>.

Designates the JMS response queue, which delivers a response message from the connector framework to the integration broker. When the integration broker is ICS, the server 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.

RHF2MessageDomain

WebSphere message brokers and WAS only.

This property allows you to configure the value of the field domain name in the JMS header. When data is sent to WMQI over JMS transport, the adapter framework writes JMS header information, with a domain name and a fixed value of `mrm`. A connfigurable domain name enables users to track how the WMQI broker processes the message data.

A sample header would look like this:

```
<mcd><Msd>mrm</Msd><Set>3</Set><Type>
Retek_POPhyDesc</Type><Fmt>CwXML</Fmt></mcd>
```

The default value is `mrm`, but it may also be set to `xml`. This property only appears when DeliveryTransport is set to JMSand WireFormat is set to `CwXML`.

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 42.

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

Message format on the transport.
- If the RepositoryDirectory is a local directory, the setting is CwXML.
- If the value of RepositoryDirectory is <REMOTE>, the setting isCwBO.

WsifSynchronousRequest Timeout

WAS integration broker only.

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.

XMLNameSpaceFormat

WebSphere message brokers and WAS integration broker only.

A strong property that allows the user to specify short and long name spaces in the XML format of business object definitions.

The default value is short.

Appendix B. Connector Configurator Express

This appendix describes how to use Connector Configurator to set configuration property values for your adapter.

You use Connector Configurator to:
- Create a connector-specific property template for configuring your connector
- Create a configuration file
- Set properties in a configuration file

Note:
> In this document, backslashes (\) are used as the convention for directory paths. For UNIX installations, substitute slashes (/) for backslashes and follow the conventions for each operating system.

The topics covered in this appendix are:
- "Overview of Connector Configurator" on page 51
- "Starting Connector Configurator" on page 52
- "Creating a connector-specific property template" on page 53
- "Creating a new configuration file" on page 55
- "Setting the configuration file properties" on page 58
- "Using Connector Configurator in a globalized environment" on page 64

Overview of Connector Configurator

Connector Configurator allows you to configure the connector component of your adapter for use with these integration brokers:
- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator, WebSphere MQ Integrator Broker, and WebSphere Business Integration Message Broker, collectively referred to as the WebSphere Message Brokers (WMQI)
- WebSphere Application Server (WAS)

You use Connector Configurator 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, with ICS, maps for use with collaborations as well as specify messaging, logging and tracing, and data handler parameters, as required.

The mode in which you run Connector Configurator, and the configuration file type you use, may differ according to which integration broker you are running. For example, if WMQI is your broker, you run Connector Configurator directly, and not from within System Manager (see "Running Configurator in stand-alone mode" on page 52).

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 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.

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 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 53 to set up a new one.

Note: Connector Configurator runs only in a Windows environment. If you are running the connector in a UNIX environment, use Connector Configurator in Windows to modify the configuration file and then copy the file to your UNIX environment.

Starting Connector Configurator

You can start and run Connector Configurator in either of two modes:
- Independently, in stand-alone mode
- From System Manager

Running Configurator in stand-alone mode

You can run Connector Configurator independently and work with connector configuration files, irrespective of your broker.

To do so:
- From **Start>Programs**, click **IBM WebSphere InterChange Server>IBM WebSphere Business Integration Toolset>Development>Connector Configurator**.
- Select **File>New>Configuration File**.
- When you click the pull-down menu next to **System Connectivity Integration Broker**, you can select ICS, WebSphere Message Brokers or WAS, depending on your broker.

You may choose to run Connector Configurator 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 57.)

Running Configurator from System Manager

You can run Connector Configurator from System Manager.

To run Connector Configurator:

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**. The Connector Configurator window opens and displays a **New Connector** dialog box.
4. When you click the pull-down menu next to **System Connectivity Integration Broker**, you can select ICS, WebSphere Message Brokers or WAS, depending on your broker.

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. Connector Configurator opens and displays the configuration file with the integration broker type and file name at the top.
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 53.
- 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 **Dependences - 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 stores the information you have entered as an XML document, under `\data\app` in the`\bin` directory where you have installed Connector Configurator.

Creating a new configuration file

When you create a new configuration file, your first step is to select an integration broker. The broker you select determines the properties that will appear in the configuration file.

To select a broker:
- In the Connector Configurator home menu, click **File>New>Connector Configuration**. The **New Connector** dialog box appears.
- In the I**ntegration Broker** field, select ICS, WebSphere Message Brokers or WAS connectivity.
- Complete the remaining fields in the **New Connector** window, as described later in this chapter.

You can also do this:
- In the System Manager window, right-click on the **Connectors** folder and select **Create New Connector**. Connector Configurator opens and displays the **New Connector** dialog box.

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 does not check the spelling of the name that you enter. You must ensure that the name is correct.

   - **System Connectivity**

     Click ICS or WebSphere Message Brokers or WAS.

   - **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 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 window, as described later in this chapter.

Using an existing file

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 ICS repository file.
  Definitions used in a previous ICS 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, 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, 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`)
   - ICS Repository (`*.in`, `*.out`)

     Choose this option if a repository file was used to configure the connector in an ICS environment. 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.
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 window displays the configuration screen, with the current attributes and values.

The title of the configuration screen displays the integration broker and connector name as specified in the file. Make sure you have the correct broker. If not, change the broker value before you configure the connector. To do so:

1. Under the **Standard Properties** tab, select the value field for the BrokerType property. In the drop-down menu, select the value ICS, WMQI, or WAS.
2. The Standard Properties tab will display the properties associated with the selected broker. You can save the file now or complete the remaining configuration fields, as described in "Specifying supported business object definitions" on page 60..
3. When you have finished your configuration, click **File>Save>To Project** or **File>Save>To File**.

   If you are saving to file, select `*.cfg` as the extension, select the correct location for the file and click **Save**.

If multiple connector configurations are open, click **Save All to File** to save all of the configurations to file, or click **Save All to Project** to save all connector configurations to a System Manager project.

Before it saves the file, Connector Configurator checks that values have been set for all required standard properties. If a required standard property is missing a value, Connector Configurator displays a message that the validation failed. You must supply a value for the property in order to save the configuration file.

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 displays a configuration screen with tabs for the categories of required configuration values.

Connector Configurator requires values for properties in these categories for connectors running on all brokers:

- Standard Properties
- Connector-specific Properties
- Supported Business Objects
- Trace/Log File values
- Data Handler (applicable for connectors that use JMS messaging with guaranteed event delivery)

Note: For connectors that use JMS messaging, an additional category may display, for configuration of data handlers that convert the data to business objects.

For connectors running on **ICS**, values for these properties are also required:

- Associated Maps
- Resources
- Messaging (where applicable)

Important: Connector Configurator accepts property values in either English or non-English character sets. However, the names of both standard and connector-specific properties, and the names of supported business objects, must use the English character set only.

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.
- Value fields are configurable.
- The **Update Method** field is informational and not configurable. This field specifies the action required to activate a property whose value has changed.

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, click **File>Exit** (or close the window), and click **No** when prompted to save changes.
   - To enter values for other categories in Connector Configurator, 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."

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 36.

Specifying supported business object definitions

Use the **Supported Business Objects** tab in Connector Configurator 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.

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 *Connector Development Guide for C++* or the *Connector Development Guide for Java*.

If ICS is your broker

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 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 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 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.

If a WebSphere Message Broker is your broker

If you are working in stand-alone mode (not connected to System Manager), you must enter the business name manually.

If you have System Manager running, you can select the empty box under the **Business Object Name** column in the **Supported Business Objects** tab. A combo box appears with a list of the business object available from the Integration Component Library project to which the connector belongs. Select the business object you want from the list.

The **Message Set ID** is an optional field for WebSphere Business Integration Message Broker 5.0, and need not be unique if supplied. However, for WebSphere MQ Integrator and Integrator Broker 2.1, you must supply a unique **ID**.

If WAS is your broker

When WebSphere Application Server is selected as your broker type, Connector Configurator does not require message set IDs. The **Supported Business Objects** tab shows a **Business Object Name** column only for supported business objects.

If you are working in stand-alone mode (not connected to System Manager), you must enter the business object name manually.

If you have System Manager running, you can select the empty box under the Business Object Name column in the Supported Business Objects tab. A combo box appears with a list of the business objects available from the Integration Component Library project to which the connector belongs. Select the business object you want from this list.

Associated maps (ICS only)

Each connector supports a list of business object definitions and their associated maps that are currently active in WebSphere InterChange Server. 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 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 ICS 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 window, click **Save to Project**.
  4. Deploy the project to ICS.
  5. Reboot the server for the changes to take effect.

Resources (ICS)

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.

Messaging (ICS)

The messaging properties are available only if you have set MQ as the value of the DeliveryTransport standard property and ICS as the broker type. These properties affect how your connector will use queues.

Setting trace/log file values

When you open a connector configuration file or a connector definition file, Connector Configurator uses the logging and tracing values of that file as default values. You can change those values in Connector Configurator.

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. Not all adapters make use of data handlers.

See the descriptions under ContainerManagedEvents in Appendix A, Standard Properties, for values to use for these properties. For additional details, see the *Connector Development Guide for C++* or the *Connector Development Guide for Java.*

Saving your configuration file

When you have finished configuring your connector, save the connector configuration file. Connector Configurator saves the file in the broker mode that you selected during configuration. The title bar of Connector Configurator always displays the broker mode (ICS, WMQI or WAS) that it is currently using.

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 following implementation guides:

- For ICS: *Implementation Guide for WebSphere InterChange Server*
- For WebSphere Message Brokers: *Implementing Adapters with WebSphere Message Brokers*
- For WAS: *Implementing Adapters with WebSphere Application Server*

Changing a configuration file

You can change the integration broker setting for an existing configuration file. This enables you to use the file as a template for creating a new configuration file, which can be used with a different broker.

Note: You will need to change other configuration properties as well as the broker mode property if you switch integration brokers.

To change your broker selection within an existing configuration file (optional):

- Open the existing configuration file in Connector Configurator.
- Select the **Standard Properties** tab.
- In the **BrokerType** field of the Standard Properties tab, select the value that is appropriate for your broker.
  When you change the current value, the available tabs and field selections on the properties screen will immediately change, to show only those tabs and fields that pertain to the new broker you have selected.

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 in a globalized environment

Connector Configurator is globalized and can handle character conversion between the configuration file and the integration broker. Connector Configurator uses native encoding. When it writes to the configuration file, it uses UTF-8 encoding.

Connector Configurator 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>
```

Notices

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

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

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

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Burlingame Laboratory Director
IBM Burlingame Laboratory
577 Airport Blvd., Suite 800

67

Burlingame, CA 94010
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

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

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

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

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

COPYRIGHT LICENSE

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

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:

IBM
the IBM logo
AIX
CrossWorlds
DB2
DB2 Universal Database
Domino
Lotus
Lotus Notes
MQIntegrator
MQSeries
Tivoli
WebSphere

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

MMX, Pentium, and ProShare are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

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

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

System Manager includes software developed by the Eclipse Project (http://www.eclipse.org/)

WebSphere Business Integration Server Express V4.3.1 and WebSphere Business Integration Server Express Plus V4.3.1

IBM ®

Printed in USA