WebSphere Business Integration Server
Express Plus

IBM

Adapter for iSeries User Guide

Version 4.3.1

WebSphere Business Integration Server
Express Plus

IBM

Adapter for iSeries User Guide

Version 4.3.1

> **Note!**
> Before using this information and the product it supports, read the information in "Notices" on page 49.

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

IBM^(R) WebSphere Business Integration Server Express Plus is 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 configuration, business object development, and troubleshooting for the Adapter for iSeries.

Audience

This document is for WebSphere consultants and customers who are implementing the connector as part of a WebSphere business-integration system. To use the information in this document, you should be knowledgeable in the following areas:

- Connector development
- Business object development
- OS/400 application architecture
- Basics of iSeries integrated file system

Related documents

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

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

Note: Important information about this product may be available in Technical Support Technotes and Flashes issued after this document was published. These can be found on the WebSphere Business Integration Support Web site, 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 variable name or a cross-reference.
blue outline	A blue outline, which is visible only when you view the manual online, indicates a cross-reference hyperlink. Click inside the outline to jump to the object of the reference.

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

Version 4.3.1

This release adds support for the following operating systems:

- IBM OS/400 V5R2, V5R3
- Red Hat Enterprise Linux AS 3.0 Update 1
- SuSE Linux Enterprise Server 8.1
- Microsoft Windows 2003 (for InterChange Server Express and adapters only, in production mode)

This release also adds support for any Integrated Language Environment (ILE) program, in addition to the Report Program Generator (RPG).

Version 4.3

This is the first release of this guide.

Chapter 1. Overview

This chapter describes the Adapter for iSeries component of the WebSphere Business Integration Server Express and Express Plus. The adapter, using the IBM Toolbox for Java (a set of Java$^{(TM)}$ classes), provides the ability to execute any **Integrated Language Environment** (ILE) programs on any iSeries running the OS/400 operating system. IBM's Toolbox for Java provides a set of classes to access and run these programs. The adapter uses these classes and information from the incoming business object to build the parameter list for the program and then execute the program. The iSeries adapter currently supports only request processing

Adapters consist of an application-specific component and the connector framework. The application-specific component contains code tailored to a particular application. The connector framework, whose code is common to all adapters, acts as an intermediary between the integration broker and the application-specific component. The connector framework provides the following services between the integration broker and the application-specific component:

- Sends business objects.
- Manages the exchange of startup and administrative messages.

This document contains information about the application-specific component and connector framework. It refers to both of these components as the adapter.

For more information about the relationship of the integration broker to the adapter, see the *IBM WebSphere Business Integration Server Express System Administration Guide, or the IBM WebSphere Business Integration Server Express Implementation Guide*.

Note: All WebSphere Business Integration Server Express adapters operate with InterChange Server Express as their integration broker.

This chapter contains the following sections:
- "An overview of the OS/400 system"
- "How the adapter works" on page 3

An overview of the OS/400 system

The IBM eServer iSeries system is a highly integrated, reliable server platform that allows businesses to run multiple operating environments simultaneously. Their integrity and security characteristics allow them to be used in many critical applications. They are mid-range computers, reliable, and able to handle hundreds of users.

Host servers handle requests from client PCs or other devices running an application as illustrated in Figure 1 on page 2 to enable printing a document and other tasks. The OS/400 computers are full function servers capable of performing many tasks at once, including file, database, applications, mail, print, multimedia, fax and wireless communications. Each task server runs as a separate job on the system, and each server job sends and receives data streams on a socket connection.

One of these host servers is the Remote Command and Distributed program call server. This server handles running the programs on an OS/400 system.

IBM's Toolbox for JAVA has a multitude of packages which handle different functionalities. For example Access classes manage sign-on information, create and maintain socket connections, send and receive data, while Command Call classes run OS/400 batch commands.

IBM's iSeries adapter uses the Access classes, and Program Call classes to call the ILE program. Data conversion classes provide the capability to convert numeric and character data between OS/400 and Java formats.
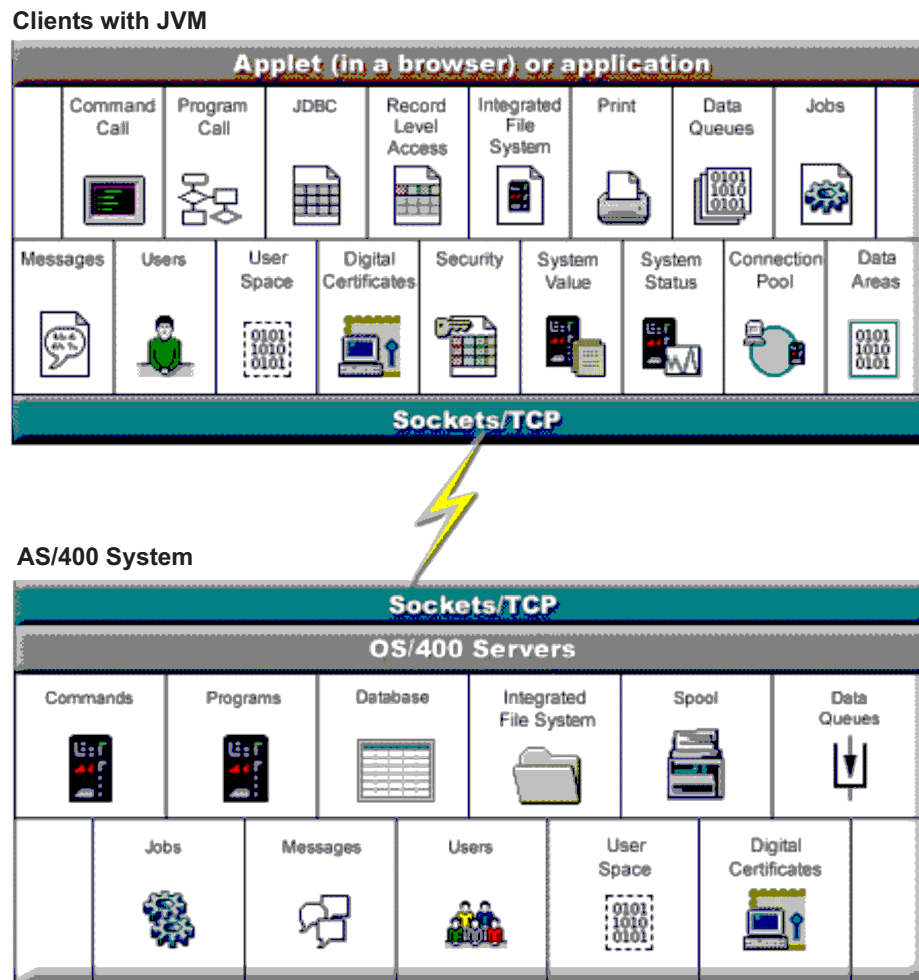

Figure 1. Overview of OS/400 client - server architecture

While OS/400 is capable of handling many types of tasks, the iSeries adapter only uses the Remote Command and Distributed program call server. This server handles running the programs on the OS/400 system.

A diagram of the iSeries Adapter connection of the client to the server is shown below in Figure 2 on page 3.
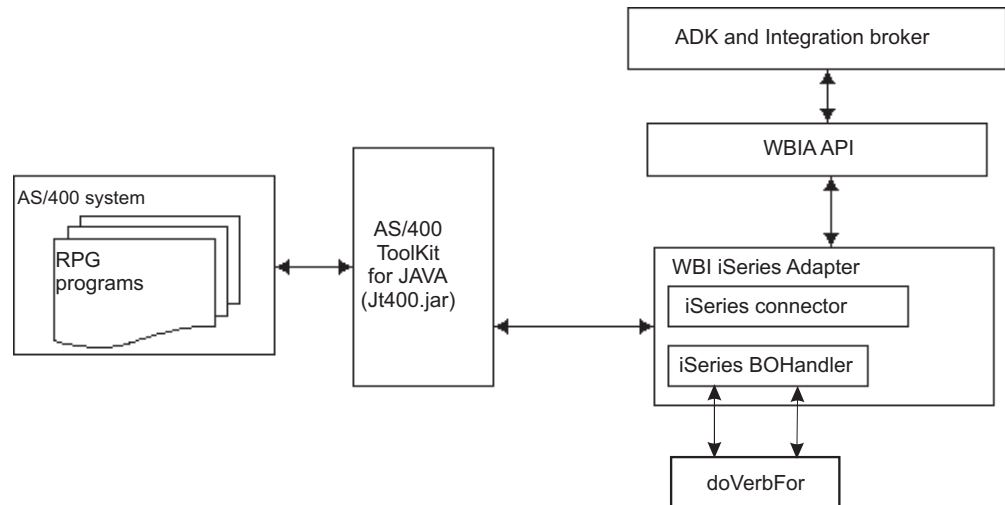
Figure 2. Diagram of the iSeries adapter's connections.

How the adapter works

The following sections describe how the adapter processes business objects.

Business object processing

The adapter receives business object requests from InterChange Server Express and builds the parameter list for the ILE program, then establishes a connection with an iSeries or OS/400 system and executes the program.

The incoming business object has the connection child attribute. The information in this attribute is used to connect to the OS/400 system.

Connector operations

The connector passes business objects between InterChange Server Express and a business object handler. The connector:

1. Registers the BOHandler with the framework.
2. The framework sends the BO request to the BOHandler.
3. The BOHandler uses the attribute information in the incoming business object to build the RPG program parameter list.
4. The BOHandler calls the ILE program running on the OS/400 system.

   Note: This is essentially a call to execute a program on the OS/400 system which then returns a success or fail message.
5. The BOHandler then returns the results of the execution of the RPG program to the adapter framework. It also populates the business object with the returned parameters.

The adapter is written in Java and consists of two components:

- Connector
- BO handler

Processing locale-dependant 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 Server Express system are written in Java. Therefore, when data is transferred between most WebSphere Business Integration Server Express 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 configuration properties, see Appendix A, "Standard configuration properties for connectors," on page 21.

Chapter 2. Installing and configuring the iSeries adapter

This chapter describes the process of installing and configuring the connector. It contains the following sections:

Prerequisites

To use the connector, your environment must have the following:

1. Java and jar files:
   - JDK 1.3 or later
   - Java Secure Socket Extension 1.0 (JSSE)
   - Jt400.jar file.

     Note: The IBM Toolbox for Java (licensed product 5722-JC1) V5R2 file can be downloaded from the Toolbox web site at: http://www-1.ibm.com/servers/eserver/iseries/toolbox/downloads.htm. The jt400.jar needs to be copied to the %Product_dir%\connectors\iSeries directory. Note that the jt400.jar does not currently support internationalization.

   - WBIA.jar file.
   - CrossWorlds.jar file.
   - BIA_iSeries.jar file.
2. The iSeries adapter is designed to connect to one of the following OS/400 versions:
   - Version 5, Releases 1 through 3
   - Version 4, Releases 1 through 3
3. The Host Servers option of OS/400 must be installed and running.

   Note: The OS/400 data queue server requires PTFs to correctly perform peek functions. You must have the appropriate PTF from the following link:
   ```
   http://www-1.ibm.com/servers/eserver/iseries/toolbox/
   hostservicepackdetail.htm
   ```

Installing the iSeries adapter and related files

For information on installing the adapter, refer to the see the *Installation Guide for WebSphere Business Integration Server Express*, 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 1 describes the Windows file structures, Table 2 describes the OS/400 file structures, and Table 3 on page 7 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 1. Installed Windows file structure for the connector

Subdirectory of %ProductDir%	Description
connectors\iSeries	Contains the connector BIA_iSeries.jar and the start_iSeries.bat and start_iSeries_service.bat.
connectors\messages	Contains the BIA_iSeriesAdapter.txt file.
repository\iSeries	Contains the BIA_CN_iSeries.txt file.
\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 2. Installed OS/400 file structure for the connector

Subdirectory of %ProductDir%	Description
connectors/iSeries	Contains the connector BIA_iSeries.jar and the start_iSeries.sh files.
connectors/messages	Contains the BIA_iSeriesAdapter.txt file.
repository/iSeries	Contains the BIA_CN_iSeries.txt file.
/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 3. Installed Linux file structure for the connector

Subdirectory of %ProductDir%	Description
connectors/iSeries	Contains the connector BIA_iSeries.jar and the start_iSeries.sh.files.
connectors/messages	Contains the BIA_iSeriesAdapter.txt file.
repository/iSeries	Contains the BIA_CN_iSeries.txt file.
/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.

Configuring the connector

The iSeries adapter uses standard connector properties for configuration as detailed in the next section and connector-specific properties as detailed in the following section.

Standard connector properties

Standard configuration properties provide information that all connectors use. See Appendix A, "Standard configuration properties for connectors," on page 21 for documentation of these properties.

You must set at least the following standard connector configuration properties before running the connector.

- AgentTraceLevel
- ApplicationName
- ControllerStoreAndForwardMode
- ControllerTraceLevel
- DeliveryTransport

Connector-specific properties

Connector-specific configuration properties provide information needed by the connector at runtime. Connector-specific properties also provide a way of changing static information or logic within the connector agent without having to recode and rebuild the agent.

Table 4 lists the connector-specific configuration properties for the connector. See the section that follows for explanations of the properties.

Table 4. Connector-specific configuration properties

Name	Possible values	Default value	Required?
ExecutionTimeout		None	Yes
UseDefaults	*default value*	None	Yes
PollQuantity	An integer greater than 1.	1	No

ExecutionTimeout
This is the Timeout where a connector calls the pgm program on an iSeries system.

UseDefaults

Typically, some of the input parameters to a program would be constant. So these attributes can be designed to have default values. If there is no default value and the UseDefaults property is set to true, the adapter errors out and throws a `VerbProcessingFailedException`. If UseDefaults is not set or set to `false`, and there are no default values, the adapter builds a String of length `MaxLength` with padded spaces for the attribute values.

PollQuantity

Designates the number of items that the connector polls for. If the adapter has set a connector-specific property for the poll quantity, the value set in the connector-specific property will override the standard property value.

Creating multiple connector instances

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`

  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`

- 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. QWBIDFT is the default name.

  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 following directory:

   - Windows:

     ```
     ProductDir\repository\initialConnectorInstance
     ```

     Any additional files you create should be in the appropriate connectorInstance 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=${CONDIR}/BIA_${CONNAME}.jar" to "CONJAR=${CONDIR}/BIA_iSeries.jar"

2. Put this startup script in the connector directory you created in "Create a new directory" on page 8.

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(QWBIISRSC) FROMLIB(QWBISVR43) OBJTYPE(*JOBD) TOLIB(QWBISVR43)
NEWOBJ(newiSeriesname)
```

where `newiSeriesname` 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 5 shows.

Table 5. Startup scripts for a connector

Operating system	Startup script
Windows	`start_connName`.bat
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.
OS/400	`start_connName`.sh

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 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 the following script in the QSHELL
  environment:
  /QIBM/ProdData/WBIServer43/bin/add_autostart_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.

- From System Monitor

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

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 the name of the InterChange Server Express instance. By default, the name of the InterChange Server Express instance 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>CWConnectorWBIiSeriesAdapter).

Stopping the connector from OS/400

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

- 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.
- Using start_connName.sh

  If you started the connector using the start_connName.sh script in QSHELL from OS/400 command entry, you simply press F3 to exit the QSHELL environment and the adapter will end.
- From System Monitor

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

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. Developing business objects for the connector

This chapter describes the structure of a business object passed by the connector. It also describes the required attributes and describes how the connector processes the business object. It contains the following sections:

- "Connector business object processing"
- "Connector business object structure"

Connector business object processing

The connector passes business objects between InterChange Server Express and the OS/400 system.

When InterChange Server Express passes a business object to the connector, the connector performs the following operations:

1. Uses the information in the connection child attribute to connect to the OS/400 system.
2. Builds the parameter list for the RPG program based on the attributes in the BO.
3. Executes the RPG program corresponding to the BO.
4. Returns the result of executing the program: success or failure.

Connector business object structure

The business object for the iSeries adapter is a flat business object. The attributes can be input, output or inout parameters. One of the attributes needs to be a key for the business object designer.

```
┌─────────────────────────────────┐
│                                 │
│    iSeries Adapter BO           │
│                                 │
│    Supported Verb =             │
│     CALLRPG                     │
│                                 │
│    Connection                   │
│    Input parameters             │
│    Output parameters            │
│                                 │
└─────────────────────────────────┘
```
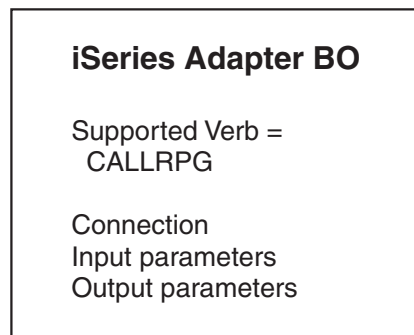
Figure 3. The iSeries parent business object.

Also, there is a child attribute of type Connection. This will in turn have information about the connection to the OS/400 machine; the `HostName`, `UserName` and `Password`.
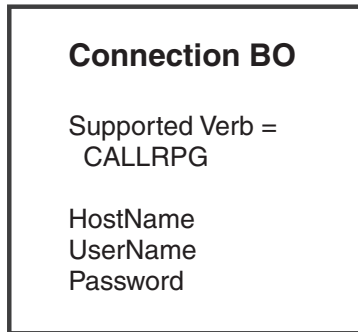
```
┌─────────────────────────┐
│                         │
│     Connection BO       │
│                         │
│   Supported Verb =      │
│     CALLRPG             │
│                         │
│   HostName              │
│   UserName              │
│   Password              │
│                         │
└─────────────────────────┘
```

Figure 4. The iSeries child business object.

```
┌─────────────────────────────────────────┐
│                                          │
│   Verb = CALLRPG                         │
│                                          │
├─────────────────────────────────────────┤
│                                          │
│   Connection instance                    │
│        HostName = (host).(domain).com    │
│        UserName = (username)             │
│        Password = (password)             │
│                                          │
├─────────────────────────────────────────┤
│   Attribute1                             │
│      [ParamType=Input;                   │
│      Signed=true;                        │
│      Datalength=2]                       │
│                                          │
└─────────────────────────────────────────┘
```
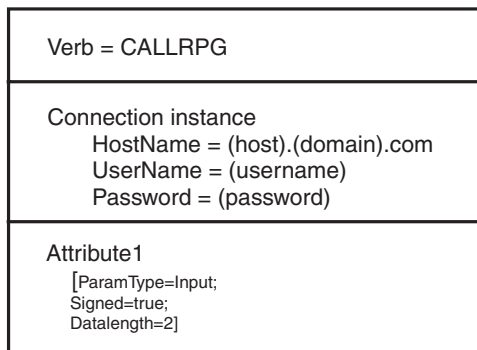
Figure 5. Example of a business object

The information within brackets in the above diagram represents the application specific information.

Use Business Object Designer Express to create the business objects. Create the business object definition, and add the required attributes. Then configure the connector to support the business object.

Business object Connection XSD schema

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
- <xsd:schema elementFormDefault="qualified"
targetNamespace="http://www.ibm.com/websphere/crossworlds
2002/BOSchema/Connection"
xmlns:bx=
"http://www.ibm.com/websphere/crossworlds/2002/BOSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
- <xsd:annotation>
<xsd:documentation>Fri Jun 06 21:20:50 IST 2003
</xsd:documentation>
</xsd:annotation>
- <xsd:element name="Connection">
- <xsd:annotation>
- <xsd:appinfo>
<bx:boDefinition version="3.0.0" />
</xsd:appinfo>
</xsd:annotation>
- <xsd:complexType>
```

```
- <xsd:sequence>
- <xsd:element name="UserName" default="[insert username]" minOccurs="1">
- <xsd:annotation>
- <xsd:appinfo>
<bx:boAttribute>
<bx:attributeInfo isForeignKey="false" isKey="true" />
</bx:boAttribute>
</xsd:appinfo>
</xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:string">
<xsd:maxLength value="255" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
- <xsd:element name="Password" default="[insert password]" minOccurs="1">
- <xsd:annotation>
- <xsd:appinfo>
- <bx:boAttribute>
<bx:attributeInfo isForeignKey="false" isKey="true" />
</bx:boAttribute>
</xsd:appinfo>
</xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:string">
<xsd:maxLength value="255" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
- <xsd:element name=
"HostName" default="(hostname).(domain name).com" minOccurs="1">
- <xsd:annotation>
- <xsd:appinfo>
- <bx:boAttribute>
<bx:attributeInfo isForeignKey="false" isKey="true" />
</bx:boAttribute>
</xsd:appinfo>
</xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:string">
<xsd:maxLength value="255" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element name=
"ObjectEventId" type="xsd:string" minOccurs="0" />
</xsd:sequence>
<xsd:attribute name="version" type="xsd:token" default="0.0.0" />
<xsd:attribute name="delta" type="xsd:boolean" default="false" />
- <xsd:attribute name="verb" use="required">
- <xsd:simpleType>
- <xsd:restriction base="xsd:NMTOKEN">
<xsd:enumeration value="CALLRPG" />
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
```

```
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

Business object PAS400rpg2 XSD schema

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
- <xsd:schema elementFormDefault="qualified
targetNamespace="http://www.ibm.com/websphere
/crossworlds/2002/BOSchema/PAS400rpg2"
xmlns:Connection=
"http://www.ibm.com/websphere/crossworlds/2002
/BOSchema/Connection"
xmlns:bx="http://www.ibm.com/websphere/crossworlds/2002
/BOSchema" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
- <xsd:annotation>
<xsd:documentation>Fri Jun 06 21:20:50 IST 2003</xsd:documentation>
</xsd:annotation>
<xsd:import namespace=
"http://www.ibm.com/websphere/crossworlds/2002
/BOSchema/Connection" schemaLocation="Connection.xsd" />
- <xsd:element name="PAS400rpg2">
- <xsd:annotation>
- <xsd:appinfo>
<bx:boDefinition version="3.0.0" />
</xsd:appinfo>
</xsd:annotation>
- <xsd:complexType>
- <xsd:sequence>
- <xsd:element name="attr1" minOccurs="1">
- <xsd:annotation>
- <xsd:appinfo>
- <bx:boAttribute>
<bx:appSpecificInfo>
ParamType=Input;PackedDec=true;DecimalPositions=0
</bx:appSpecificInfo>
<bx:attributeInfo isForeignKey="false" isKey="false" />
</bx:boAttribute>
</xsd:appinfo>
</xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:string">
<xsd:maxLength value="3" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
- <xsd:element name="attr2" minOccurs="1">
- <xsd:annotation>
- <xsd:appinfo>
- <bx:boAttribute>
<bx:appSpecificInfo>
ParamType=Input;PackedDec=true;DecimalPositions=0
</bx:appSpecificInfo>
<bx:attributeInfo isForeignKey="false" isKey="false" />
</bx:boAttribute>
</xsd:appinfo>
```

```
</xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:string">
<xsd:maxLength value="2" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
- <xsd:element name="attr3" minOccurs="0">
- <xsd:annotation>
- <xsd:appinfo>
- <bx:boAttribute>
<bx:appSpecificInfo>
ParamType=Inout;PackedDec=true;DecimalPositions=0
</bx:appSpecificInfo>
<bx:attributeInfo isForeignKey="false" isKey="false" />
</bx:boAttribute>
</xsd:appinfo>
</xsd:annotation>
- <xsd:simpleType>
- <xsd:restriction base="xsd:string">
<xsd:maxLength value="3" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
- <xsd:element name="Connection" minOccurs="1">
- <xsd:annotation>
- <xsd:appinfo>
- <bx:boAttribute>
<bx:appSpecificInfo>ParamType=Input</bx:appSpecificInfo>
<bx:attributeInfo isForeignKey="false" isKey="true" />
<bx:childObjectInfo relationship="Containment" version="3.0.0" />
</bx:boAttribute>
</xsd:appinfo>
</xsd:annotation>
- <xsd:complexType>
- <xsd:sequence>
<xsd:element ref="Connection:Connection" />
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:element name="ObjectEventId" type="xsd:string" minOccurs="0" />
</xsd:sequence>
<xsd:attribute name="version" type="xsd:token" default="0.0.0" />
<xsd:attribute name="delta" type="xsd:boolean" default="false" />
- <xsd:attribute name="verb" use="required">
- <xsd:simpleType>
+ <xsd:restriction base="xsd:NMTOKEN">
- <xsd:enumeration value="CALLRPG">
- <xsd:annotation>
- <xsd:appinfo>
- <bx:boVerb>
<bx:appSpecificInfo>
/QSYS.LIB/XYZ.LIB/RPG3.PGM
</bx:appSpecificInfo>
</bx:boVerb>
</xsd:appinfo>
```

```
</xsd:annotation>
</xsd:enumeration>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

Business Object Attribute Level Application Text

The following information is part of the business object attribute level application text.

Table 6. BO Attributes

Property	Values	Description
ParamType	Input/Output/InOut	Indicates what type of parameter the attribute represents.
Offset	Any integer value	Indicates the offset in the byte array from where the parameter value starts.
Signed	True/false	For integer /short/long types, the property indicates if it is signed. If not set, the value is taken as unsigned.
DataLength	Any integer value	Applies to integer/short/long types. Used to distinguish the data length for the signed/unsigned types. If not set, a default of 4 is assumed.
DecimalPositions	Any integer value	Applies for zoned decimal and packed decimal types. The value represents the number of decimal positions.
PackedDec	True/false	If set to true, the attribute represents a packed decimal.
ZonedDec	True/false	If set to true, the attribute represents a zoned decimal.

Data conversion from OS/400 toolbox

The Toolbox for OS/400 has data conversion classes included. The following table matches the OS/400 data types with the corresponding IBM WebSphere Business Integration datatypes, along with the data conversion class used.

Table 7. Conversion datatypes and classes

OS/400 data type	WebSphere Business Integration Express datatype	Data conversion class
Signed two byte OS/400 number.	Integer - The app specific info - Signed=true; DataLength=2	OS400Bin2
Signed four byte OS/400 number.	Integer - The app specific info - Signed=true; DataLength=4	OS400Bin4

Table 7. Conversion datatypes and classes (continued)

OS/400 data type	WebSphere Business Integration Express datatype	Data conversion class
Signed two byte OS/400 floating point number.	Float	OS400Float4
Signed four byte OS/400 floating point number.	Double	OS400Float8
Unsigned two byte OS/400 number.	Integer - the app specific info Signed=false; DataLength=2	OS400UnsignedBin2
Unsigned four byte OS/400 number.	Integer - the app specific info Signed=false; DataLength=4	OS400UnsignedBin4
Packed-Decimal OS/400 number.	String - the MaxLength attribute property needs to have the number of digits. App specific info - DecimalPositions=<number of decimal positions>; PackedDec=true	OS400PackedDecimal
Zoned-Decimal OS/400 number.	String - the MaxLength attribute property needs to have the number of digits. App specific info - DecimalPositions=<number of decimal positions>; ZonedDec=true	OS400ZonedDecimal
Character data	String - MaxLength has the maximun length for the character data.	OS400Text
Date data	String - MaxLength has the maximun length for the date data.	OS400Text

Appendix A. Standard configuration properties for connectors

This appendix describes the standard configuration properties for the connector component of the adapters in WebSphere Business Integration Server Express, running on WebSphere InterChange Server Express.

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

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

Configuring standard connector properties

Adapter connectors have two types of configuration properties:
- Standard configuration properties
- Connector-specific configuration properties

This section describes the standard configuration properties. For information on configuration properties specific to a connector, see its adapter user guide.

Using Connector Configurator Express

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

Setting and updating property values

The default length of a property field is 255 characters.

The connector uses the following order to determine a property's value (where the highest number overrides other values):
1. Default
2. Repository
3. Local configuration file
4. Command line

A connector obtains its configuration values at startup. If you change the value of one or more connector properties during a run-time session, the property's **Update Method** determines how the change takes effect. There are four different update methods for standard connector properties:
- **Dynamic**
  The change takes effect immediately after it is saved in System Manager.
- **Component restart**
  The change takes effect only after the connector is stopped and then restarted in System Manager. You do not need to stop and restart the application-specific component or the integration broker.

21

- **Server restart**
  The change takes effect only after you stop and restart the application-specific component and the integration broker.
- **Agent restart**
  The change takes effect only after you stop and restart the application-specific component.

To determine how a specific property is updated, refer to the **Update Method** column in the Connector Configurator Express window, or see the Update Method column in the Property Summary table below.

Summary of standard properties

Table 8 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 8. Summary of standard configuration properties

Property name	Possible values	Default value	Update method	Notes
AdminInQueue	Valid JMS queue name	`CONNECTORNAME /ADMININQUEUE`	Component restart	Delivery Transport is JMS
AdminOutQueue	Valid JMS queue name	`CONNECTORNAME/ADMINOUTQUEUE`	Component restart	Delivery Transport is JMS
AgentConnections	1-4	1	Component restart	Delivery Transport is IDL
AgentTraceLevel	0-5	0	Dynamic	
ApplicationName	Application name	Value specified for the connector application name	Component restart	
BrokerType	ICS	ICS		
CharacterEncoding	`ascii7, ascii8, SJIS, Cp949, GBK, Big5, Cp297, Cp273, Cp280, Cp284, Cp037, Cp437` **Note:** This is a subset of supported values.	`ascii7`	Component restart	
ConcurrentEventTriggeredFlows	1 to 32,767	1	Component restart	Repository directory is <REMOTE>
ContainerManagedEvents	`No value or JMS`	`No value`	Component restart	Delivery Transport is JMS
ControllerStoreAndForwardMode	`true or false`	`truetrue`	Dynamic	Repository directory is <REMOTE>

Table 8. Summary of standard configuration properties (continued)

Property name	Possible values	Default value	Update method	Notes
ControllerTraceLevel	0-5	0	Dynamic	Repository directory is <REMOTE>
DeliveryQueue		*CONNECTORNAME*/DELIVERYQUEUE	Component restart	JMS transport only
DeliveryTransport	IDL or JMS	IDL	Component restart	
DuplicateEventElimination	true or false	false	Component restart	JMS transport only: Container Managed Events must be <NONE>
EnableOidForFlowMonitoring	true or false	false	Component restart	
FaultQueue		*CONNECTORNAME*/FAULTQUEUE	Component restart	JMS transport only
jms.FactoryClassName	CxCommon.Messaging.jms .IBMMQSeriesFactory or any Java class name	CxCommon.Messaging. jms.IBMMQSeriesFactory	Component restart	JMS transport only
jms.MessageBrokerName	crossworlds.queue. manager	crossworlds.queue.manager	Component restart	JMS transport only
jms.NumConcurrentRequests	Positive integer	10	Component restart	JMS transport only
jms.Password	Any valid password		Component restart	JMS transport only
jms.UserName	Any valid name		Component restart	JMS transport only
JvmMaxHeapSize	Heap size in megabytes	128m	Component restart	Repository directory is <REMOTE>
JvmMaxNativeStackSize	Size of stack in kilobytes	128k	Component restart	Repository directory is <REMOTE>
JvmMinHeapSize	Heap size in megabytes	1m	Component restart	Repository directory is <REMOTE>
Locale	en_US, ja_JP, ko_KR, zh_CN, zh_TW, fr_FR, de_DE, it_IT, es_ES, pt_BR **Note:** This is a subset of the supported locales.	en_US	Component restart	
LogAtInterchangeEnd	true or false	false	Component restart	
MaxEventCapacity	1-2147483647	2147483647	Dynamic	Repository Directory is <REMOTE>
MessageFileName	Path or filename	InterchangeSystem.txt	Component restart	

Table 8. Summary of standard configuration properties (continued)

Property name	Possible values	Default value	Update method	Notes
MonitorQueue	Any valid queue name	`CONNECTORNAME/MONITORQUEUE`	Component restart	JMS transport only: DuplicateEvent Elimination must be true
OADAutoRestartAgent	`true` or `false`	`false`	Dynamic	Repository Directory is <REMOTE>
OADMaxNumRetry	A positive number	`1000`	Dynamic	Repository Directory is <REMOTE>
OADRetryTimeInterval	A positive number in minutes	`10`	Dynamic	Repository Directory is <REMOTE>
PollEndTime	`HH:MM` (HH is 0-23, MM is 0-59)	`HH:MM`	Component restart	
PollFrequency	A positive integer in milliseconds no (to disable polling) key (to poll only when the letter p is entered in the connector's Command Prompt window)	`10000`	Dynamic	
PollQuantity	`1-500`	`1`	Agent restart	JMS transport only: Container Managed Events is specified
PollStartTime	`HH:MM`(HH is 0-23, MM is 0-59)	`HH:MM`	Component restart	
RequestQueue	Location of metadata repository		Agent restart	Set to <REMOTE>
RequestQueue	Valid JMS queue name	`CONNECTORNAME/REQUESTQUEUE`	Component restart	Delivery Transport is JMS
ResponseQueue	Valid JMS queue name	`CONNECTORNAME/RESPONSEQUEUE`	Component restart	Delivery Transport is JMS:
RestartRetryCount	`0-99`	`3`	Dynamic	
RestartRetryCount	A sensible positive value in minutes: 1 - 2147483547	`1`	Dynamic	
SourceQueue	Valid JMS queue name	`CONNECTORNAME/SOURCEQUEUE`	Agent restart	Only if Delivery Transport is JMS and Container Managed Events is specified
SynchronousRequestQueue	Valid JMS queue name	`CONNECTORNAME/ SYNCHRONOUSREQUESTQUEUE`	Component restart	Delivery Transport is JMS

Table 8. Summary of standard configuration properties (continued)

Property name	Possible values	Default value	Update method	Notes
SynchronousRequestTimeout	0 - any number (millisecs)	0	Component restart	Delivery Transport is JMS
SynchronousResponseQueue	Valid JMS queue name	*CONNECTORNAME*/ SYNCHRONOUSRESPONSEQUEUE	Component restart	Delivery Transport is JMS
WireFormat	CwBO	CwBO	Agent restart	

Standard configuration properties

This section lists and defines each of the standard connector configuration properties.

AdminInQueue

The queue that is used by the integration broker to send administrative messages to the connector.

The default value is CONNECTORNAME/ADMININQUEUE.

AdminOutQueue

The queue that is used by the connector to send administrative messages to the integration broker.

The default value is CONNECTORNAME/ADMINOUTQUEUE.

AgentConnections

The AgentConnections property controls the number of ORB connections opened by orb.init[].

By default, the value of this property is set to 1. There is no need to change this default.

AgentTraceLevel

Level of trace messages for the application-specific component. The default is 0. The connector delivers all trace messages applicable at the tracing level set or lower.

ApplicationName

Name that uniquely identifies the connector's application. This name is used by the system administrator to monitor the WebSphere business integration system environment. This property must have a value before you can run the connector.

BrokerType

Identifies the integration broker that you are using, which is ICS.

CharacterEncoding

Specifies the character code set used to map from a character (such as a letter of the alphabet, a numeric representation, or a punctuation mark) to a numeric value.

Note: Java-based connectors do not use this property. A C++ connector currently uses the value `ascii7` for this property.

By default, a subset of supported character encodings only is displayed in the drop list. To add other supported values to the drop list, you must manually modify the `\Data\Std\stdConnProps.xml` file in the product directory. For more information, see the appendix on using Connector Configurator Express in this guide.

ConcurrentEventTriggeredFlows

Determines how many business objects can be concurrently processed by the connector for event delivery. Set the value of this attribute to the number of business objects you want concurrently mapped and delivered. For example, set the value of this property to 5 to cause five business objects to be concurrently processed. The default value is 1.

Setting this property to a value greater than 1 allows a connector for a source application to map multiple event business objects at the same time and deliver them to multiple collaboration instances simultaneously. This speeds delivery of business objects to the integration broker, particularly if the business objects use complex maps. Increasing the arrival rate of business objects to collaborations can improve overall performance in the system.

To implement concurrent processing for an entire flow (from a source application to a destination application), you must:
- Configure the collaboration to use multiple threads by setting its `Maximum number of concurrent events` property high enough to use multiple threads.
- Ensure that the destination application's application-specific component can process requests concurrently. That is, it must be multi-threaded, or be able to use connector agent parallelism and be configured for multiple processes. Set the `Parallel Process Degree` configuration property to a value greater than 1.

The `ConcurrentEventTriggeredFlows` property has no effect on connector polling, which is single-threaded and performed serially.

ContainerManagedEvents

This property allows a JMS-enabled connector with a JMS event store to provide guaranteed event delivery, in which an event is removed from the source queue and placed on the destination queue as a single JMS transaction.

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

The default value is `No value`.

When ContainerManagedEvents is set to `JMS`, you must configure the following properties to enable guaranteed event delivery:
- PollQuantity = 1 to 500
- SourceQueue = CONNECTORNAME/SOURCEQUEUE

You must also configure a data handler with the MimeType, DHClass, and DataHandlerConfigMOName (optional) properties. To set those values, use the **Data Handler** tab in Connector Configurator Express. The fields for the values under the Data Handler tab will be displayed only if you have set `ContainerManagedEvents` to JMS.

Note: When ContainerManagedEvents is set to JMS, the connector does *not* call its pollForEvents() method, thereby disabling that method's functionality.

ControllerStoreAndForwardMode

Sets the behavior of the connector controller after it detects that the destination application-specific component is unavailable.

If this property is set to true and the destination application-specific component is unavailable when an event reaches 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

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 WebSphere InterChange Server Express.

The default value is CONNECTORNAME/DELIVERYQUEUE.

DeliveryTransport

Specifies the transport mechanism for the delivery of events. Possible values are IDL for CORBA IIOP or JMS for Java Messaging Service. The default is IDL.

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

JMS

Enables communication between the connector and client connector framework using Java Messaging Service (JMS).

If you select JMS as the delivery transport, additional JMS properties such as jms.MessageBrokerName, jms.FactoryClassName, jms.Password, and jms.UserName, appear in Connector Configurator Express. The first two of these properties are required for this transport.

Important: There may be a memory limitation if you use the JMS transport mechanism for a connector running on WebSphere InterChange Server Express.

In this environment, you may experience difficulty starting both the connector controller (on the server side) and the connector (on the client side) due to memory use within the WebSphere MQ client.

DuplicateEventElimination

When you set this property to `true`, a JMS-enabled connector can ensure that duplicate events are not delivered to the delivery queue. To use this feature, the connector must have a unique event identifier set as the business object's **ObjectEventId** attribute in the application-specific code. This is done during connector development.

This property can also be set to `false`.

Note: When `DuplicateEventElimination` is set to `true`, you must also configure the `MonitorQueue` property to enable guaranteed event delivery.

EnableOidForFlowMonitoring

When you set this property to `true`, the adapter framework will mark the incoming **ObjectEventId** as a foreign key for the purpose of flow monitoring.

The default is `false`.

FaultQueue

If the connector experiences an error while processing a message then the connector moves the message to the queue specified in this property, along with a status indicator and a description of the problem.

The default value is `CONNECTORNAME/FAULTQUEUE`.

JvmMaxHeapSize

The maximum heap size for the agent (in megabytes).

The default value is 128m.

JvmMaxNativeStackSize

The maximum native stack size for the agent (in kilobytes).

The default value is 128k.

JvmMinHeapSize

The minimum heap size for the agent (in megabytes).

The default value is 1m.

jms.FactoryClassName

Specifies the class name to instantiate for a JMS provider. You *must* set this connector property when you choose JMS as your delivery transport mechanism (DeliveryTransport).

The default is `CxCommon.Messaging.jms.IBMMQSeriesFactory`.

jms.MessageBrokerName

Specifies the broker name to use for the JMS provider. You *must* set this connector property when you choose JMS as your delivery transport mechanism (see DeliveryTransport).

The default is `crossworlds.queue.manager`.

jms.NumConcurrentRequests

Specifies the maximum number of concurrent service call requests that can be sent to a connector at the same time. Once that maximum is reached, new service calls block and wait for another request to complete before proceeding.

The default value is 10.

jms.Password

Specifies the password for the JMS provider. A value for this property is optional.

There is no default.

jms.UserName

Specifies the user name for the JMS provider. A value for this property is optional.

There is no default.

Locale

Specifies the language code, country or territory, and, optionally, the associated character code set. The value of this property determines such cultural conventions as collation and sort order of data, date and time formats, and the symbols used in monetary specifications.

A locale name has the following format:
`ll_TT.codeset`

where:

`ll`	a two-character language code (usually in lower case)
`TT`	a two-letter country or territory code (usually in upper case)
`codeset`	the name of the associated character code set; this portion of the name is often optional.

By default, only a subset of supported locales appears in the drop list. To add other supported values to the drop list, you must manually modify the `\Data\Std\stdConnProps.xml` file in the product directory. For more information, see the appendix on using Connector Configurator Express in this guide.

The default value is en_US. If the connector has not been globalized, the only valid value for this property is en_US. 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

Specifies whether to log errors to the integration broker's log destination. Logging to the broker's log destination also turns on e-mail notification, which generates e-mail messages for the MESSAGE_RECIPIENT specified in the InterchangeSystem.cfg file when errors or fatal errors occur.

For example, when a connector loses its connection to its application, if LogAtInterChangeEnd is set to true, an e-mail message is sent to the specified message recipient. The default is false.

MaxEventCapacity

The maximum number of events in the controller buffer. This property is used by flow control.

The value can be a positive integer between 1 and 2147483647. The default value is 2147483647.

MessageFileName

The name of the connector message file. The standard location for the message file is \connectors\messages. Specify the message filename in an absolute path if the message file is not located in the standard location.

If a connector message file does not exist, the connector uses InterchangeSystem.txt as the message file. This file is located in the product directory.

Note: To determine whether a specific connector has its own message file, see the individual adapter user guide.

MonitorQueue

The logical queue that the connector uses to monitor duplicate events. It is used only if the DeliveryTransport property value is JMS and DuplicateEventElimination is set to TRUE.

The default value is CONNECTORNAME/MONITORQUEUE

OADAutoRestartAgent

Specifies whether the connector uses the automatic and remote restart feature. This feature uses the MQ-triggered Object Activation Daemon (OAD) to restart the connector after an abnormal shutdown, or to start a remote connector from System Monitor.

This property must be set to true to enable the automatic and remote restart feature. For information on how to configure the MQ-triggered OAD feature, see the *Installation Guide for Windows*.

The default value is false.

OADMaxNumRetry

Specifies the maximum number of times that the MQ-triggered OAD automatically attempts to restart the connector after an abnormal shutdown. The OADAutoRestartAgent property must be set to true for this property to take effect.

The default value is 1000.

OADRetryTimeInterval

Specifies the number of minutes in the retry-time interval for the MQ-triggered OAD. If the connector agent does not restart within this retry-time interval, the connector controller asks the OAD to restart the connector agent again. The OAD repeats this retry process as many times as specified by the `OADMaxNumRetry` property. The `OADAutoRestartAgent` property must be set to `true` for this property to take effect.

The default is 10.

PollEndTime

Time to stop polling the event queue. The format is `HH:MM`, where *HH* represents 0-23 hours, and *MM* represents 0-59 seconds.

You must provide a valid value for this property. The default value is `HH:MM`, but must be changed.

PollFrequency

The amount of time between polling actions. Set `PollFrequency` to one of the following values:
- The number of milliseconds between polling actions.
- The word `key`, which causes the connector to poll only when you type the letter `p` in the connector's Command Prompt window. Enter the word in lowercase.
- The word `no`, which causes the connector not to poll. Enter the word in lowercase.

The default is `10000`.

Important: Some connectors have restrictions on the use of this property. To determine whether a specific connector does, see the installing and configuring chapter of its adapter guide.

PollQuantity

Designates the number of items from the application that the connector should poll for. If the adapter has a connector-specific property for setting the poll quantity, the value set in the connector-specific property will override the standard property value.

PollStartTime

The time to start polling the event queue. The format is *HH:MM*, where *HH* represents 0-23 hours, and *MM* represents 0-59 seconds.

You must provide a valid value for this property. The default value is `HH:MM`, but must be changed.

RequestQueue

The queue that is used by WebSphere InterChange Server Express to send business objects to the connector.

The default value is `CONNECTOR/REQUESTQUEUE`.

RepositoryDirectory

The location of the repository from which the connector reads the XML schema documents that store the meta-data for business object definitions.

This value must be set to <REMOTE> because the connector obtains this information from the InterChange Server Express repository.

ResponseQueue

Applicable only if `DeliveryTransport` is JMS.

Designates the JMS response queue, which delivers a response message from the connector framework to the integration broker. WebSphere InterChange Server Express sends the request and waits for a response message in the JMS response queue.

RestartRetryCount

Specifies the number of times the connector attempts to restart itself. When used for a parallel connector, specifies the number of times the master connector application-specific component attempts to restart the slave connector application-specific component.

The default is 3.

RestartRetryInterval

Specifies the interval in minutes at which the connector attempts to restart itself. When used for a parallel connector, specifies the interval at which the master connector application-specific component attempts to restart the slave connector application-specific component. Possible values ranges from 1 to 2147483647.

The default is 1.

SourceQueue

Applicable only if `DeliveryTransport` is JMS and ContainerManagedEvents is specified.

Designates the JMS source queue for the connector framework in support of guaranteed event delivery for JMS-enabled connectors that use a JMS event store. For further information, see "ContainerManagedEvents" on page 26

The default value is `CONNECTOR/SOURCEQUEUE`.

SynchronousRequestQueue

Applicable only if `DeliveryTransport` is JMS.

Delivers request messages that require a synchronous response from the connector framework to the broker. This queue is necessary only if the connector uses synchronous execution. With synchronous execution, the connector framework sends a message to the `SynchronousRequestQueue` and waits for a response back from the broker on the `SynchronousResponseQueue`. The response message sent to the connector bears a correlation ID that matches the ID of the original message.

The default is `CONNECTORNAME/SYNCHRONOUSREQUESTQUEUE`

SynchronousResponseQueue

Applicable only if `DeliveryTransport` is JMS.

Delivers response messages sent in reply to a synchronous request from the broker to the connector framework. This queue is necessary only if the connector uses synchronous execution.

The default is `CONNECTORNAME/SYNCHRONOUSRESPONSEQUEUE`

SynchronousRequestTimeout

Applicable only if `DeliveryTransport` is JMS.

Specifies the time in minutes that the connector waits for a response to a synchronous request. If the response is not received within the specified time, then the connector moves the original synchronous request message into the fault queue along with an error message.

The default value is 0.

WireFormat

This is the message format on the transport. The setting is `CwBO`.

Appendix B. Connector Configurator Express

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

The topics covered in this appendix are:

Overview of Connector Configurator Express

Connector Configurator Express allows you to configure the connector component of your adapter for use with WebSphere InterChange Server Express.

You use Connector Configurator Express to:

- Create a **connector-specific property template** for configuring your connector.
- Create a **connector configuration file**; you must create one configuration file for each connector you install.
- Set properties in a configuration file.
  You may need to modify the default values that are set for properties in the connector templates. You must also designate supported business object definitions and maps for use with collaborations as well as specify messaging, logging and tracing, and data handler parameters, as required.

Connector configuration properties include both standard configuration properties (the properties that all connectors have) and connector-specific properties (properties that are needed by the connector for a specific application or technology).

Because **standard properties** are used by all connectors, you do not need to define those properties from scratch; Connector Configurator Express incorporates them into your configuration file as soon as you create the file. However, you do need to set the value of each standard property in Connector Configurator Express.

The range of standard properties may not be the same for all brokers and all configurations. Some properties are available only if other properties are given a specific value. The Standard Properties window in Connector Configurator Express will show the properties available for your particular configuration.

For **connector-specific properties**, however, you need first to define the properties and then set their values. You do this by creating a connector-specific property template for your particular adapter. There may already be a template set up in your system, in which case, you simply use that. If not, follow the steps in "Creating a new template" on page 37 to set up a new one.

> **Note:** Connector Configurator Express runs only in a Windows environment. If you are running the connector in another environment, use Connector Configurator Express in Windows to modify the configuration file and then copy the file to the other environment.

Starting Connector Configurator Express

You can start and run Connector Configurator Express in either of two modes:

- Independently, in stand-alone mode
- From System Manager

Running Configurator Express in stand-alone mode

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

To do so:

- From **Start>Programs**, click **IBM WebSphere Business Integration Server Express> Toolset Express>Development>Connector Configurator Express**.
- Select **File>New>Configuration File**.

You may choose to run Connector Configurator Express independently to generate the file, and then connect to System Manager to save it in a System Manager project (see "Completing a configuration file" on page 41.)

Running Configurator Express from System Manager

You can run Connector Configurator Express from System Manager.

To run Connector Configurator Express:

1. Open the System Manager.
2. In the System Manager window, expand the **Integration Component Libraries** icon and highlight **Connectors**.
3. From the System Manager menu bar, click **Tools>Connector Configurator Express**. The Connector Configurator Express window opens and displays a **New Connector** dialog box.

To edit an existing configuration file:

1. In the System Manager window, select any of the configuration files listed in the Connector folder and right-click on it.
2. Click the Standard Properties tab to see which properties are included in this configuration file.

Creating a connector-specific property template

To create a configuration file for your connector, you need a connector-specific property template as well as the system-supplied standard properties.

You can create a brand-new template for the connector-specific properties of your connector, or you can use an existing file as the template.

- To create a new template, see "Creating a new template" on page 37.
- To use an existing file, simply modify an existing template and save it under the new name.

Creating a new template

This section describes how you create properties in the template, define general characteristics and values for those properties, and specify any dependencies between the properties. Then you save the template and use it as the base for creating a new connector configuration file.

To create a template:

1. Click **File>New>Connector-Specific Property Template**.
2. The **Connector-Specific Property Template** dialog box appears, with the following fields:

   • **Template**, and **Name**

   Enter a unique name that identifies the connector, or type of connector, for which this template will be used. You will see this name again when you open the dialog box for creating a new configuration file from a template.

   • **Old Template**, and **Select the Existing Template to Modify**

   The names of all currently available templates are displayed in the Template Name display.

   • To see the connector-specific property definitions in any template, select that template's name in the **Template Name** display. A list of the property definitions contained in that template will appear in the **Template Preview** display. You can use an existing template whose property definitions are similar to those required by your connector as a starting point for your template.

3. Select a template from the **Template Name** display, enter that template name in the **Find Name** field (or highlight your selection in **Template Name**), and click **Next**.

If you do not see any template that displays the connector-specific properties used by your connector, you will need to create one.

Specifying general characteristics

When you click **Next** to select a template, the **Properties - Connector-Specific Property Template** dialog box appears. The dialog box has tabs for General characteristics of the defined properties and for Value restrictions. The General display has the following fields:

• **General:**
  Property Type
  Updated Method
  Description

• **Flags**
  Standard flags

• **Custom Flag**
  Flag

After you have made selections for the general characteristics of the property, click the **Value** tab.

Specifying values

The **Value** tab enables you to set the maximum length, the maximum multiple values, a default value, or a value range for the property. It also allows editable values. To do so:

1. Click the **Value** tab. The display panel for Value replaces the display panel for General.

2. Select the name of the property in the **Edit properties** display.

3. In the fields for **Max Length** and **Max Multiple Values**, make any changes. The changes will not be accepted unless you also open the **Property Value** dialog box for the property, described in the next step.

4. Right-click the box in the top left-hand corner of the value table and click **Add**. A **Property Value** dialog box appears. Depending on the property type, the dialog box allows you to enter either a value, or both a value and range. Enter the appropriate value or range, and click **OK**.

5. The **Value** panel refreshes to display any changes you made in **Max Length** and **Max Multiple Values**. It displays a table with three columns:

   The **Value** column shows the value that you entered in the **Property Value** dialog box, and any previous values that you created.

   The **Default Value** column allows you to designate any of the values as the default.

   The **Value Range** shows the range that you entered in the **Property Value** dialog box.

   After a value has been created and appears in the grid, it can be edited from within the table display. To make a change in an existing value in the table, select an entire row by clicking on the row number. Then right-click in the **Value** field and click **Edit Value**.

Setting dependencies

When you have made your changes to the **General** and **Value** tabs, click **Next**. The **Dependencies - Connector-Specific Property Template** dialog box appears.

A dependent property is a property that is included in the template and used in the configuration file *only if* the value of another property meets a specific condition. For example, `PollQuantity` appears in the template only if JMS is the transport mechanism and `DuplicateEventElimination` is set to `True`.
To designate a property as dependent and to set the condition upon which it depends, do this:

1. In the **Available Properties** display, select the property that will be made dependent.

2. In the **Select Property** field, use the drop-down menu to select the property that will hold the conditional value.

3. In the **Condition Operator** field, select one of the following:

   == (equal to)

   != (not equal to)

   > (greater than)

   < (less than)

   >= (greater than or equal to)

   <=(less than or equal to)

4. In the **Conditional Value** field, enter the value that is required in order for the dependent property to be included in the template.

5. With the dependent property highlighted in the **Available Properties** display, click an arrow to move it to the **Dependent Property** display.

6. Click **Finish**. Connector Configurator Express stores the information you have entered as an XML document, under \data\app in the\bin directory where you have installed Connector Configurator Express.

Creating a new configuration file

You create a connector configuration file from a connector-specific template or by modifying an existing configuration file.

Creating a configuration file from a connector-specific template

Once a connector-specific template has been created, you can use it to create a configuration file:

1. Click **File>New>Connector Configuration**.
2. The **New Connector** dialog box appears, with the following fields:
    * **Name**

      Enter the name of the connector. Names are case-sensitive. The name you enter must be unique, and must be consistent with the file name for a connector that is installed on the system.

      Important: Connector Configurator Express does not check the spelling of the name that you enter. You must ensure that the name is correct.
    * **System Connectivity**

      The default broker is ICS. You cannot change this value.
    * **Select Connector-Specific Property Template**

      Type the name of the template that has been designed for your connector. The available templates are shown in the **Template Name** display. When you select a name in the Template Name display, the **Property Template Preview** display shows the connector-specific properties that have been defined in that template.

      Select the template you want to use and click **OK**.
3. A configuration screen appears for the connector that you are configuring. The title bar shows the integration broker and connector names. You can fill in all the field values to complete the definition now, or you can save the file and complete the fields later.
4. To save the file, click **File>Save>To File** or **File>Save>To Project**. To save to a project, System Manager must be running.
   If you save as a file, the **Save File Connector** dialog box appears. Choose *.cfg as the file type, verify in the File Name field that the name is spelled correctly and has the correct case, navigate to the directory where you want to locate the file, and click **Save**. The status display in the message panel of Connector Configurator Express indicates that the configuration file was successfully created.

      Important: The directory path and name that you establish here must match the connector configuration file path and name that you supply in the startup file for the connector.
5. To complete the connector definition, enter values in the fields for each of the tabs of the Connector Configurator Express window, as described later in this chapter.

Using an existing file

To use an existing file to configure a connector, you must open the file in
Connector Configurator Express, revise the configuration, and then save the file as
a configuration file (*.cfg).

You may have an existing file available in one or more of the following formats:

- A connector definition file.
  This is a text file that lists properties and applicable default values for a specific
  connector. Some connectors include such a file in a \repository directory in
  their delivery package (the file typically has the extension .txt; for example,
  CN_XML.txt for the XML connector).

- An InterChange Server Express repository file.
  Definitions used in a previous InterChange Server Express implementation of
  the connector may be available to you in a repository file that was used in the
  configuration of that connector. Such a file typically has the extension .in or
  .out.

- A previous configuration file for the connector.
  Such a file typically has the extension *.cfg.

Although any of these file sources may contain most or all of the connector-specific
properties for your connector, the connector configuration file will not be complete
until you have opened the file and set properties, as described later in this chapter.

To use an existing file to configure a connector, you must open the file in
Connector Configurator Express, revise the configuration, and then resave the file.

Follow these steps to open a *.txt, *.cfg or *.in file from a directory:

1. In Connector Configurator Express, click **File>Open>From File**.
2. In the **Open File Connector** dialog box, select one of the following file types to
   see the available files:
   - Configuration (*.cfg)
   - InterChange Server Express Repository (*.in, *.out)

     Choose this option if a repository file was used to configure the connector. A
     repository file may include multiple connector definitions, all of which will
     appear when you open the file.
   - All files (*.*)

     Choose this option if a *.txt file was delivered in the adapter package for
     the connector, or if a definition file is available under another extension.
3. In the directory display, navigate to the appropriate connector definition file,
   select it, and click **Open**.

Follow these steps to open a connector configuration from a System Manager
project:

1. Start System Manager. A configuration can be opened from or saved to System
   Manager only if System Manager has been started.
2. Start Connector Configurator Express.
3. Click **File>Open>From Project**.

Completing a configuration file

When you open a configuration file or a connector from a project, the Connector Configurator Express window displays the configuration screen, with the current attributes and values.

Connector Configurator Express requires values for properties described in the following sections:

- "Setting standard connector properties"
- "Setting application-specific configuration properties" on page 42
- "Specifying supported business object definitions" on page 43
- "Associated maps" on page 44
- "Setting trace/log file values" on page 45

Note: For connectors that use JMS messaging, an additional category may display, for special configuration of data handlers that convert the data to business objects. For further information, see "Data handlers" on page 46.

Setting the configuration file properties

When you create and name a new connector configuration file, or when you open an existing connector configuration file, Connector Configurator Express displays a configuration screen with tabs for the categories of required configuration values.

Standard properties differ from connector-specific properties as follows:

- Standard properties of a connector are shared by both the application-specific component of a connector and its broker component. All connectors have the same set of standard properties. These properties are described in Appendix A of each adapter guide. You can change some but not all of these values.
- Application-specific properties apply only to the application-specific component of a connector, that is, the component that interacts directly with the application. Each connector has application-specific properties that are unique to its application. Some of these properties provide default values and some do not; you can modify some of the default values. The installation and configuration chapters of each adapter guide describe the application-specific properties and the recommended values.

The fields for **Standard Properties** and **Connector-Specific Properties** are color-coded to show which are configurable:

- A field with a grey background indicates a standard property. You can change the value but cannot change the name or remove the property.
- A field with a white background indicates an application-specific property. These properties vary according to the specific needs of the application or connector. You can change the value and delete these properties.
- You can configure Value fields.
- The **Update Method** displayed for each property indicates whether a component or agent restart is necessary to activate changed values.

Setting standard connector properties

To change the value of a standard property:

1. Click in the field whose value you want to set.
2. Either enter a value, or select one from the drop-down menu if it appears.

3. After entering all the values for the standard properties, you can do one of the following:
   - To discard the changes, preserve the original values, and exit Connector Configurator Express, click **File>Exit** (or close the window), and click **No** when prompted to save changes.
   - To enter values for other categories in Connector Configurator Express, select the tab for the category. The values you enter for **Standard Properties** (or any other category) are retained when you move to the next category. When you close the window, you are prompted to either save or discard the values that you entered in all the categories as a whole.
   - To save the revised values, click **File>Exit** (or close the window) and click **Yes** when prompted to save changes. Alternatively, click **Save>To File** from either the File menu or the toolbar.

Setting application-specific configuration properties

For application-specific configuration properties, you can add or change property names, configure values, delete a property, and encrypt a property. The default property length is 255 characters.

1. Right-click in the top left portion of the grid. A pop-up menu bar will appear. Click **Add** to add a property. To add a child property, right-click on the parent row number and click **Add child**.
2. Enter a value for the property or child property.
3. To encrypt a property, select the **Encrypt** box.
4. Choose to save or discard changes, as described for "Setting standard connector properties" on page 41.

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 21.

Connector properties are almost all static and the **Update Method** is `Component restart`. For changes to take effect, you must restart the connector after saving the revised connector configuration file.

Specifying supported business object definitions

Use the **Supported Business Objects** tab in Connector Configurator Express to specify the business objects that the connector will use. You must specify both generic business objects and application-specific business objects, and you must specify associations for the maps between the business objects.

For you to specify a supported business object, the business objects and their maps must exist in the system. Business object definitions, including those for data handler meta-objects, and map definitions should be saved into Integration Component Library (ICL) projects. For more information on ICL projects, see the *User Guide for WebSphere Business Integration Server Express*.

Note: Some connectors require that certain business objects be specified as supported in order to perform event notification or additional configuration (using meta-objects) with their applications. For more information, see the chapter on business objects in this guide as well as the *Business Object Development Guide*.

To specify that a business object definition is supported by the connector, or to change the support settings for an existing business object definition, click the **Supported Business Objects** tab and use the following fields.

Business object name

To designate that a business object definition is supported by the connector, with System Manager running:

1. Click an empty field in the **Business Object Name** list. A drop-down list displays, showing all the business object definitions that exist in the System Manager project.
2. Click on a business object to add it.
3. Set the **Agent Support** (described below) for the business object.
4. In the File menu of the Connector Configurator Express window, click **Save to Project**. The revised connector definition, including designated support for the added business object definition, is saved to the project in System Manager.

To delete a business object from the supported list:

1. To select a business object field, click the number to the left of the business object.
2. From the **Edit** menu of the Connector Configurator Express window, click **Delete Row**. The business object is removed from the list display.
3. From the **File** menu, click **Save to Project**.

Deleting a business object from the supported list changes the connector definition and makes the deleted business object unavailable for use in this implementation of this connector. It does not affect the connector code, nor does it remove the business object definition itself from System Manager.

Agent support

If a business object has Agent Support, the system will attempt to use that business object for delivering data to an application via the connector agent.

Typically, application-specific business objects for a connector are supported by that connector's agent, but generic business objects are not.

To indicate that the business object is supported by the connector agent, check the **Agent Support** box. The Connector Configurator Express window does not validate your Agent Support selections.

Maximum transaction level

The maximum transaction level for a connector is the highest transaction level that the connector supports.

For most connectors, `Best Effort` is the only possible choice.

You must restart the server for changes in transaction level to take effect.

Associated maps

Each connector supports a list of business object definitions and their associated maps that are currently active in InterChange Server Express. This list appears when you select the **Associated Maps** tab.

The list of business objects contains the application-specific business object which the agent supports and the corresponding generic object that the controller sends to the subscribing collaboration. The association of a map determines which map will be used to transform the application-specific business object to the generic business object or the generic business object to the application-specific business object.

If you are using maps that are uniquely defined for specific source and destination business objects, the maps will already be associated with their appropriate business objects when you open the display, and you will not need (or be able) to change them.

If more than one map is available for use by a supported business object, you will need to explicitly bind the business object with the map that it should use.

The **Associated Maps** tab displays the following fields:

- **Business Object Name**

  These are the business objects supported by this connector, as designated in the **Supported Business Objects** tab. If you designate additional business objects under the Supported Business Objects tab, they will be reflected in this list after you save the changes by choosing **Save to Project** from the **File** menu of the Connector Configurator Express window.

- **Associated Maps**

  The display shows all the maps that have been installed to the system for use with the supported business objects of the connector. The source business object for each map is shown to the left of the map name, in the **Business Object Name** display.

- **Explicit**

  In some cases, you may need to explicitly bind an associated map.

  Explicit binding is required only when more than one map exists for a particular supported business object. When InterChange Server Express boots, it tries to automatically bind a map to each supported business object for each connector.

If more than one map takes as its input the same business object, the server attempts to locate and bind one map that is the superset of the others.

If there is no map that is the superset of the others, the server will not be able to bind the business object to a single map, and you will need to set the binding explicitly.

To explicitly bind a map:

1. In the **Explicit** column, place a check in the check box for the map you want to bind.
2. Select the map that you intend to associate with the business object.
3. In the **File** menu of the Connector Configurator Express window, click **Save to Project**.
4. Deploy the project to InterChange Server Express.
5. Reboot the server for the changes to take effect.

Resources

The **Resource** tab allows you to set a value that determines whether and to what extent the connector agent will handle multiple processes concurrently, using connector agent parallelism.

Not all connectors support this feature. If you are running a connector agent that was designed in Java to be multi-threaded, you are advised not to use this feature, since it is usually more efficient to use multiple threads than multiple processes.

Setting trace/log file values

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

To change the logging and tracing values:

1. Click the **Trace/Log Files** tab.
2. For either logging or tracing, you can choose to write messages to one or both of the following:
   - To console (STDOUT):
     Writes logging or tracing messages to the STDOUT display.

     Note: You can only use the STDOUT option from the **Trace/Log Files** tab for connectors running on the Windows platform.
   - To File:
     Writes logging or tracing messages to a file that you specify. To specify the file, click the directory button (ellipsis), navigate to the preferred location, provide a file name, and click **Save**. (If your connector is not running on the Windows platform on which you have installed Connector Configurator Express, you must first map a drive to a location on the system where you want the file.) Logging or tracing message are written to the file and location that you specify.

     Note: Both logging and tracing files are simple text files. You can use the file extension that you prefer when you set their file names. For tracing files, however, it is advisable to use the extension `.trace` rather than `.trc`, to avoid confusion with other files that might reside on the system. For logging files, `.log` and `.txt` are typical file extensions.

Data handlers

The data handlers section is available for configuration only if you have designated a value of JMS for DeliveryTransport and a value of JMS for ContainerManagedEvents. Adapters that make use of the guaranteed event delivery enable this tab.

See the descriptions under `ContainerManagedEvents` in the Standard Properties appendix for values to use for these properties.

Saving your configuration file

After you have created the configuration file and set its properties, you need to deploy it to the correct location for your connector. Save the configuration in an ICL project, and use System Manager to load the file into InterChange Server Express.

The file is saved as an XML document. You can save the XML document in three ways:

- From System Manager, as a file with a `*.con` extension in an Integration Component Library, or
- In a directory that you specify.
- In stand-alone mode, as a file with a `*.cfg` extension in a directory folder.

For details about using projects in System Manager, and for further information about deployment, see the *User Guide for IBM WebSphere Business Integration Server Express.*

Completing the configuration

After you have created a configuration file for a connector and modified it, make sure that the connector can locate the configuration file when the connector starts up.

To do so, open the startup file used for the connector, and verify that the location and file name used for the connector configuration file match exactly the name you have given the file and the directory or path where you have placed it.

Using Connector Configurator Express in a globalized environment

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

Connector Configurator Express supports non-English characters in:

- All value fields
- Log file and trace file path (specified in the **Trace/Log files** tab)

The drop list for the `CharacterEncoding` and `Locale` standard configuration properties displays only a subset of supported values. To add other values to the drop list, you must manually modify the `\Data\Std\stdConnProps.xml` file in the product directory.

For example, to add the locale en_GB to the list of values for the Locale property, open the stdConnProps.xml file and add the line in boldface type below:

```
<Property name="Locale"
isRequired="true"
updateMethod="component restart">
                <ValidType>String</ValidType>
          <ValidValues>
                                <Value>ja_JP</Value>
                                <Value>ko_KR</Value>
                                <Value>zh_CN</Value>
                                <Value>zh_TW</Value>
                                <Value>fr_FR</Value>
                                <Value>de_DE</Value>
                                <Value>it_IT</Value>
                                <Value>es_ES</Value>
                                <Value>pt_BR</Value>
                                <Value>en_US</Value>
                                <Value>en_GB</Value>

                <DefaultValue>en_US</DefaultValue>
          </ValidValues>
     </Property>
```

Notices

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 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.

Note: 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.

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product 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 ServerExpress V4.3.1 and WebSphere Business Integration Server Express Plus V4.3.1

IBM ®

Printed in USA