





Note

Before using this information and the product it supports, read the information in "Notices" on page 221.

December 2008

This edition applies to version 6, release 2, modification 0 of IBM WebSphere Adapter for FTP and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, email <mailto://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.

© Copyright International Business Machines Corporation 2006, 2008. All rights reserved. US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

© **Copyright International Business Machines Corporation 2006, 2008.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Chapter 1. Overview of WebSphere

Adapter for FTP	1
Hardware and software requirements	1
Technical overview of the Adapter for FTP	2
Outbound processing	2
Inbound processing	8
Business objects	17
WebSphere Application Server environment variables	18
External service wizard	19

Chapter 2. Planning for adapter implementation 21

Before you begin	21
Security	21
Configuring secure socket layers	21
Configuring support for SFTP	24
Configuring the adapter for federal information processing standard 140	24
Server verification	25
Public key authentication.	25
Support for confidential logging and tracing	26
User authentication.	27
Deployment options	27
WebSphere Adapters in clustered environments	30
Adapter customization with Customer Parser Class	32
Migrating to version 6.2	32
Migration considerations	32
Performing the migration.	34
Updating but not migrating a version 6.0.2.x or version 6.1.x project	35

Chapter 3. Samples and tutorials 37

Chapter 4. Configuring the module for deployment. 39

Roadmap for configuring the module.	39
Creating an authentication alias	41
Creating the module	42
Defining business objects	44
Converting business objects to COBOL copybook files during outbound processing	46
Converting COBOL copybook files to business objects during inbound processing.	56
Defining WebSphere Application Server environment variables	66
Creating a simple service with the adapter pattern wizard	68
Starting the external service wizard	78
Configuring the module for outbound processing	78
Setting deployment and runtime properties.	79
Selecting a data type and operation name	81
Configuring the data binding	83
Configuring data handlers	85

Setting interaction specification properties and generating the service	89
Configuring the module for inbound processing	92
Setting deployment and runtime properties.	92
Selecting a data type and operation name	103
Configuring the data binding	105
Configuring data handlers	107
Generating the service	111

Chapter 5. Changing interaction specification properties using the assembly editor 113

Chapter 6. Deploying the module 115

Deployment environments	115
Deploying the module for testing.	115
Generating and wiring a target component for testing inbound processing	115
Adding the module to the server.	117
Testing the module for outbound processing using the test client	118
Deploying the module for production	119
Installing the RAR file (for modules using stand-alone adapters only)	119
Exporting the module as an EAR file	120
Installing the EAR file	121

Chapter 7. Administering the adapter module 123

Changing configuration properties for embedded adapters	123
Setting resource adapter properties for embedded adapters	123
Setting managed (J2C) connection factory properties for embedded adapters	125
Setting activation specification properties for embedded adapters	127
Changing configuration properties for stand-alone adapters	129
Setting resource adapter properties for stand-alone adapters	129
Setting managed (J2C) connection factory properties for stand-alone adapters	130
Setting activation specification properties for stand-alone adapters	132
Starting the application that uses the adapter.	133
Stopping the application that uses the adapter	133
Monitoring performance using Performance Monitoring Infrastructure	134
Configuring Performance Monitoring Infrastructure	134
Enabling tracing with the Common Event Infrastructure (CEI)	136
Viewing performance statistics	137

Chapter 8. Troubleshooting and support 139

Support for the Log and Trace Analyzer 139
Handling Leap Year Exception 140
ServerToServerFileTransfer 140
Configuring logging and tracing 141
 Configuring logging properties 141
 Changing the log and trace file names 142
First-failure data capture (FFDC) support 143
Business faults 144
XAResourceNotAvailableException 145
org.xml.sax.SAXParseException 146
Self-help resources. 146

Chapter 9. Reference information. 149

Business object information. 149
 Business object structure. 149
 Naming conventions 152
 Business object attribute properties 153
 Business object operation support 153
 Custom business objects. 153
Custom file splitting 154

Outbound configuration properties 155
 Adapter type properties 156
 Resource adapter properties 158
 Managed (J2C) connection factory properties 162
 Wrapper and interaction specification properties 172
Inbound configuration properties. 181
 Adapter type properties 183
 Resource adapter properties 184
 Activation specification properties 189
Globalization 212
 Globalization and bidirectional transformation 212
 Bidirectional transformation in business objects 215
 Properties enabled for bidirectional data transformation 216
Adapter messages 218
Related information 218

Notices 221

Programming interface information 223
Trademarks and service marks 223

Index 225

Chapter 1. Overview of WebSphere Adapter for FTP

With WebSphere Adapter for FTP, you can create integrated processes that use WebSphere Process Server and WebSphere Enterprise Service Bus to access files managed by an FTP server without having to know the details of FTP communications or protocols.

Once configured, the adapter acts like a service provider in a Service oriented architecture (SOA) implementation, providing operations to send and retrieve files. The adapter is part of a module that is deployed to WebSphere Process Server or WebSphere Enterprise Service Bus.

The adapter exposes a service interface that hides the mechanics of how the data or operations are obtained or run. Services outside of the module interact with the adapter instead of directly with the FTP server, so authentication details (such as user name and password) that you provide when you set up a module are shielded from services outside of the module.

What is the benefit? The module, which you create with the external service wizard in WebSphere Integration Developer, is a reusable unit designed to perform a specific inbound or outbound service. Each module uses a consistent interface and standard business objects, so applications consuming the service do not have to understand the lower-level details of the FTP server.

The following illustration shows how the adapter functions as part of an SOA implementation.

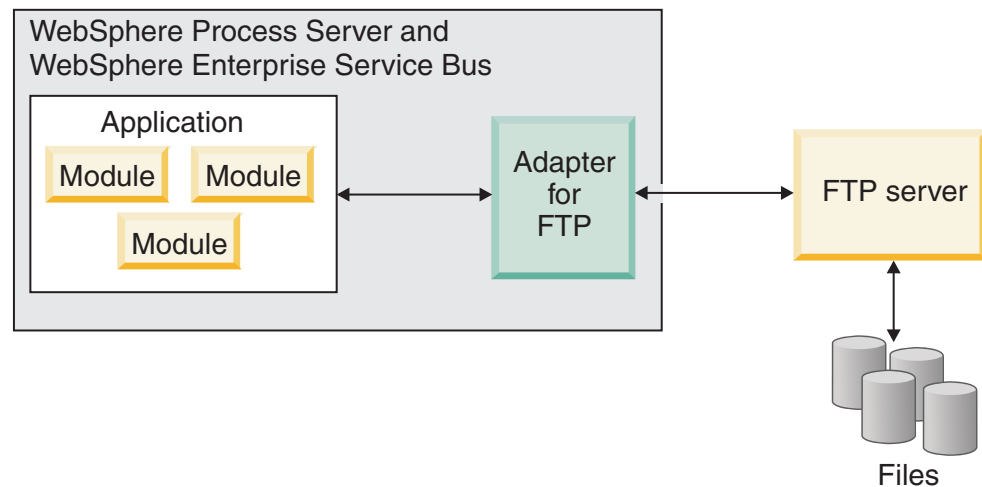


Figure 1. Adapter overview

Hardware and software requirements

The hardware and software requirements for WebSphere Adapters are provided on the IBM Support Web site.

To view hardware and software requirements for WebSphere Adapters, see <http://www.ibm.com/support/docview.wss?uid=swg27006249>

Additional information

The following links provide additional information you might need to configure and deploy your adapter:

- The compatibility matrix for WebSphere Business Integration Adapters and WebSphere Adapters identifies the supported versions of required software for your adapter. To view this document, go to the WebSphere Adapters support page and click **Compatibility Matrix** beneath the **Related** heading in the **Additional support links** section: <http://www.ibm.com/software/integration/wbiadapters/support/>.
- Technotes for WebSphere Adapters provide workarounds and additional information that are not included in the product documentation. To view the technotes for your adapter, go to the following Web page, select the name of your adapter from the **Product category** list, and click the search icon: <http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm>.

Technical overview of the Adapter for FTP

WebSphere Adapter for FTP provides the means for services running on WebSphere Integration Developer to communicate with one or more FTP servers.

The services are contained in a module, which consists of both a project in WebSphere Integration Developer and a unit of deployment to WebSphere Process Server. The module is packaged and deployed to WebSphere Process Server as an enterprise archive (EAR) file.

The module contains components, which are the actual services, imports and exports. Imports identify services outside of a module, making them callable from within the module. Exports allow components in a module to provide their services to external clients. Imports and exports require binding information, which specifies the means of transporting the data from the modules. The assembly editor in WebSphere Integration Developer sets up the imports and exports, lists the supported bindings, and simplifies their creation.

Outbound processing

The Adapter for FTP supports outbound request processing. When the adapter receives a request, which is sent in the form of a business object from the module, it processes the request and returns the result, when applicable, in a business object.

The following illustration shows the outbound processing flow for the WebSphere Adapter for FTP.

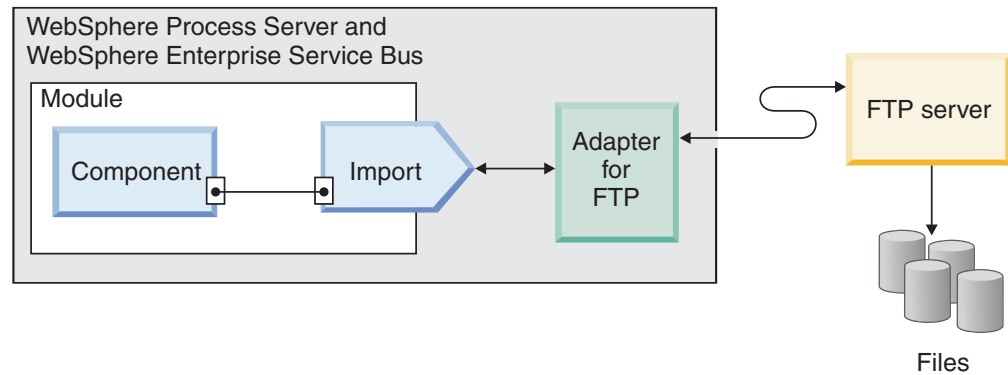


Figure 2. Outbound processing flow

Outbound data transformation

Data transformation during outbound communications refers to the process by which the adapter transforms business objects into an event record created in a native format, such as bytes or a string. The adapter uses an adapter-specific data binding and data handlers to accomplish this.

Data transformation allows external applications to send and receive data in a format that they can understand and process easily. The data bindings and data handlers that the adapter uses to create the event record from the corresponding attributes in a business object are configured using the external service wizard in WebSphere Integration Developer.

Data bindings

Data bindings are responsible for reading the fields in a business object and filling the corresponding fields in an event record. Each data binding is a map that defines how a business object should be formatted. The adapter for FTP uses the `FTPFileBaseDataBinding` data binding during outbound communication.

During outbound communications, the data binding takes the following fields from a business object and populates the equivalent fields in an event record with their values:

- `DirectoryPath`
- `Filename`
- `DataConnectionMode`
- `FileTransferType`
- `SecondServerDirectory`
- `SecondServerUsername`
- `SecondServerPassword`
- `IncludeEndBODElimiter`
- `FileInLocalDirectory`
- `LocalDirectoryPath`
- `LocalArchivingEnabledForCreate`
- `LocalArchiveDirForCreate`
- `StagingDirectory`
- `GenerateUniqueFile`

- SplittingFunctionClassName
- SplitCriteria
- DeleteOnRetrieve
- ArchiveDirectoryForRetrieve
- FileContentEncoding

For data that does not require transformation, the adapter conducts what is called pass-through processing because data passes through the system without being altered.

Data handlers

In addition to data bindings, data transformation requires the use of a data handler. Data handlers perform the conversions between a business object and a native format. In version 6.2 of WebSphere Adapter for FTP, the adapter provides the following data handlers:

- Delimited
- Fixed width
- XML

Supported operations

An operation is an action that the adapter can perform on remote file systems accessible through an FTP server during outbound processing. The name of the operation typically indicates the type of action that the adapter takes, such as *Create* or *Append*.

During outbound processing, WebSphere Adapter for FTP supports the following operations.

Table 1. Supported outbound operations

Operation	Result
Create	<p>The file with the specified name is created in the specified directory of the FTP server. The content of the file can either be sent as part of the request or it can be retrieved from the local file system.</p> <p>When the file content is received as part of the request, the adapter provides the option to archive the file on the adapter workstation before creating it.</p> <p>The file can be created in a staging directory and then sent to the actual directory. If a staging directory is not specified, the file is directly created in actual directory.</p> <p>After the file is created, the file name is sent back to the calling component to indicate that the file was created successfully. If the file to be created already exists, a DuplicateRecord exception is sent, and no file is created. The existing file is not overwritten.</p> <p>The adapter provides a feature to generate unique file names. See “Generating unique file names” on page 8.</p> <p>The adapter provides a feature to create a file sequence for the output files created. See “Generating a file sequence during Create operations” on page 7.</p>

Table 1. Supported outbound operations (continued)

Operation	Result
Append	<p>The file with the specified name in the specified directory of the FTP server is appended with the content sent in the request.</p> <p>If the file to be appended exists, the content is appended, and the file name is sent back to the calling component indicating a successful response.</p> <p>If the staging directory is specified, the file to be appended is copied from the specified output directory to the staging directory, and the content is appended to that file in the staging directory. The appended file is then moved back to the original directory.</p> <p>If the file to be appended does not exist and the CreateIfFileNotExist property is set to true, the adapter creates a new file.</p> <p>If the file to be appended does not exist, a RecordNotFound exception is sent to the calling component.</p>
Delete	<p>The file in the specified directory is deleted on the FTP server and the adapter returns true to the calling component to indicate that the file was successfully deleted.</p> <p>If the file to be deleted does not exist, a RecordNotFound exception is sent to the calling component.</p>
Retrieve	<p>The content of the file or files in the specified request is returned.</p> <p>The file content is split based on the SplittingFunctionClassName and SplitCriteria properties. The file content is transformed into a business object based on the configured data handler.</p> <p>After the content of the file is retrieved it is sent as the response. The file content can either be sent back to the calling component or saved to the local file system. If the file to be retrieved does not exist, a RecordNotFound exception is sent to the calling component.</p> <p>The adapter provides an option to delete the file from the FTP server directory after it is retrieved through the DeleteOnRetrieve property.</p> <p>The adapter supports an option to archive the file on the FTP server before it is deleted through the ArchiveDirectoryForDeleteOnRetrieve property.</p> <p>While configuring the Retrieve operation for data transformation, create custom retrieve wrappers like CustomerRetrieveWrapper, CustomerRetrieveWrapperBG, or OrderRetrieveWrapper, or OrderRetrieveWrapperBG and use the wrapper for the output type in the operation window.</p> <p>For a Retrieve operation without data transformation, the default wrapper RetrieveResponseWrapper is used.</p> <p>Note: The backward compatibility may use RetrieveResponseWrapper for retrieving XML data with datatransformation.</p>

Table 1. Supported outbound operations (continued)

Operation	Result
Overwrite	<p>Overwrites the file in the directory with the content specified in the request.</p> <p>After, the content is overwritten, the file name is sent back to the calling component indicating a successful response.</p> <p>The file to be overwritten is copied from the specified directory to the staging directory, if specified, and the content is overwritten for that file in the staging directory. The file is then moved back to the specified directory. If a staging directory is not specified, the content is overwritten on the file in the specified directory.</p> <p>If the file to be overwritten does not exist, and the CreateIfFileNotExist property is set to true, the adapter creates a new file.</p> <p>If the file to be overwritten does not exist, a RecordNotFound exception is sent to the calling component.</p>
Exists	<p>If the file name in the request exists in the specified directory or any of the sub folders, the adapter returns true and the full path of the file to the calling component. If a file with the same name exists in more than one directory, the adapter returns true and the full path of the first file found to the calling component.</p> <p>If the file name does not exist, or the directory does not exist, the adapter returns false to the calling component.</p>
List	<p>All the file names and directories that are specified in the request are returned to the calling component.</p> <p>If only the directory is specified, all the file names in the directory are retrieved and sent as a response to the calling component.</p> <p>If the specified directory does not exist, a RecordNotFound exception is sent to the calling component.</p>
ServerToServer FileTransfer	<p>The specified file is transferred from one FTP server directory to another FTP server directory. After the file has been transferred successfully, true is returned to the calling component.</p> <p>Both the FTP servers have to support the ServerToServerFileTransfer operation and a connection must be established between the FTP servers and the workstation where the adapter is running.</p> <p>If the request does not contain all necessary information about the two servers, the adapter sends an FTPFileServerToServerFileTransfer exception to the calling component. Note: The ServerToServerFileTransfer operation is not supported for the SFTP protocol.</p>
ExecuteFTPScript	<p>The commands contained in an FTP script file are run in the adapter workstation. The operation runs only the commands that are supported by the FTP server. If the operation fails, the adapter sends an FTPFileExecuteFTPScript exception to the calling component.</p> <p>The script file must not contain connection-related commands such as open because the adapter uses an established connection to run the commands.</p> <p>The location of the script file is specified in the DirectoryPath and Filename properties.</p> <p>If the commands in the script file need to be run in a particular directory on the FTP server, then the script file must first contain the command to change to that directory.</p> <p>A list of commands runs and their reply strings are returned to the calling component. The adapter also supports parameter substitution in the FTP script file (replacing parameters %1, %2 with actual values). The values are sent as part of the request. Note: The script file should contain commands that are supported by the selected protocol.</p>

Generating a file sequence during Create operations

Adapter for FTP supports the generation of a file sequence during an outbound Create operation. The FileSequenceLog property is introduced to specify the full path of the file where the sequences are stored.

A sequence file is a file used to store the sequence number. The adapter obtains the sequence number in this file for the current operation and increments the existing number by one and updates the file. When a sequence file is created, the file does not contain any data and the adapter starts generating the sequence number from 1.

For every request, the adapter reads the sequence number, increments it by 1 and then updates the sequence file. A sequence number is used while creating a request file in the target folder. If the number is not valid, for instance, if it is non-numeric, consists of special characters, or is zero or negative, the adapter starts the sequence again from 1. The adapter uses the existing sequence number in the file when the adapter is restarted.

Note: The sequence number is the only content in the sequence file that is used for an outbound create operation regardless of any directory or filename.

When a value is specified for the FileSequenceLog property, the adapter generates file sequence numbers that it appends to the file name of the files that it creates.

The sequence number takes the following format:

`$FILENAME.$SEQUENCE_NUMBER.$FILE_EXT`. For example, if HostName = localhost and Filename = Customer.txt, the output files are Customer.1.txt, Customer.2.txt, Customer.3.txt, and so on. The format is the same for all platforms, including z/OS® and i5/OS®. The sequence number continues to increment after multiple adapter restarts.

When the adapter is operating in a stand-alone mode, the value for the FileSequenceLog property must be a file on the local file system. When the adapter is operating in a clustered environment, the value for the FileSequenceLog property must be a file on the mapped drive that is accessible by all of the clusters. The adapter must have write permission for the sequence log file or an IOException takes place.

Note: The file sequence number can be reset either by deleting the entry in the file or by deleting the file. A new sequence begins at 1. When the FileSequenceLog property and GenerateUniqueFilename property are both enabled, the GenerateUniqueFilename property value takes precedence, and the FileSequenceLog property is not generated.

You can generate the file sequence names. To generate file sequence names, specify:

1. The sequence file, which is the full path of the file where the sequence numbers are stored
2. The default target file name

The adapter generates a file name that consists of the default target file name with the sequence number appended to it. If the default file name has an extension, the sequence number is appended before the extension. For example, if the default file name is Customer.txt on the managed connection factory, the output file names that are created are Customer.1.txt, Customer.2.txt, and so on.

The adapter performs the following steps to support compatibility with earlier versions:

1. The adapter reads the sequence file and checks for an entry of the form path = sequenceNumber.
2. If such an entry exists in the file, the sequence file contains the data in the form supported by IBM WebSphere Adapter for FTP version 6.2.
3. The adapter gets the highest sequence number available from all the entries.
4. This number is used to create the new file.
5. The adapter increments the number and overwrites the entire file with just the new number.

Note: Two different managed connection factories should not access the same sequence file. Also, two different adapter instances should not access the same sequence file unless they are part of a cluster, in which case they access a shared sequence file.

Generating unique file names

The Create operation supports the generation of unique file names under the following conditions:

- For the Create operation when the GenerateUniqueFile property is set to true.

The properties that control the generation of unique file names are located in three places:

- The managed connection factory properties (the Default target file name and Sequence file properties)
- The interaction specification properties (the Default target file name and Generate a unique file properties)
- The wrapper business object

The properties in the business object take precedence over the properties in the interaction specification, which take precedence over the managed connection factory properties. Unless you want to handle a particular object differently, use the properties on the managed connection factory to control the generation of file names.

If the FTP server supports the STOU command specified in RFC 1123, the adapter uses this server support to generate the unique file names.

If the FTP server does not support the STOU command, the adapter for FTP generates a unique file and creates it on the FTP servers. The format of the file created by the adapter is F followed by the combination of TP and random numbers. The number ranges between 0 and 99999. The following examples illustrate this format: FTP0, FTP9, FTP729, FTP99999

Note: The adapter does not support both the GenerateUniqueFile and StagingDirectory options simultaneously.

Inbound processing

The Adapter for FTP supports inbound processing of events. The adapter polls a file system associated with an FTP server for events at specified intervals. Each time a file is created or updated, the adapter tracks this as an event. When the adapter detects an event, it requests a copy of the file, converts the file data into a business object, and sends it to the consuming service.

The following illustration shows the inbound processing flow for WebSphere Adapter for FTP.

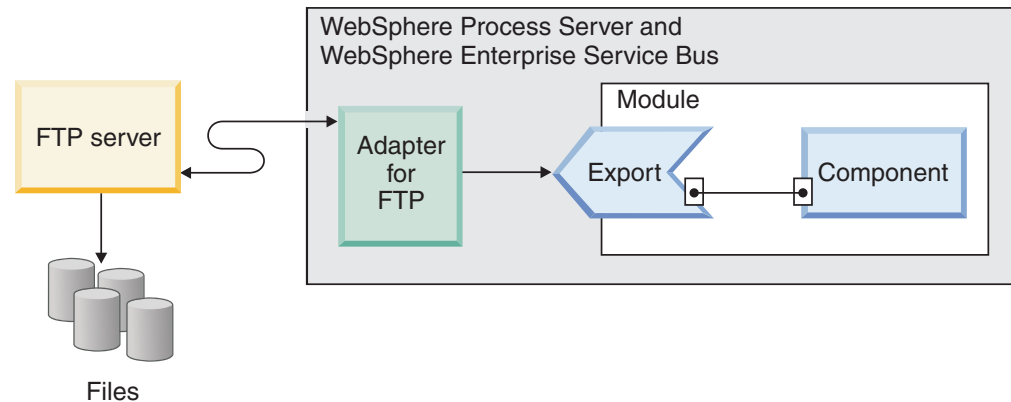


Figure 3. Inbound processing flow

The adapter polls files from the event directory of the FTP server at regular intervals based on the `FTPPollFrequency` property. When a file arrives in the event directory, the adapter reads the entire file and downloads the file to a local event directory on the adapter workstation. After the file is downloaded, the adapter either archives the file in the FTP server in an archive directory given by the `FTPArchiveDirectory` property or deletes it based on your configuration. The event directory, archive directory, poll frequency, and poll quantity (the number of files to poll in a single poll cycle) are all configurable properties.

After the business objects are successfully posted to the export, the events in the local staging directory are either archived in an archive directory on the local file system or deleted, based on your configuration. The adapter must archive or delete the events or they will be polled again.

Inbound event processing consists of the following steps:

1. FTP server generates events in the form of files.
2. The Adapter for FTP polls the event directory.
3. The files are downloaded to the adapter.
4. The files are split based on the `SplittingFunctionClassName` and `SplitCriteria` properties. The event file is split into several chunks and each chunk is posted to the export separately. This reduces memory loading during event processing.
 - If splitting is done based on a delimiter, the class that performs this function and the split criteria are provided.
 - If splitting is done based on file size, the class name that performs this function is provided.
 - If splitting is done based on other criteria, you must provide your own file splitting class.
5. The adapter sends the data, including the location of the polled document and the host name of the machine that the file was retrieved from, to the export through a function selector, where the configured data binding is invoked, to convert the text record into a business object.

Supported inbound operation

The adapter supports the emitFTPFile operation, which is taken as the default operation during inbound configuration.

Event file locking

File locking behavior is operating system dependent. On Windows®, if any of the files being polled by the adapter from the event directory are in use by another application and in the process of being copied to the event directory, they are not made available to the adapter for processing.

However, in UNIX® environments, such as AIX®, there is no file locking mechanism that prevents applications from accessing files that are being written to. A file that is being copied to the event directory by another application is made available to the adapter for processing, causing erroneous results. There is no platform-independent way in Java™ to check whether a file is being written to.

To prevent this situation from occurring, you can first copy the event file to a staging directory and then move it to the event directory using the move command. Some sample UNIX scripts are provided as part of the adapter. The script file named CheckIfFileIsOpen.sh is available in the Unix-script-file folder in the adapter installer.

Rule based filtering of events

The adapter supports the rule based filtering of events which is optional for inbound processing. You can filter the events based on multiple rules. You can define a combination of these rules, group them with Boolean logic and filter the events by following metadata:

- FileName
- File Size
- Last Modified

For example, you can use *FileName* "MatchesPattern" *.txt, where *FileName* is the property type, "MatchesPattern" is the operator and "*.txt" is the value.

Though rule is optional and event file mask is mandatory, the rule takes the higher precedence over event file mask, when both rule and event file mask are specified. Event file mask will be effective only when there is no rule specified. By default, event file mask will have "*" as default value.

Rule based filtering do not support logical "OR" operator between multiple rules.

Note: Adapter does not support rule based filtering when the EIS is on MVS™ platform

Table 2. Metadata filtering properties

Property	Valid operators	Value	Prerequisites
FileName	MATCHES_FILE_PATTERN	For example: *.txt	Nil
	MATCHES_REGEXP	Java Regular Expression	

Table 2. Metadata filtering properties (continued)

Property	Valid operators	Value	Prerequisites
FileSize	GT (greater than), LT (less than), GE (greater than or equal to), LE (less than or equal to), EQ (equal to), NE (not equal to).	Numeric value in Bytes. For example: FileSize LE 3628927	Nil
LastModified	GT (greater than), LT (less than), GE (greater than or equal to), LE (less than or equal to), EQ (equal to), NE (not equal to).	Day of the week or Time. The day-of-the-week value will accept only EQ (equal to) operator. For example :(LastModified EQ MONDAY), (LastModified EQ 20:41:10)	Nil
END-OF-RULE	END-OF-RULE	END-OF-RULE	Nil

Function selectors

During inbound processing, a function selector returns the appropriate operation to be called on the service. You choose a function selector when you configure the adapter for inbound processing in the external service wizard. The adapter provides two function selectors, `FilenameFunctionSelector` and `EmbeddedNameFunctionSelector`.

FilenameFunctionSelector

`FilenameFunctionSelector` is a rule-based function selector that provides object name resolution based on regular expressions that map to file names. A regular expression is a string that is used to describe or match a set of strings according to certain syntax rules.

The following table shows examples of matching rules, where a rule consists of the `ObjectName` and `Rule` fields.

Table 3. Examples of matching rules for `FilenameFunctionSelector`

FileName	ObjectName	Rule
Customer0001.txt	Customer	CUST.*TXT
22310RZ93.z21	Order	[0-9]*OR[A-Z][0-9]{2}.*
22310RZ93.z21	Order	*OR.*

Note that the rules in the second and third rows resolve to the same name, but the rule in the second row is less “greedy” because it requires a specific sequence of numbers and letters in order for the file name to match, whereas the rule in the third row resolves anything with the characters “OR” in the file name. The character combination “.*” indicates that any character may occur any number of times.

To generate the native function name, the function selector prepends `emit` to the object name that you provide. For example, if the object name is `Customer`, the function selector returns the function name `emitCustomer`. The object name should be the payload object name, for example, `Customer` or `Order`, and not the wrapper or business graph name. For pass-through scenarios, use `FTPFile` as the object name.

You can configure `FilenameFunctionSelector` with multiple rules, each containing an object name and a regular expression to match against the file name. If more

than one rule matches, the function selector returns the object name based on the first matching rule. If no rule matches, the adapter generates an error. If no rules are present in the configuration, the function selector uses the function name `emitFTPFile`.

For a detailed explanation of the rules governing the use of regular expressions, see the Java Class Pattern documentation at <https://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html>.

EmbeddedNameFunctionSelector

`EmbeddedNameFunctionSelector` is used for content-specific business objects, where the object name is embedded in the event file. It returns the function name based on the desired content data, not on the wrapper. For example, if the content-specific business object is `CustomerWrapperBG`, the function returned by the function selector is `emitCustomer`.

`EmbeddedNameFunctionSelector` must be configured with a data handler. The data binding must be the adapter-specific `WrapperDataBinding`, and it must be configured to use the same data handler that is configured with the function selector.

Inbound data transformation

Data transformation during inbound communications refers to the process by which the adapter transforms an event record created in a native format, such as bytes or a string, into a business object. The adapter uses an adapter-specific data binding and data handlers to accomplish this.

The data bindings and data handlers that the adapter uses to read the contents of the event record and fill the corresponding attributes in a business object are configured using the external service wizard in WebSphere Integration Developer.

Data bindings

To take fields from an event record, created in a native format, and populate a business object, the adapter needs a data binding. Data bindings are responsible for reading the event record fields and populating the corresponding fields in a business object. The adapter for FTP uses the `FTPFileBaseDataBinding` data binding during inbound communication.

During inbound communications, the data binding takes the following fields from an event record and populates the following business object attributes with their values:

- `Filename`
- `ChunkInfo`
- `DirectoryPath`
- `FileContentEncoding`
- `FtpServerHostName`
- `FtpServerEventDirectory`

For data that does not require transformation, the adapter conducts what is called pass-through processing because data passes through the system without being altered.

Data handlers

In addition to data bindings, data transformation requires the use of a data handler. A data handler converts data from a native format into a business object. In version 6.2 of WebSphere Adapter for FTP, the adapter provides the following data handlers:

- Delimited
- Fixed width
- XML

Passing files by reference

The adapter also supports a PassByReference feature, where only the event file name is sent to the export. The event file is appended with a time stamp and is available in the local archive directory. This feature is used when data transformation is not necessary.

Splitting files

The inbound event processing mode supports an optional file splitting feature, where the event file is split into several business objects, also known as chunks, and each business object is posted to the export separately. This reduces memory loading during event processing. File splitting is performed based on either a delimiter or on a file size specified in the SplitCriteria property.

The adapter provides SplitBySize and SplitByDelimiter classes for file splitting. Optionally, you can provide a custom file splitter class and use it by inputting the class name into the SplittingFunctionClassName property.

Splitting files by size

The size value is set in the SplittingFunctionClassName property.

Chunks refer to the resulting files after file splitting is performed. When chunking is enabled, each chunk of the file is posted to the export separately. The number of business objects that are specified in the PollQuantity property is posted to the export. For example, if the value for PollQuantity is 3, then:

The number of business objects polled is 3.

The number of business objects received by the export is 3.

The adapter does not reassemble chunked data. It provides the information about the chunked data for an external application to merge the chunks. The chunking information is set in the chunkInfo property, which is contained in the business object. This information includes the chunk size, in bytes, and the event ID. An example event ID is:

```
AbsolutePathOfTheEventFileNameInLocalEventDirectory/_/yyyy_MM_dd_HH_mm_ss_SSS.currentBONumber/_/to
```

Splitting files by delimiter

Delimiters are specified values used for splitting event files. The delimiter is specified in the SplitCriteria property.

The following rules apply to the use of delimiters:

- The specified delimiter must not be the same as any of the data contained within the business object. If it is the same, file splitting can produce incorrect results.
- The delimiter must contain the exact value of new line representation in the event file. If the event file is created on a MAC machine, the new line character is `\r`. On UNIX machines it is `\n`, and for windows machines it is `\r\n`.
- If there is more than one delimiter, each delimiter must be separated by a semicolon (`;`). If the semicolon is part of the delimiter, the semicolon must be escaped as `\;`. For example, if the delimiter is `##\;##` then it is processed as `##\;##`, which means that the semicolon is part of the delimiter.
- To skip content that is part of the delimiter, specify a double semicolon (`;;`) in front of it so that the content between the delimiters is skipped. For example, if the event file contains a business object in the following format and the delimiter is `##\;$$`, then:

```
Name=Smith
Company=IBM
```

```
##this is the content that will be skipped by the adapter$$
```

The adapter will consider `##$$` the delimiter and skip "this is the content that will be skipped by the adapter."

- The delimiter takes any value and there are no restrictions. The following are valid delimiter examples:
 - `####;\n;\n`
 - `####;$$$;\n;####`
 - `%%%;$$$$;#####`
 - `\n;\n;$$$$`
 - `####\;#####;\n;$$$$`
 - `\n;\n;\n`
 - `####;$$$$`
 - `\r`
 - `\r\n`
 - `$$$$;\r\n`
- If the delimiter is located at the end of the file, the `SplitCriteria` property uses `END_OF_FILE` to determine the physical end of the file.
- When each business object record in an event file is separated by a valid delimiter and if there is no delimiter or an invalid delimiter for the last business object record, the adapter will still be able to process the business object records.
- During inbound processing and splitting of the event file based on a delimiter, if the business object records present in an event file are separated by a delimiter and this delimiter is present at the beginning of each record instead of end of the record then the adapter considers that the delimiter is always present at the beginning of each record and processes accordingly.

Example 1:

```
John Doe,123,Washington Ave,222-123-4567
Jane Smith,234,Washington Ave,222-123-4568
```

The separator is the end of line character. In this example you would specify `\r\n` for Windows, `\r` for MAC, and `\n` for Unix.

Example 2:

John Doe
123 Washington Ave
222-123-4567

Jane Smith
234 Washington Ave
222-123-4568

The separator is ####.

Example 3:

```
ISAJohnDoe1*IBM*****USA*****  
ISAJohnDoe2*IBM*****USA*****  
ISAJohnDoe3*IBM*****USA*****
```

Content in an event file with a delimiter (ISA) at the beginning of each record.

Event recovery

The adapter supports event recovery for inbound processing in case of abrupt termination. During event processing, the adapter persists the event state in an event persistence table located on the data source. You must set up this data source before you can create the event persistence table.

To use the recovery feature provided by WebSphere Process Server, you must set the `AssuredOnceDelivery` property in the activation specification to `true`. If it is set to `false`, failed events cannot be recovered. Duplicate events can be delivered if `AssuredOnceDelivery` is set to `false`. To improve performance, you can set the event recovery, duplicate events, and `AssuredOnceDelivery` properties to `false`.

Event persistence table

The event persistence table is a persistent cache where events are saved until the adapter can process them. The adapter uses event persistence tables to keep track of inbound requests as they make their way through the system. Each time a file is created, updated, or deleted, the adapter tracks the activity as an event and updates the status of the event in the event persistence table. The status of each event is continually updated by the adapter for recovery purposes until the events are delivered to a configured export.

If the adapter detects that there is no event persistence table, it automatically creates one when the module is deployed to the runtime environment. Each event persistence table created by the adapter is associated with a specific inbound module. The adapter does not support multiple adapter modules pointing to the same event persistence table.

When the adapter polls the FTP server, it creates an entry in the event persistence table for each event that matches the search criteria specified in the activation specification properties. The adapter records the status of each new entry as `NEW`. When the adapter copies the event from the FTP server to the in progress folder on the local system, it marks the entry as `IN PROGRESS`. When the adapter sends the event to the function selector for data transformation, it deletes the entry from the event table.

Note: When guaranteed event delivery is not required, the adapter can poll for events without the existence of an event persistence table.

The following table describes each event persistence table value.

Table 4. Event persistence table structure

Column Name	Type	Description
EVNTID	Varchar(255)	A unique event ID for tracking purposes. The adapter uses this ID to track events during inbound processing.
EVNTSTAT	integer	The status of the event. The adapter uses the status to determine whether an event is new or in process. Event status values: NEWEVENT (0) The event is ready to be processed. PROCESSED (1) The adapter successfully processed and delivered the event. FAILED (-1) The adapter was unable to process this event due to one or more problems.
XID	Varchar(255)	Used by the adapter for assured event delivery and recovery.
EVNTDATA	Varchar(255)	Used by the adapter to mark the failed events as ARCHIVED to ensure that they are not processed again during adapter startup or recovery.

Event archive

Archived events are stored in the archive directory with a file extension that is specified in the FTPRenameExt property. Event archiving is an optional feature, which provides you with a record of all of the events that have been processed. You can use this information to review whether or not the events were processed successfully.

Event archiving is used differently in different configurations:

- When both the FTPArchiveDirectory and FTPRenameExt values are provided and FTPRenameExt is set to processed, the archived file is located in the specified archive directory with the following syntax:
filename_timestamp.processed
- When only the FTPArchiveDirectory value is provided, the archived file is located in the specified archive directory in the following syntax:
filename_timestamp
- When neither the FTPArchiveDirectory nor the FTPRenameExt values are provided, the event file is deleted from the event directory of the FTP server after the file is successfully downloaded to the local event directory.
- When only the FTPRenameExt value is provided and is set to processed, the archived file is located in the event directory of the FTP server with the following syntax: *filename_timestamp.processed*

Archiving on MVS platforms

Multiple Virtual Storage (MVS) operating systems do not support special characters such as an underscore in dataset or recordset names. On Windows and UNIX platforms, use a time stamp in the original file name while archiving the file.

This prevents duplicate file names in an archive folder, thereby preventing the overwriting of an existing file. Use the following format for MVS systems:

Event File: Test Archived

file: Test.TSyyyyMM.TSDDHHMM.TSSsSss

Where:

yyyy -- year

MM -- month

DD -- date

HH -- hour

MM -- minutes

Ss -- seconds

Sss -- milliseconds

The dataset or recordset separator is . (decimal) on MVS platforms. The maximum number of . (decimals) allowed in a dataset or recordset is six. The dataset or recordset name must not exceed eight characters per . (decimal), and the total number of characters must not exceed 44. An example of a file name in this format is:

FTPRenameExt: ARCHIVE

Archived File: TEST.TS200304.TS290535.TS42234.ARCHIVE

Business objects

A business object is a structure that consists of data, the action to be performed on the data, and additional instructions, if any, for processing the data. The data can represent either a business entity, such as an invoice or an employee record, or unstructured text.

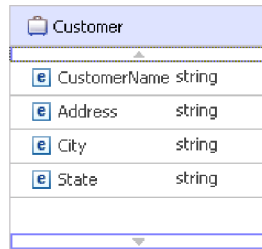
How the adapter uses business objects

The adapter uses business objects to send data to or obtain data from the FTP server. The adapter's main job during inbound operations is to take information from an event record created in a native format, convert it to a business object, and forward it to a service. For outbound operations, this process happens in reverse. The adapter receives a business object from a service, creates an event record from the details it finds in the business object, and then sends the event record to the FTP server.

How data is represented in business objects

Business objects are created using the business object editor in WebSphere Integration Developer, which provides a graphical view of your business objects. As shown in the following illustration, a business object consists of a set of fields and their values. This is a customer business object. As you can see, it records

name, address, and phone number information for a customer record. This example uses string values, but many other values are supported by business the object editor.



Customer	
e CustomerName	string
e Address	string
e City	string
e State	string

Figure 4. How data is represented in business objects

How business objects are created

You can create business objects by using either the external service wizard or the business object editor, both of which can be launched from WebSphere Integration Developer.

If you have defined XSD files using the business object editor prior to starting the external service wizard, the adapter will create business objects from these schemas. For instructions on how to use the business object editor to create business objects, refer to the following link: <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/in>. After you create your business objects, you can use the business object editor to define the hierarchy of the business objects.

Business graphs

You can optionally choose, during adapter configuration, to generate a business graph. In version 6.0.2, each top-level business object is contained in a business graph, which includes a verb that an application can use in version 6.0.2 to specify additional information about the operation to be performed. In Version 6.1.x, business graphs are optional; they are required only when you are adding business objects to a module created with a version of WebSphere Integration Developer earlier than Version 6.1.x. If business graphs exist, they are processed, but the verb is ignored.

WebSphere Application Server environment variables

When you configure the adapter for inbound or outbound processing using the external service wizard, you set values for various required local files and directories. You can later change these values in the deployed application from the WebSphere Process Server administrative console.

With WebSphere Process Server Version 6.2, instead of hard coding values for directories and files, you can declare them as WebSphere Application Server environment variables, and specify the environment variable names when you run the external service wizard. When you deploy your application, the environment variable name is replaced with the actual value and used by the adapter. If you want to change the property value, you can just change the environment variable in the WebSphere Process Server administrative console.

WebSphere Application Server environment variables can be used for all string property values (not boolean or integer variables) that are set in inbound and outbound configuration.

You define WebSphere Application Server environment variables using the WebSphere Process Server administrative console.

When you define a WebSphere Application Server environment variable, you specify:

- The name of the environment variable, for example, `EVENT_DIRECTORY`.
- The value that the symbolic name represents, for example: `C:\ftp\event`.
- The scope for the environment variable. The scope level determines the level at which the environment variable is visible in the administrative console. The scope level can be server, node, or cell:
 - Server scope limits visibility to the named server. The server scope is the most specific scope for defining environment variables.
 - Node scope limits visibility to all the servers on the named node. This is the default scope.
 - Cell scope limits visibility to all servers on the named cell.

See the topic [Defining WebSphere Application Server environment variables](#) for detailed information about how to create a WebSphere Application Server environment variable.

External service wizard

The external service wizard in WebSphere Adapter for FTP is used to create services and to generate business objects from the selected objects. The wizard also generates the service artifacts that enable the adapter to run as a Service Component Architecture (SCA) component.

Chapter 2. Planning for adapter implementation

To implement the IBM WebSphere Adapter for FTP, you must plan for inbound and outbound processing and consider security and performance requirements.

Before you begin

Before you begin to set up and use the adapter, you should possess a thorough understanding of business integration concepts, the capabilities and requirements of the integration development tools and run-time environment you will use.

To configure and use WebSphere Adapter for FTP, you should understand and have experience with the following concepts, tools, and tasks:

- The business requirements of the solution you are building.
- Business integration concepts and models, including the Service Component Architecture (SCA) programming model.
- The capabilities provided by the integration development tools you will use to build the solution. You should know how to use these tools to create modules, test components, and complete other integration tasks.
- The capabilities and requirements of the run-time environment you will use for the integration solution. You should know how to configure and administer the host server and how to use the administrative console to set and modify property definitions, configure connections, and manage events.
- The File Transfer Protocol (FTP), the protocol for exchanging files over the Internet.
- The FTP server being used to access the files on a specific file system in your solution.

Security

For secure communication, Secure Socket Layers (SSL) can be configured to protect the integrity of information being passed between the FTP server and the adapter. The adapter also supports FTP over SSH (SFTP), which is a network protocol that runs on a secure SSH channel on port 22. For users who require it, the adapter can also be configured to run in support of the Federal Information Processing Standard (FIPS) 140.

Configuring secure socket layers

Data that travels across a network can be intercepted by third parties. When this data includes private information such as passwords or credit card numbers, steps should be taken to make this data unintelligible to unauthorized users. By configuring secure socket layers (SSL), you protect the integrity of information being passed between the FTP server and the adapter. When the adapter is configured to work in secure FTP using Secure Socket Layer (SSL), both the control connection and data connection are encrypted.

Before you begin

To enable SSL, the following prerequisites must be satisfied:

- The FTP server supports secure communication using SSL.

- The FTP server has its own private key and certificate.
- The adapter uses a passive FTP mode of data transfer with a secure FTP server. If there is a firewall between the client and the server, the firewall settings might need to be configured to enable this mode.

Note: The adapter works in secure FTP using the SSL version 3.0 in implicit mode only. It does not support explicit mode. The adapter does not support:

- RFC 2228 FTP commands when working in secure FTP using SSL version 3.0
- Transport Layer Security (TLS) version 1.0

Note: The TLS version 1.0 is not supported in the non-FIPS mode.

About this task

Files passing through the FTP server are vulnerable to third party interference when SSL is not configured for use with the adapter. Using SSL prohibits data from being modified intentionally or unintentionally during transport and protects it from being intercepted. SSL is effective because it uses several cryptographic processes: public key cryptography for authentication with the FTP server and secret key cryptography and digital signatures for privacy and data integrity. SSL allows the adapter to authenticate the identity of the FTP server.

Procedure

1. Set the FTP client trust store. A trust store helps a FTP client decide what it can trust. When using SSL, WebSphere Process Server sends its certificate to the FTP client for verification. The FTP client verifies the certificate to ascertain that it is communicating with the intended FTP server. To enable this verification process, the FTP server's certificate should be present in the client's trust store.
 - a. In WebSphere Integration Developer, right-click the server instance and click **Run administrative console**.
 - b. Expand **Security**.
 - c. Select **SSL certificate and key management**.
 - d. Under **Related items**, select **Key stores and certificates**.
 - e. Select **NodeDefaultTrustStore**. See the Figure 5 on page 23 figure below.

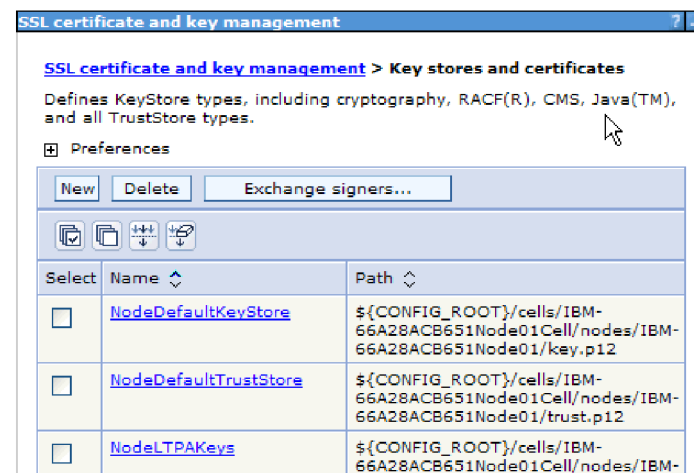


Figure 5. Selecting NodeDefaultTrustStore

- f. Under Additional properties, select **Signer certificates**.
- g. Click **Add**.
- h. In the **Alias** field, type the certificate name. See the Figure 6 figure below.

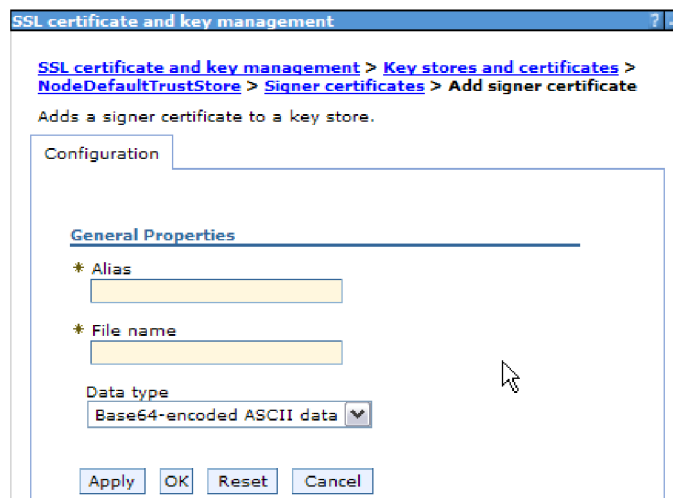


Figure 6. Adding signer certificate properties for the FTP server certificate

- i. In the **File name** field, type the full path of the FTP server certificate.
- j. Click **OK**

Note:

In both clustered and non-clustered WebSphere Process Server environments, when the trust store is created, the trust store information must be added to the default trust store provided by WebSphere Process Server, and not to the adapter-specific trust store. If the adapter uses its own trust store in clustered environments, it does not start during failover scenarios.

2. Configure SSL properties for the adapter.
 - a. In the external service wizard, set the Protocol to FTPS - file transfer protocol over secure socket layer (SSL).

Configuring support for SFTP

SSH over FTP (SFTP) is a network protocol that provides a mechanism for file transfer over a reliable data stream. SFTP runs on a secure SSH channel on port 22 and encrypts all traffic using either user name and password authentication or public key authentication. Public key authentication uses a pair of computer generated keys, one public and one private.

About this task

Configure the adapter for FTP to work with a secure SSH server:

Procedure

1. Install and configure your SSH server. There are various SSH servers to choose from. Install and configure your selected server using the provider-specific installation information.
2. See either Outbound or Inbound **Setting deployment and runtime properties** to select **SFTP - secure shell (SSH) file transfer protocol** and specify the SFTP server connection and security information in the external service wizard.

Configuring the adapter for federal information processing standard 140

The federal information processing standard 140 (FIPS) is an United States government standard for cryptographic features like encryption, decryption, hashing (message digests), secure socket layers, transport layer security, Internet protocol security, secure shell, signatures, key exchange, and key or certificate generation used in software products and modules. If you are a user working with the United States government who must conform to the FIPS standard, you can configure the adapter to run in FIPS mode.

About this task

Configuring the adapter to run in FIPS mode restricts the adapter to working with modules whose cryptographic features comply with FIPS approved methods and providers. From an adapter perspective, running in FIPS mode restricts the adapter to using the transport layer security (TLS) secure socket protocol.

Note: For the adapter to run in FIPS mode, the FTP server must support SSL v3.1, which is the same as TLS v1.0, and it must be enabled through the wizard of the FTP server. If not properly supported by SSL v3.1, the SSL handshake with the adapter may fail.

To run the adapter in FIPS mode, you must instruct the adapter to use the IBM Java Secure Socket Extension (IBMJSSE2) provider package. The IBMJSSE2 provider is the preregistered Java secure socket extension provider in the Java security file in IBM SDK, version 5.0. IBMJSSE2 uses FIPS-approved packages.

Complete the following steps to run the adapter in FIPS mode:

Procedure

1. In the IBMJSSE2 provider, set the `com.ibm.jsse2JSSEFIPS` property to `True`.
2. Set the following security properties so the IBMJSSE2 provider will handle all JSSE requests.

- a. Set the `ssl.SocketFactory.provider` property to `com.ibm.jsse2SSLSocketFactoryImpl`.
- b. Set the `ssl.ServerSocketFactory.provider` property to `com.ibm.jsse2SSLServerSocketFactoryImpl`.
3. In the security properties file, add the IBMJCEFIPS provider `com.ibm.crypto.fips.provider.IBMJCEFIPS` to the provider list above the IBMJCE provider. Follow the `security.provider.n=providername` format where *n* denotes the order of the provider. The provider with a value of 1 is considered before the provider with a value of 2. Do not remove the IBMJCE provider.
4. Set system properties in the WebSphere Process Server administrative console Java virtual machine (JVM) properties. Follow the `-Dpropertyname=propertyvalue` format.
5. Set security properties in the `Java.security` file (located in the *WebSphere Process Server Java virtual machine/lib/security* directory).

For more details on configuring security details, see the security documentation for WebSphere Process Server or WebSphere Enterprise Service Bus.

Server verification

Server verification is a method where the client verifies the identity of the server before establishing the connection.

The adapter performs the server verification when the SFTP protocol is enabled. The adapter checks the SFTP server that it is trying to establish a connection with to see whether it is a trusted server or not.

Server verification requires a host key file as the input. The host key file must be available on the adapter workstation with the host keys of the trusted servers added to it. The entries in the host key file have to adhere with the OpenSSH format of the `KNOWN_HOSTS` file.

The adapter verifies the server by comparing the host key presented by the server with the host keys present in the host key file. The adapter connects to the server only if the host key of the server is available in the host key file. If the host key of a trusted server is different from the one that is present in the host key file, then the host key entry in the host key file has to be modified to reflect the new entry.

If the server is not trusted (the host key is not present in the host key file), the adapter does not connect to the server, and the connection request fails, indicating that a connection was attempted to a non-trusted server and connection could not be established due to security reasons.

Note: If you want to provide both the host name and the IP address, specify them, separating them with a comma, in the host key entry.

Public key authentication

Public key authentication is one of the most secure methods used to authenticate when using a Secure Shell. Public key authentication uses a pair of computer generated keys, one public and one private. The public key can be distributed and resides in the SFTP server. The private key is unique to the user and must not be shared.

The following properties are required to enable public key authentication:

- Host name
- Port number
- User name
- Private key

Passphrase is an optional property that is used to provide extra protection for the private key.

The key-pair can be generated using any third party service and you can choose any of the standard encryption algorithms. The most commonly used algorithm is RSA; however, other algorithms such as DSA can be used.

Note: The key-pair must be in the OpenSSH format.

If both the Password (username and password authentication) and the Private key (public key authentication) values are specified in the external service wizard, the Private key property value takes precedence. The adapter then tries to authenticate to the server using public key authentication.

Support for confidential logging and tracing

The adapter provides the ability to protect sensitive or confidential data in log and trace files from being seen by those without authorization.

Log and trace files for the adapter can contain data from your FTP server, which might contain sensitive or confidential information. Sometimes it is necessary for these files to be seen by individuals without authorization to view sensitive customer data, for example, a support specialist who must use the log and trace files to troubleshoot a problem.

To protect the data in situations like this, the adapter provides the HideConfidentialTrace property. The HideConfidentialTrace property specifies whether or not you want to prevent confidential user data from displaying in the adapter log and trace files. When this property is enabled, the adapter replaces the confidential data with XXX's.

The following types of information are considered potentially sensitive data and will be hidden:

- The contents of a business object
- The contents of an event record
- User ID
- Business object data in an intermediate form, such as a comma-delimited version of a file

The following types of information are not considered user data and will not be hidden:

- Business object schemas
- Transaction IDs
- Event IDs
- Call sequences

User authentication

The adapter supports several methods for supplying the user name and password that are needed to connect to the FTP server. By understanding the features and limitations of each method, you can pick a method that provides the appropriate level of security and convenience for your application.

To integrate an adapter into your application, you must provide the user name and password for the adapter to use at run time on WebSphere Process Server or WebSphere Enterprise Service Bus to connect to the FTP server to process outbound requests and inbound events.

At run time, the adapter needs to provide the user name and password to connect to the FTP server. To connect without user intervention, the adapter must access a saved copy of the user information. In a server environment, there are several methods for saving user information. The external service wizard lets you configure the adapter to get the user information by using any of the following methods:

- Adapter properties
- Data source
- J2C authentication alias

Saving the user name and password in adapter properties is a direct way to provide this information at run time. You provide the user name and password when you use the external service wizard to configure your module. Although directly specifying the user name and password seems the most straightforward method, it has important limitations. Adapter properties are not encrypted; the password is stored as clear text in fields that are accessible to others on the server. Also, when the password changes, you must update the password in all instances of the adapter that accesses that FTP server, including the adapters embedded in application EAR files as well as adapters that are separately installed on the server.

By using a data source, you can use a connection already established for another application. For example, if multiple applications access the same database with the same user name and password, the applications can be deployed using the same data source. The user name and password can be known only to the first person who deploys an application to that data source or who defines a data source separately.

Using a J2C authentication data entry, or authentication alias, created with the Java Authentication and Authorization Service (JAAS) feature of Java 2 security is a robust, secure way to deploy applications. An administrator creates the authentication alias that is used by one or more applications that need to access a system. The user name and password can be known only to that administrator, who can change the password in a single place when a change is required.

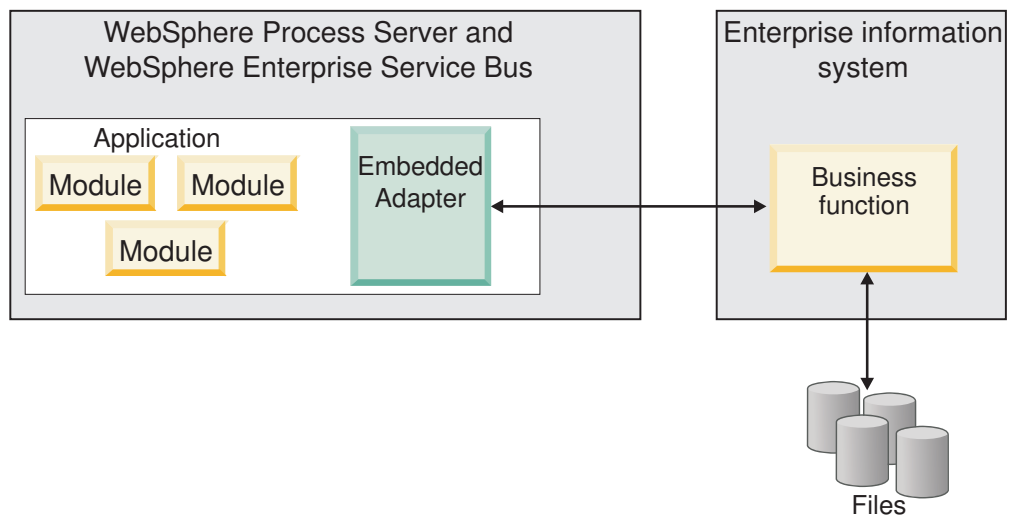
Deployment options

There are two ways to deploy the adapter. You can either embed it as part of the deployed application, or you can deploy it as a stand-alone RAR file. The requirements of your environment will affect the type of deployment option you choose.

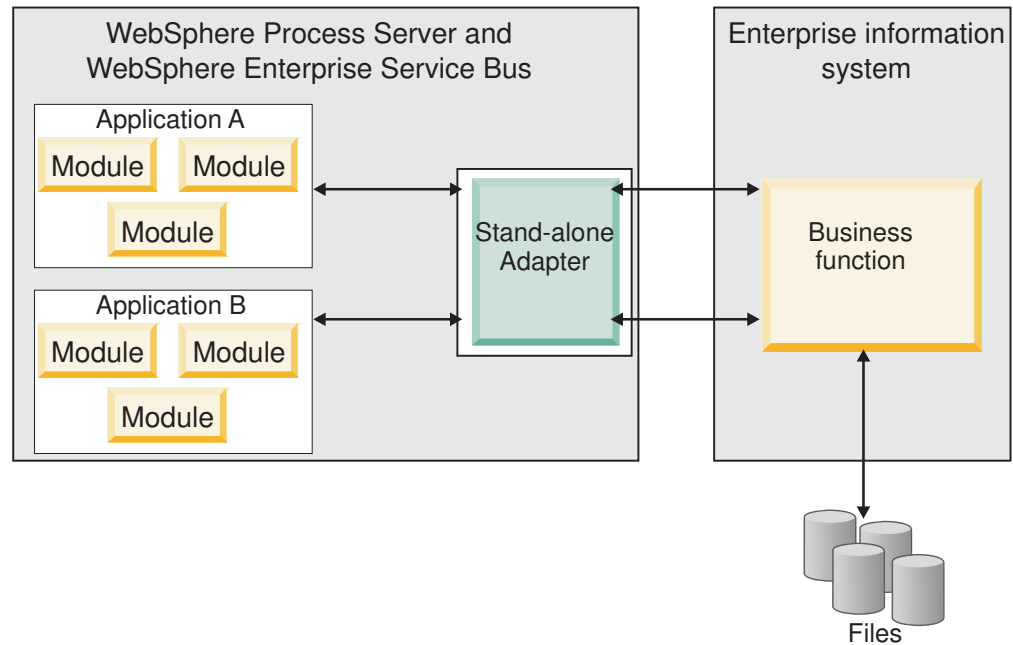
The deployment options are described below:

- **With module for use by single application.** With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
- **On server for use by multiple applications.** If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.

An embedded adapter is bundled within an enterprise archive (EAR) file and is available only to the application with which it is packaged and deployed.



A stand-alone adapter is represented by a stand-alone resource adapter archive (RAR) file, and when deployed, it is available to all deployed applications in the server instance.



While creating the project for your application using WebSphere Integration Developer, you can choose how to package the adapter [either bundled with the (EAR) file or as a stand-alone (RAR) file]. Your choice will affect how the adapter is used in the runtime environment, as well as how the properties for the adapter are displayed on the administrative console.

Choosing either to embed an adapter with your application or to deploy the adapter as a stand-alone module depends on how you want to administer the adapter. If you want a single copy of the adapter and do not care about disruption to multiple applications when you upgrade the adapter, then you would be more likely to deploy the adapter as a stand-alone module.

If you plan on running multiple versions, and if you care more about potential disruption when you upgrade the adapter, you would be more likely to embed the adapter with the application. Embedding the adapter with the application allows you to associate an adapter version with an application version and administer it as a single module.

Considerations for embedding an adapter in the application

Take into consideration the following items if you plan on embedding the adapter with your application:

- An embedded adapter has class loader isolation.
A class loader affects the packaging of applications and the behavior of packaged applications deployed on runtime environments. *Class loader isolation* means the adapter cannot load classes from another application or module. Class loader isolation prevents two similarly named classes in different applications from interfering with each other.
- Each application in which the adapter is embedded must be administered separately.

Considerations for using a stand-alone adapter

Take into consideration the following items if you plan on using a stand-alone adapter:

- Stand-alone adapters have no class loader isolation.
Because stand-alone adapters have no class loader isolation, only one version of any given Java artifact is run and the version and sequence of that artifact is undetermined. For example, when you use a stand-alone adapter there is only *one* resource adapter version, *one* adapter foundation class (AFC) version, or *one* third-party JAR version. All adapters deployed as stand-alone adapters share a single AFC version, and all instances of a given adapter share the same code version. All adapter instances using a given third-party library must share that library.
- If you update any of these shared artifacts, all applications using the artifacts are affected.
For instance, if you have an adapter that is working with server version X, and you update the version of the client application to version Y, your original application might stop working.
- Adapter Foundation Classes (AFC) is compatible with previous versions, but the latest AFC version must be in every RAR file that is deployed in a stand-alone manner.
If more than one copy of any JAR file is in the class path in a stand-alone adapter, the one that is used is random; therefore, they all must be the latest version.

WebSphere Adapters in clustered environments

You can improve adapter performance and availability by deploying the module to a clustered server environment. The module is replicated across all servers in a cluster, regardless of whether you deploy the module using a stand-alone or embedded adapter.

WebSphere Process Server or WebSphere Enterprise Service Bus, WebSphere Application Server Network Deployment, and WebSphere Extended Deployment support clustered environments. Clusters are groups of servers that are managed together to balance workloads and to provide high availability and scalability. When you set up a server cluster, you create a Deployment Manager profile. The HAManager, a subcomponent of the Deployment Manager, notifies the Java 2 Platform, Enterprise Edition (J2EE) Connector Architecture (JCA) container to activate the adapter instance. The JCA container provides a runtime environment for adapter instances. For information about creating clustered environments, see the following link: http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm_cluster_v61.html.

Using WebSphere Extended Deployment, you can optionally enhance the performance of adapter instances in your clustered environment. WebSphere Extended Deployment extends the WebSphere Application Server Network Deployment capabilities by using a dynamic workload manager instead of a static workload manager, which is used by WebSphere Application Server Network Deployment. The dynamic workload manager can optimize the performance of adapter instances in the cluster by dynamically balancing the load of the requests. This means that application server instances can be automatically stopped and started based on the load variations, allowing machines with different capacities

and configurations to evenly handle load variations. For information on the benefits of WebSphere Extended Deployment, see the following link: <http://publib.boulder.ibm.com/infocenter/wxdinfo/v6r1/index.jsp>.

In clustered environments, adapter instances can handle both inbound and outbound processes.

Restriction: During inbound communication WebSphere Adapter for FTP is not able to switch polling between a WebSphere Process Server or WebSphere Enterprise Service Bus cluster backup node and the cluster's primary node when each node is installed on a different operating system. For example, if the adapter starts polling on a primary Windows node, it cannot switch to a backup UNIX node because it cannot process the Windows path used for the directory storing in progress events.

High availability for inbound processes

Inbound processes are based on events triggered as a result of updates to data in the FTP server. WebSphere Adapter for FTP is configured to detect updates by polling an event table. The adapter then publishes the event to its endpoint.

Important: In a clustered environment, the event directory should be on a shared file system and not local to any of the cluster machines.

When you deploy a module to a cluster, the JCA container checks the `enableHASupport` resource adapter property. If the value for the `enableHASupport` property is true, which is the default setting, all of the adapter instances are registered with the `HAManager` with a policy 1 of N. This policy means that only one of the adapter instances starts polling for events. Although other adapter instances in the cluster are started, they remain dormant with respect to the active event until the active adapter instance finishes processing the event. If the server on which the polling thread was started shuts down for some reason, an adapter instance that is running on one of the backup servers is activated.

Important: Do not change the setting of the `enableHASupport` property.

High availability for outbound processes

In clustered environments, multiple adapter instances are available to perform outbound process requests. Accordingly, if your environment has multiple applications that interact with WebSphere Adapter for FTP for outbound requests, then you might improve performance by deploying the module to a clustered environment. In a clustered environment, multiple outbound requests can be processed simultaneously, as long as they are not attempting to process the same record.

If multiple outbound requests are attempting to process the same record, such as a Customer address, the workload management capability in WebSphere Application Server Network Deployment distributes the requests among the available adapter instances in the sequence they were received. As a result, these types of outbound requests in a clustered environment are processed in the same manner as those in a single server environment: one adapter instance processes only one outbound request at a time. For more information on workload management, see the following link: http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html.

Adapter customization with Customer Parser Class

The WebSphere Adapter for FTP uses Apache Commons Net API v1.4.1 to connect to the FTP server. Commons Net API provides an interface, through the Custom Parser Class Name property, that you can implement to parse the long list (ls -l) output.

This is helpful when the adapter must connect to FTP servers where the ls -l output deviates from the standard output. By using the class that implements this interface, the adapter can work with these FTP servers.

The adapter provides a basic implementation of this interface. The class name is com.ibm.j2ca.ftp.util.FTPLongListEntryParser. This class has been tested with FTP server running on Windows® 2000 server.

The Custom Parser Class Name property, which is located in the activation specification and also in the managed connection factory, takes the fully qualified name of this class.

The following methods are located in the Java™ interface:

```
package org.apache.commons.net.ftp;
public interface FTPFileEntryParser{
FTPFile parseFTPEntry(String listEntry);
String readNextEntry(BufferedReader reader) throws IOException;
List preparse(List original);
}
```

For more information about each of these methods in the Apache Commons Net 1.4.1 documentation, see <http://commons.apache.org/net/api/index.html>

Migrating to version 6.2

By migrating to version 6.2 of WebSphere Adapter for FTP, you automatically upgrade from the previous version of the adapter. Additionally, you can migrate your applications that embed an earlier version of the adapter, so that the applications can utilize features and capabilities present in version 6.2.

Migration considerations

WebSphere Adapter for FTP version 6.2 includes updates that might affect your existing adapter applications. Before migrating applications that will use WebSphere Adapter for FTP, you must consider some factors that might affect your existing applications.

Compatibility with earlier versions

WebSphere Adapter for FTP version 6.2 is fully compatible with version 6.0.2.x and version 6.1.x of the adapter and can work with custom business objects (XSD files) and data bindings.

Because version 6.2 of WebSphere Adapter for FTP is fully compatible with version 6.0.2.x and version 6.1.x, any of your applications that used version 6.0.2.x or version 6.1.0.x of WebSphere Adapter for FTP will run unchanged when you upgrade to version 6.2. However, if you want your applications to use features and functionality present in version 6.2 of the adapter, run the migration wizard.

The migration wizard replaces (upgrades) version 6.0.2.x or version 6.1.x of the adapter with version 6.2 and enables version 6.2 features and functionality for use with your applications.

Note: The migration wizard does not create components or modify existing components, such as mappers and mediators to work with version 6.2 of the adapters. If you are upgrading to version 6.2 and any of your applications embed an adapter that is version 6.1.x or earlier, and you want your applications to take advantage of the features and functions in version 6.2, you might need to make changes to those applications.

If artifacts are inconsistent with regard to versioning within a single module, the entire module is marked and will not be selectable for migration. Version inconsistencies are recorded in the workspace log, because they might indicate corruptions to the project.

If the WebSphere Integration Developer version does not match that of WebSphere Adapter for FTP, the products cannot be run together.

The external service wizard cannot be run when the following product combinations are used:

Table 5. Product combinations not supported

WebSphere Integration Developer	WebSphere Adapter for FTP
version 6.2	version 6.1.x
version 6.1.x	version 6.0.2.x
version 6.0.2.x	version 6.2
version 6.0.2.x	version 6.1.x

Deciding whether to upgrade or to upgrade and migrate

By default, the migration wizard upgrades the adapter and migrates the application artifacts so that the applications can use features and functions in version 6.2 of the adapter. When you choose to upgrade the adapter by selecting a project, the wizard automatically selects the associated artifacts for migration.

If you decide that you want to upgrade the adapter from version 6.0.2.x or version 6.1.x to version 6.2, but you do not want to migrate the adapter artifacts, you can do so by deselecting the adapter artifacts from the appropriate area of the migration wizard.

Running the migration wizard without any adapter artifacts selected will install and upgrade your adapter, but your artifacts will not be migrated and your applications will not be able to take advantage of the features and capabilities that exist in version 6.2 of the adapter.

Running the migration wizard in a test environment first

Because adapter migration might require you to make changes to those applications that will use version 6.2 of WebSphere Adapter for FTP, you should always perform the migration in a development environment first and test your applications before deploying the application to a production environment.

The migration wizard is fully integrated with the development environment.

Deprecated features

A deprecated feature is one that is supported but no longer recommended and that might become obsolete. The following features from earlier versions of WebSphere Adapter for FTP have been deprecated in version 6.1.x and version 6.2 and might require changes to your applications:

- The EventContentType and DefaultObjectName Activation specification properties
- The FTPURL Managed Connection Factory property
- The FTPFileDataBinding data binding
- The annotation tags contained in the XSD files

Performing the migration

You can migrate a project or EAR file to version 6.2, by using the adapter migration wizard. When the tool is finished, the migration is complete and you can work in the project or deploy the module.

Before you begin

Review the information in *Migration considerations*.

About this task

To perform the migration in WebSphere Integration Developer, complete the following steps.

Note: After migration is complete, the module will no longer be compatible with previous versions of WebSphere Process Server or WebSphere Enterprise Service Bus, or WebSphere Integration Developer.

Note: The following steps describe how to run the adapter migration wizard from the connector project context menu while in the Java EE perspective in WebSphere Integration Developer.

Procedure

1. Import the PI (project interchange) file for an existing project or the EAR (enterprise archive) file for a deployed application into the workspace.
2. Change to the Java EE perspective.
3. Right-click the module and select **Update** → **Update Connector Project**.

Note: Additionally, you can launch the adapter migration wizard in the following ways:

- Right-click the project in the Java EE perspective and select **Update** → **Migrate Adapter Artifacts**.
- From the Problems view, right-click a migration-specific message and select **Quick Fix** to correct the problem.

When you launch the migration wizard from the connector project context menu while in the Java EE perspective, by default all of the dependent artifact projects are selected. If you deselect a dependent artifact project, that project is not migrated. You will be allowed to migrate any projects that you deselect at a later time. Previously migrated projects, projects with a current version, and projects that contain errors are unavailable for migration and are not selected.

4. Review the tasks and warnings presented on the welcome page, and click **Next**.
5. A warning window opens with the message, "The properties that are not supported in the version of the target adapter will be removed during the migration." Click **OK** to continue to the next step.
6. On the Review changes window you can review the migration changes that will occur in each of the artifacts that you are migrating by clicking the + sign.
7. Click **Finish** to perform the migration.
 Prior to running the migration process, the wizard performs a backup up of all projects affected by the migration. The projects are backed up to a temporary folder within the workspace. Should the migration fail for any reason, or should you decide to cancel the migration before it completes, the wizard deletes the modified projects and replaces them with the projects stored in the temporary folder.
 Upon completing the migration successfully, all backed up projects are deleted.
8. Check the Problems view for messages from the migration wizard, which start with the string CWPAD.
9. If you are migrating an EAR file, optionally create a new EAR file with the migrated adapter and artifacts, and deploy it to WebSphere Process Server or WebSphere Enterprise Service Bus. For more information about exporting and deploying an EAR file, see the topics devoted to it in this documentation.

Results

The project or EAR file is migrated to version 6.2. You do not need to run the external service wizard after exiting the adapter migration wizard.

Updating but not migrating a version 6.0.2.x or version 6.1.x project

You can upgrade the adapter from version 6.0.2.x or version 6.1.x to version 6.2, while choosing not to migrate the adapter project artifacts.

About this task

Because the internal name of the adapter changed in version 6.2, artifacts in a version 6.0.2.x or version 6.1.x project must be updated to use the new name before you can use the adapter wizard in WebSphere Integration Developer, version 6.2. Use the migration wizard to update a version 6.0.2.x or version 6.1.x project. Then use the Quick Fix feature of WebSphere Integration Developer to change the adapter name in project artifacts.

Procedure

1. Import the project interchange (PI) file into the workspace.
2. In the Java EE perspective, right-click the project name and click **Update** → **Update Connector Project**. The adapter migration wizard opens.
3. On the welcome page, click **Next**.
4. On the Select Projects window, deselect the dependent artifact projects, and then click **Finish**.
5. A warning window opens with the message, "The properties that are not supported in the version of the target adapter will be removed during the migration." Click **OK** to continue to the next step.
6. In the Quick Fix window, make sure the fix **Rename the referenced adapter** is selected, and then click **OK**.

7. If the error remains visible, click **Project** → **Clean**, select the project you just updated, and then click **OK**.

Results

The project can now be used with WebSphere Adapter for FTP, version 6.2.

Chapter 3. Samples and tutorials

To help you use WebSphere Adapters, samples and tutorials are available from the Business Process Management Samples and Tutorials Web site.

You can access the samples and tutorials in either of the following ways:

- From the welcome page that opens when you start WebSphere Integration Developer. To see samples and tutorials for WebSphere Adapter for FTP, click **Retrieve**. Then browse the displayed categories to make your selections.
- At this location on the Web: Business Process Management Samples and Tutorials <http://publib.boulder.ibm.com/bpcsamp/index.html>.

Chapter 4. Configuring the module for deployment

To configure the adapter so that it can be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus, use WebSphere Integration Developer to create a module, which is exported as an EAR file when you deploy the adapter. You then specify the business objects you want to build and the system on which you want to build them.

Roadmap for configuring the module

Before you can use WebSphere Adapter for FTP in a runtime environment, you must configure the module. Understanding this task at a high level helps you perform the steps that are needed to accomplish the task.

You configure the module for WebSphere Adapter for FTP by using WebSphere Integration Developer. The following figure illustrates the flow of the configuration task, and the steps that follow the figure describe this task at a high level only. For the details about how to perform each of these steps, see the topics following this roadmap.

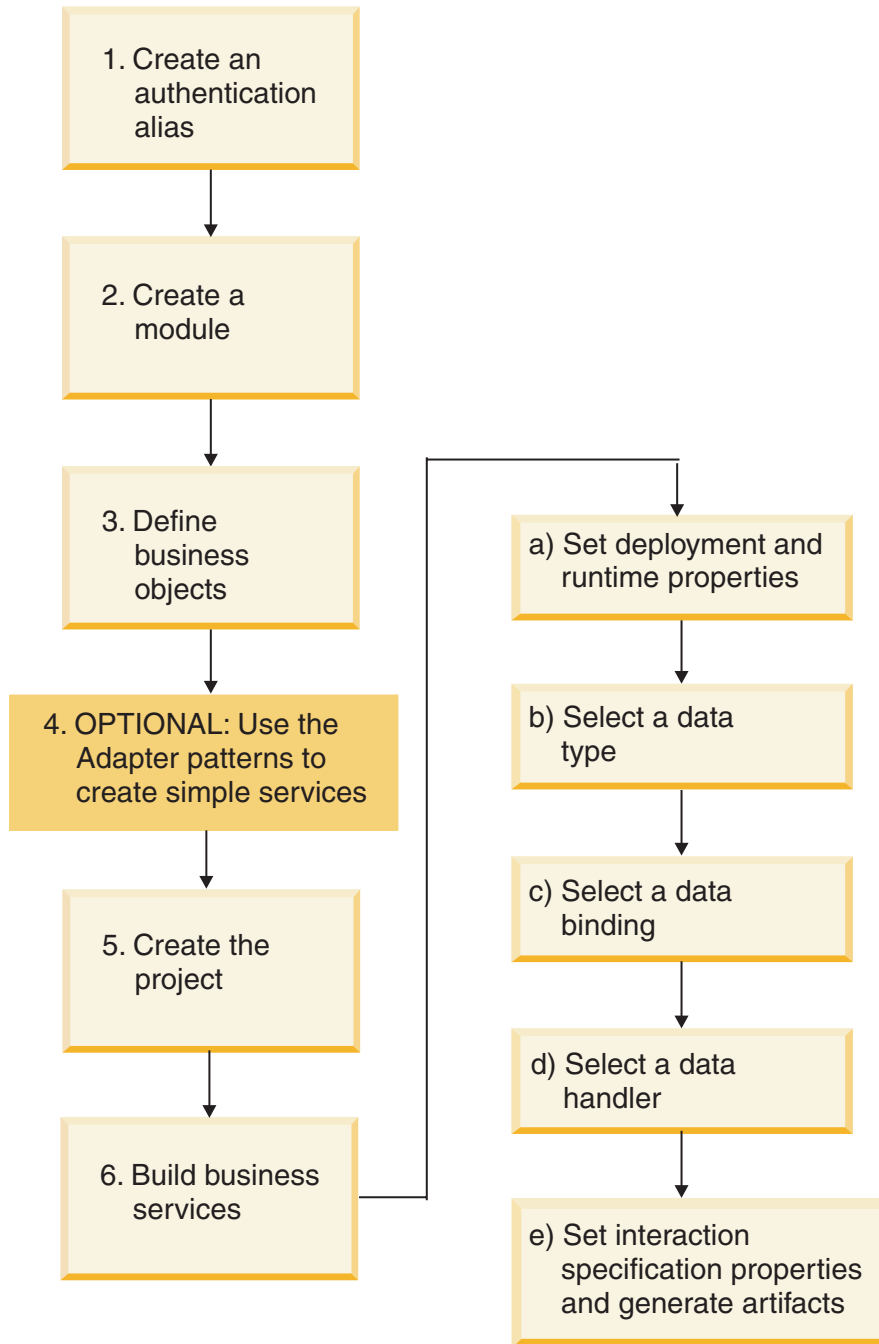


Figure 7. Roadmap for configuring the module

Configuring the module

This task consists of the following steps, which are described at a high level.

Note: These steps assume that you are using user-defined business objects that require data transformation. If using generic business objects, which do not require data transformation, some of the following steps will be ignored. For example, you will not need to select a data binding and a data handler.

1. Create an authentication alias to access the FTP server. Perform this step using the administrative console on the server.

2. Create a module in WebSphere Integration Developer. You create business objects in the module.
3. Define the business objects that will be used by the project.
4. Use the Adapter patterns wizard to create simple services. For more information, see “Creating a simple service with the adapter pattern wizard” on page 68.
5. Create a project, which is used to organize the files associated with the adapter using the external service wizard in WebSphere Integration Developer.
6. Build business services by running the external service wizard from WebSphere Integration Developer, then performing the following steps:
 - a. Specify the following deployment and runtime properties:
 - Connection properties
 - Security properties
 - Deployment options
 - Function selector - Inbound only
 - b. Select a data type and name the operation associated with this data type. For each operation, specify the following:
 - The operation kind. For example, Create, Append, Exists.
 - Specify if the operation is passthrough or user defined.
 - c. Select the data binding. Each data type has an equivalent data binding used to read the fields in a business object and fill the corresponding fields in a file.
 - d. Select the data handler that will perform the conversions between a business object and a native format.
 - e. Specify interaction specification property values and generate artifacts. The output from running the external service wizard is saved to a business integration module, which contains the business object or objects, and the import or export file.

Creating an authentication alias

An authentication alias is a feature that encrypts the password used by the adapter to access the FTP server. The adapter can use it to connect to the FTP server instead of using a user ID and password stored in an adapter property.

Before you begin

To create an authentication alias, you must have access to the administrative console of WebSphere Process Server or WebSphere Enterprise Service Bus. You must also know the user name and password to use to connect to the FTP server.

The following procedure shows you how to gain access to the administrative console through WebSphere Integration Developer. If you are using the administrative console directly (without going through WebSphere Integration Developer, log in to the administrative console and skip to step 2 on page 42.

About this task

Using an authentication alias eliminates the need to store the password in clear text in an adapter configuration property, where it might be visible to others.

To create an authentication alias, use the following procedure.

Procedure

1. Start the administrative console.
To start the administrative console through WebSphere Integration Developer, perform the following steps:
 - a. In the Business Integration perspective of WebSphere Integration Developer, click the **Servers** tab.
 - b. If the server does not show a status of **Started**, right-click the name of the server (for example, **WebSphere Process Server**) and click **Start**.
 - c. Right-click the name of the server and click **Run administrative console**.
 - d. Log onto the administrative console. If your administrative console requires a user ID and password, type the ID and password and click **Log in**. If the user ID and password are not required, click **Log in**.
2. In the administrative console, click **Security** → **Secure administration, applications, and infrastructure**.
3. Under **Authentication**, click **Java Authentication and Authorization Service** → **J2C authentication data**.
4. Create an authentication alias.
 - a. In the list of J2C authentication aliases that is displayed, click **New**.
 - b. Click the **Configuration** tab, then type the name of the authentication alias in the **Alias** field.
 - c. Type the user ID and password that are required to establish a connection to the FTP server.
 - d. Optional: Type a description of the alias.
 - e. Click **OK**.
The newly created alias is displayed.
The full name of the alias contains the node name and the authentication alias name you specified. For example, if you create an alias on the node widNode with the name ProductionServerAlias, then the full name is widNode/ProductionServerAlias. This full name is the one you use in subsequent configuration windows.
 - f. Click **Save**, and then click **Save** again.
5. Click **New**.

Results

You have created an authentication alias, which you will use when you configure the adapter properties.

Creating the module

You create the module in WebSphere Integration Developer. The module allows you to define business objects that will be used by the project.

Procedure

1. Right-click anywhere within the Business Integration workspace of the WebSphere Integration Developer window, and then select **New** → **Module**.

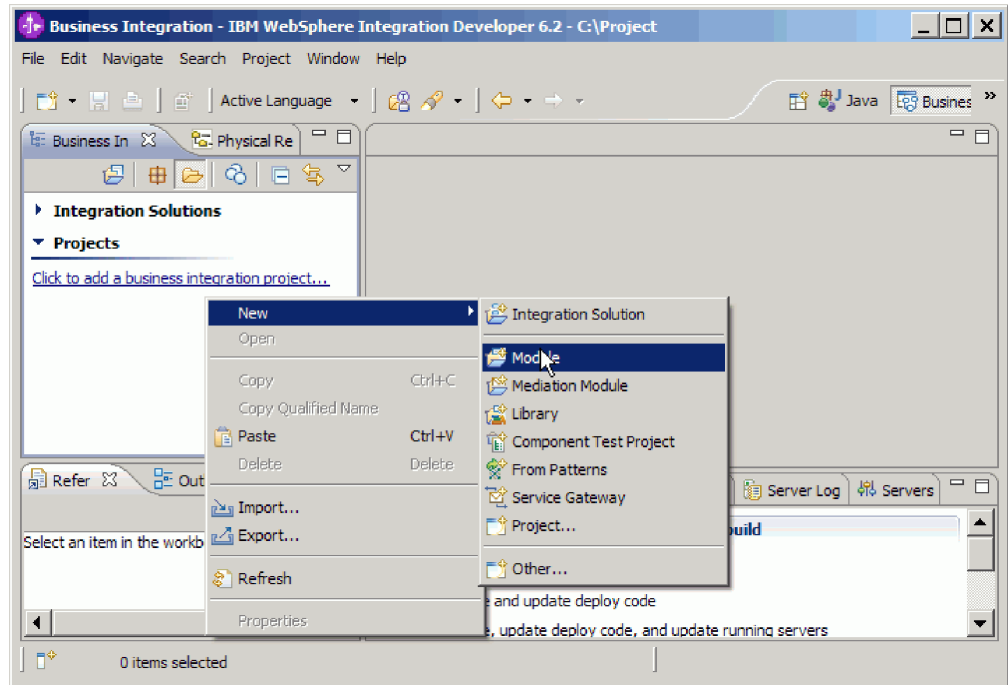


Figure 8. Creating a new module from the Business Integration perspective

2. Type a new **Module Name** in the New Module window.

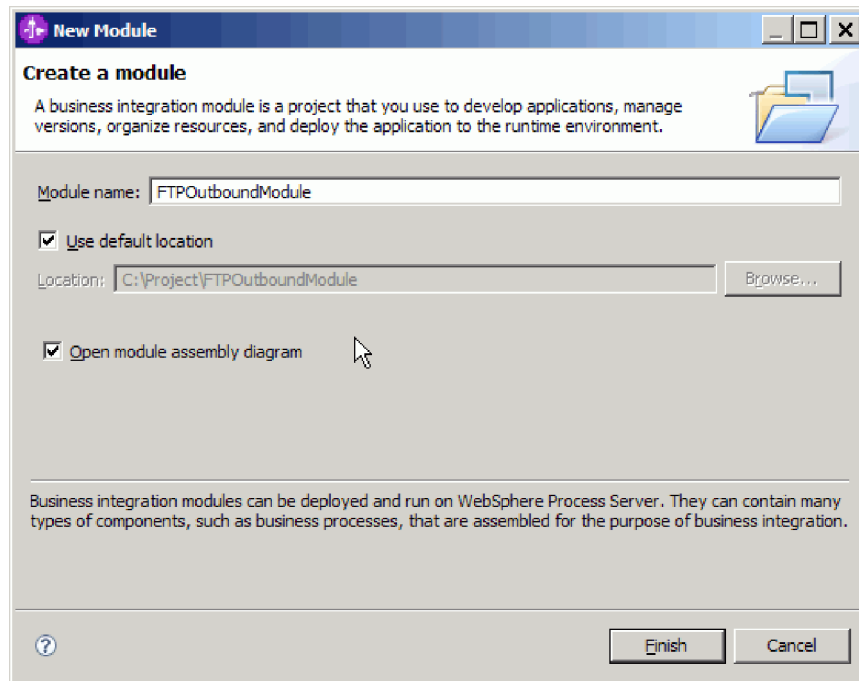


Figure 9. New Module window

3. Click **Finish**.

Results

A new module is listed in the Business Integration window.

What to do next

Create a project, which is used to organize the files associated with the adapter.

Defining business objects

Predefine the business objects in WebSphere Integration Developer that will be used by the project that you will create in the next topic.

Procedure

1. Expand the new module located inside of the Business Integration section of the WebSphere Integration Developer window.
2. Right-click the **Data Types** folder and select **New > Business Object**.

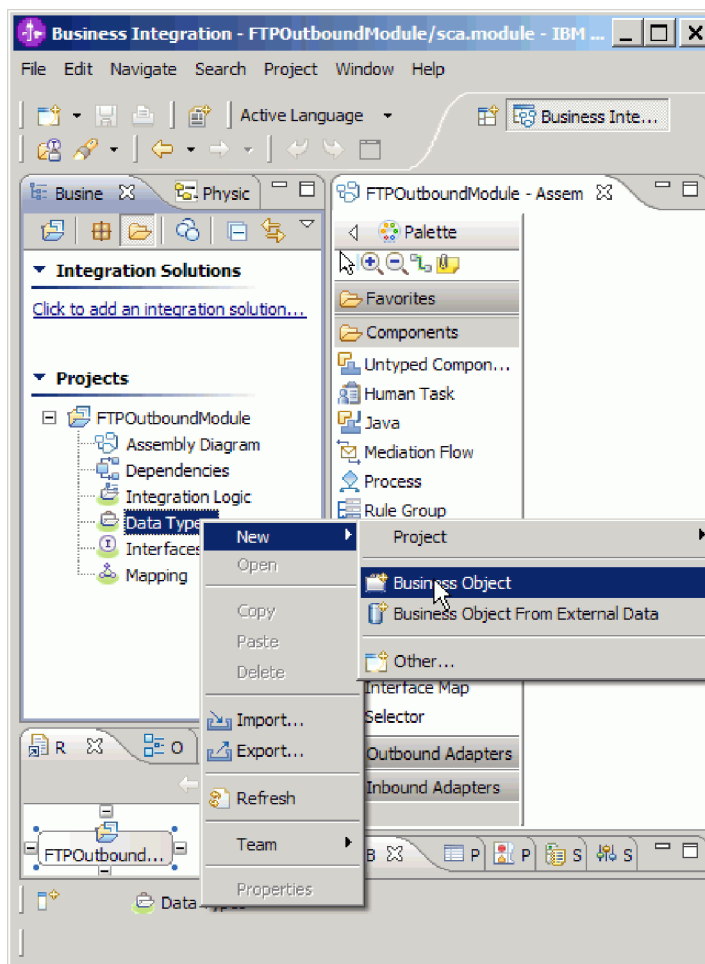


Figure 10. New Business Object create view

3. Type in a new **Name** in the Business Object window.

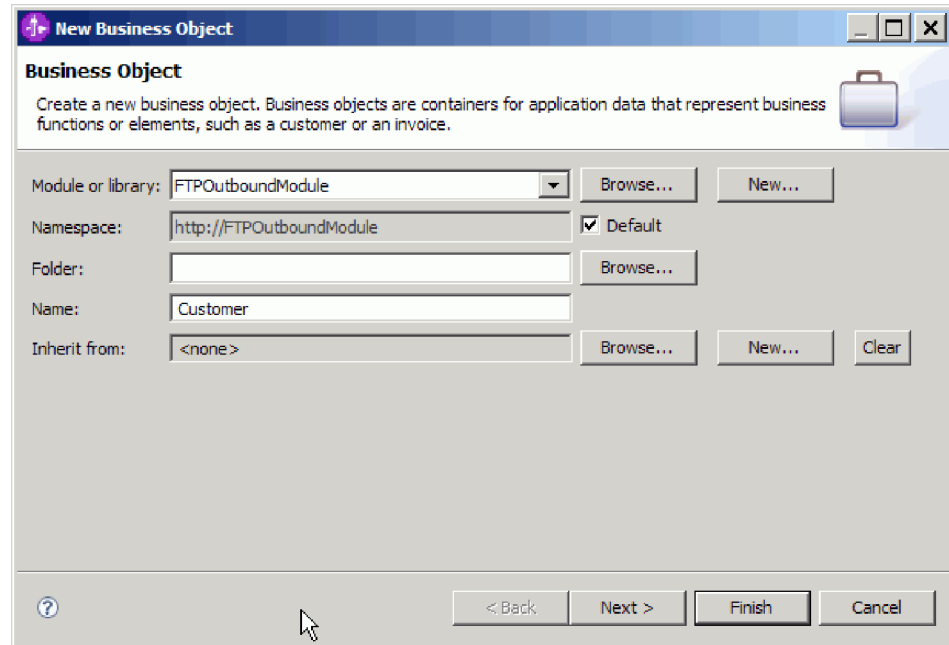


Figure 11. Business Object window

4. Click **Finish**. The new business object is added to the **Data Types** folder.
5. Click the **Add a field to a business object** icon and add the necessary fields to the business object.

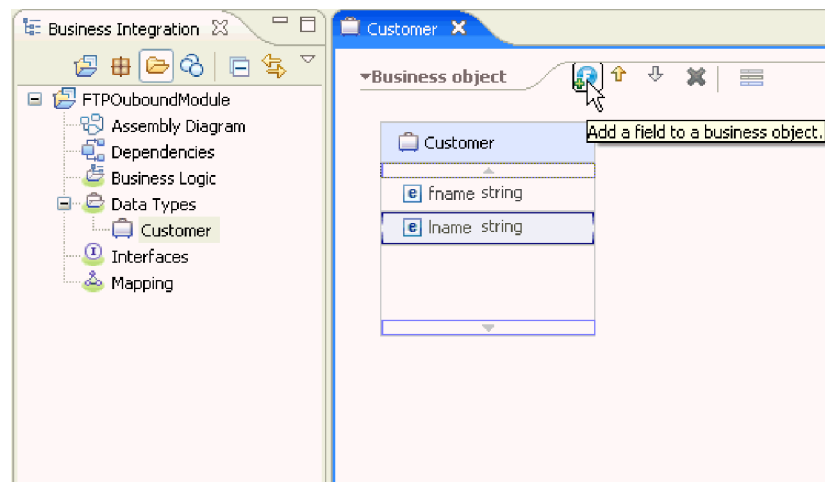


Figure 12. Add Business object fields icon

6. Click the Save icon.
7. Repeat the previous steps for each business object that you want to create.

Results

The new business objects are defined.

What to do next

Create a project, which is used to organize the files associated with the adapter.

Converting business objects to COBOL copybook files during outbound processing

Use the external data wizard in WebSphere Integration Developer to generate business object definitions from a COBOL program source file. These business object definitions are used during outbound processing.

Before you begin

Before you perform this task, make sure that:

1. You have created a module in WebSphere Integration Developer.
2. The COBOL program source file (.ccp file) is in a local directory on your workstation.
3. If you are going to generate a wrapper business object definition, you have imported the adapter RAR file into your workspace.

About this task

Use the external data wizard to generate a business object definition for a COBOL program source file. After you have generated the business object definition, you can run the external data wizard again to generate a wrapper business object definition from the generated business object.

Procedure

1. Generate the business object for the COBOL program source file.
 - a. In the Business Integration section of the window, right-click the module and select **New** → **Business Object From External Data**.
 - b. In the Business Object From External Data window, expand **Languages** and select **Cobol**.

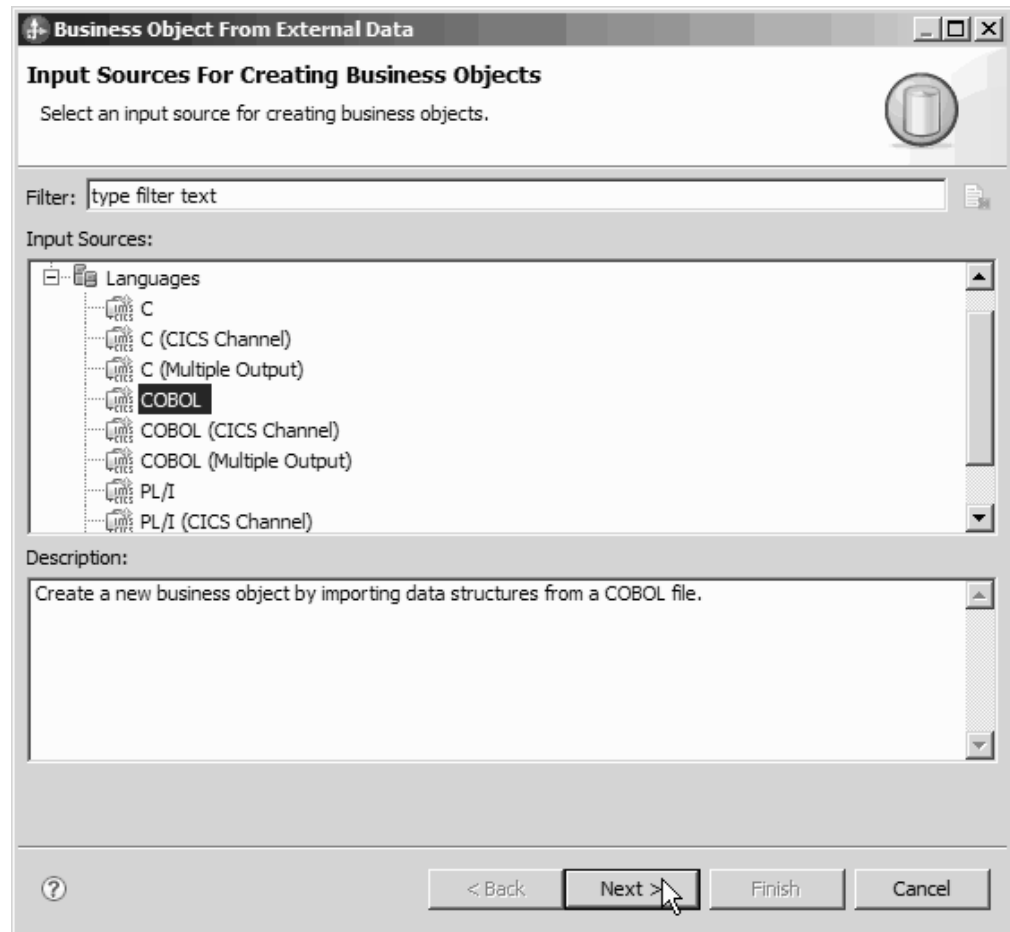


Figure 13. The Business Object From External Data window

- c. Click **Next**.
- d. In the Business Object Mapping Details window, click **Browse** and select the .ccp file (in this example, taderc99.ccp is used).

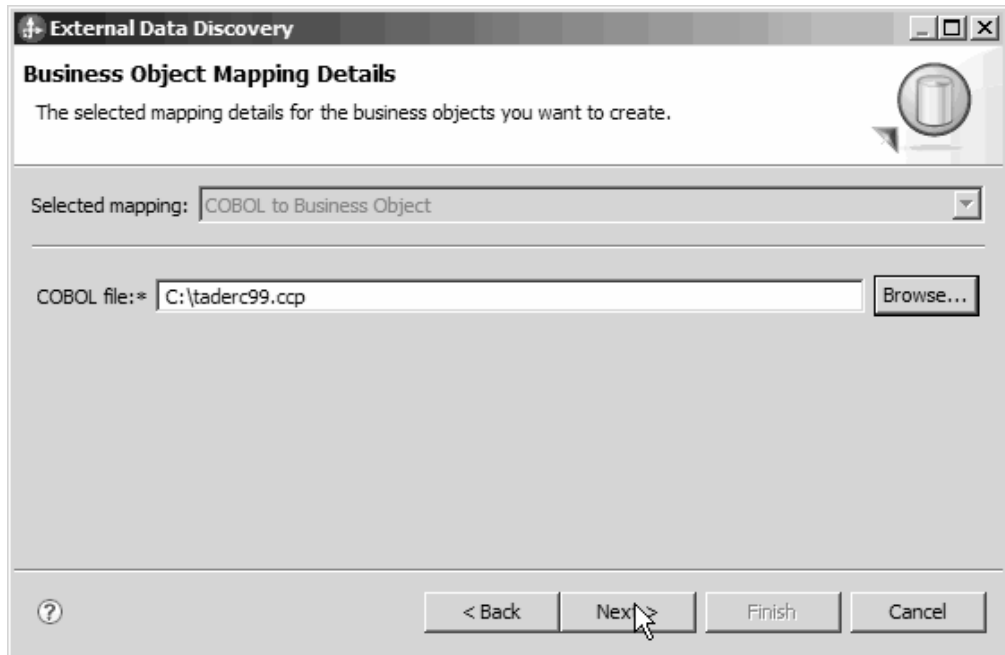


Figure 14. The Business Object Mapping Details window

- e. Click **Next**.
- f. In the Select Data Structures window, click **Find**. The new business object called DFHCOMMAREA is displayed.

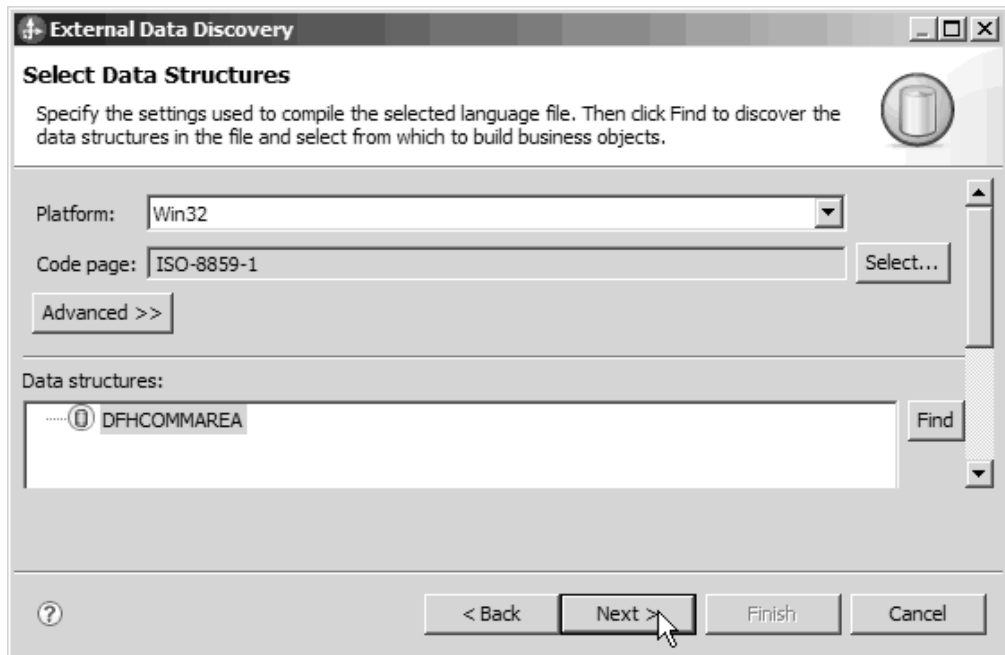


Figure 15. The Select Data Structures window

- g. Select DFHCOMMAREA and click **Next**.
- h. Click **Finish**.

A business object called DFHCOMMAREA is created in the module.

2. Optional: Generate a wrapper business object definition. Wrapper business object definitions wrap existing business object definitions with additional function. The generation of wrapper business object definitions is optional. The option to generate wrapper business object definitions is displayed only if the adapter RAR file is imported into the workspace.

Note: If you want to generate wrapper business object definitions, then you must generate them before running the external service wizard.

- a. In the Business Integration section of the window, right-click on the module and select **New** → **Business Object From External Data**.
- b. In the Business Object From External Data window, expand **Adapters**, select the adapter connector project for which you want to generate the wrapper business object .

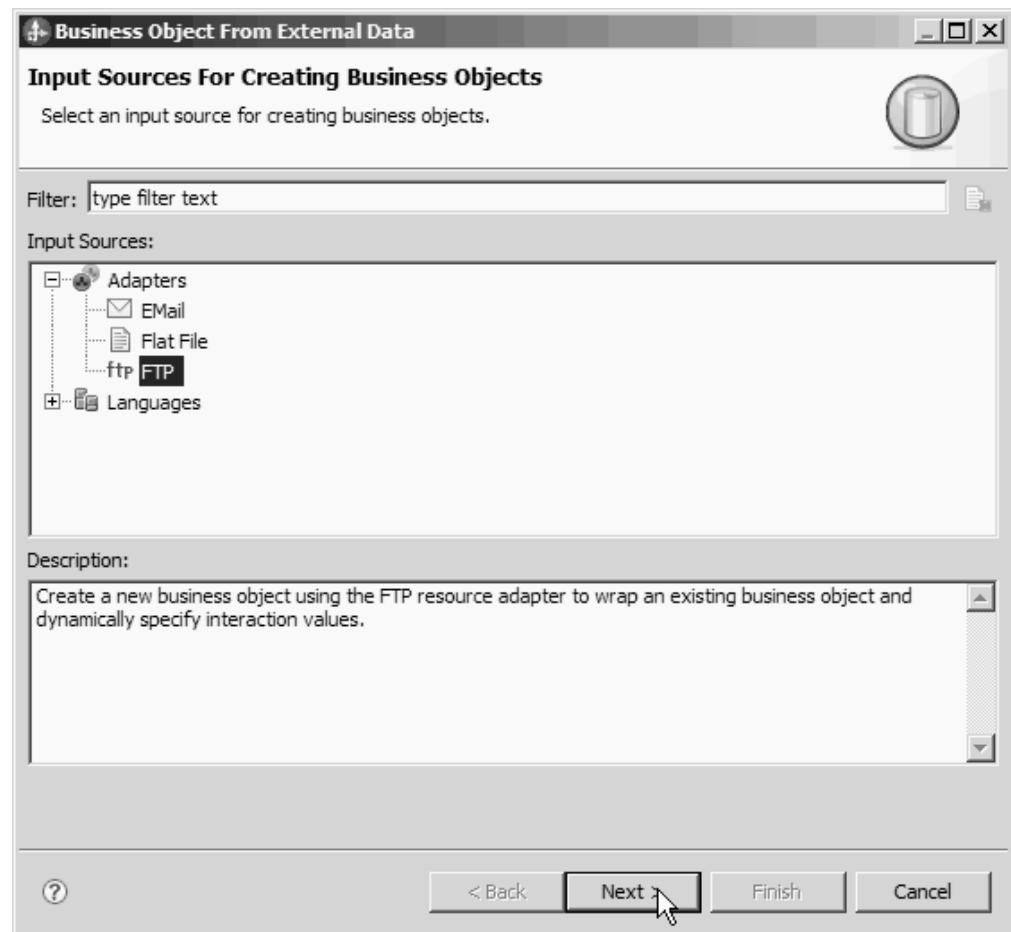


Figure 16. The Business Object From External Data window

- c. Click **Next**.
- d. In the Business Object Properties window, click **Browse** and select DFHCOMMAREA for the data type.
- e. To generate a business graph, select the **Generate business graph for each business object** check box. To generate retrieve wrapper, select the **Generate retrieve container to retrieve multiple business objects** check box.

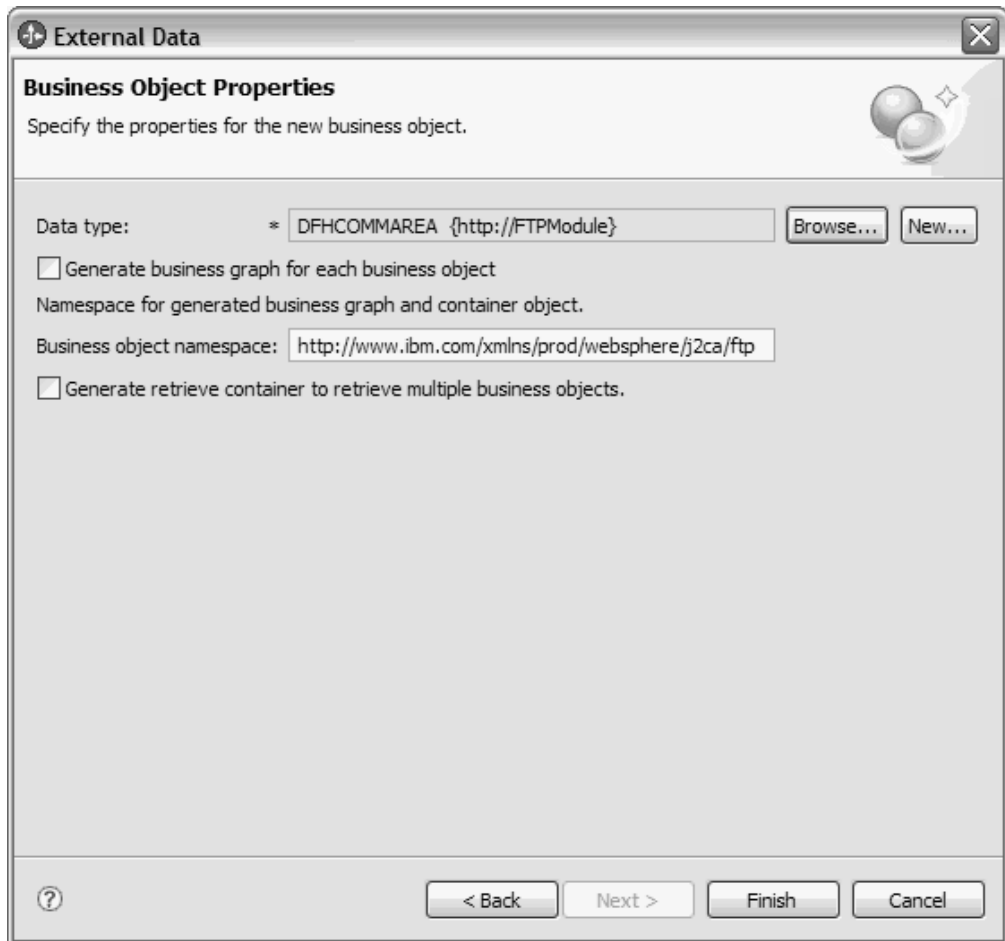


Figure 17. The Business Object Properties window

f. Click **Finish**.

A wrapper business object called DFHCOMMAREAWrapper and a business graph called DFHCOMMAREAWrapperBG are listed for the current module in the Business Integration window. If the user has selected **Generate business graph for each business object** and **Generate retrieve container to retrieve multiple business objects**, then a business object called DFHCOMMAREARetrieveWrapper and a business graph called DFHCOMMAREARetrieveWrapperBG are also listed for the current module in the Business Integration window.

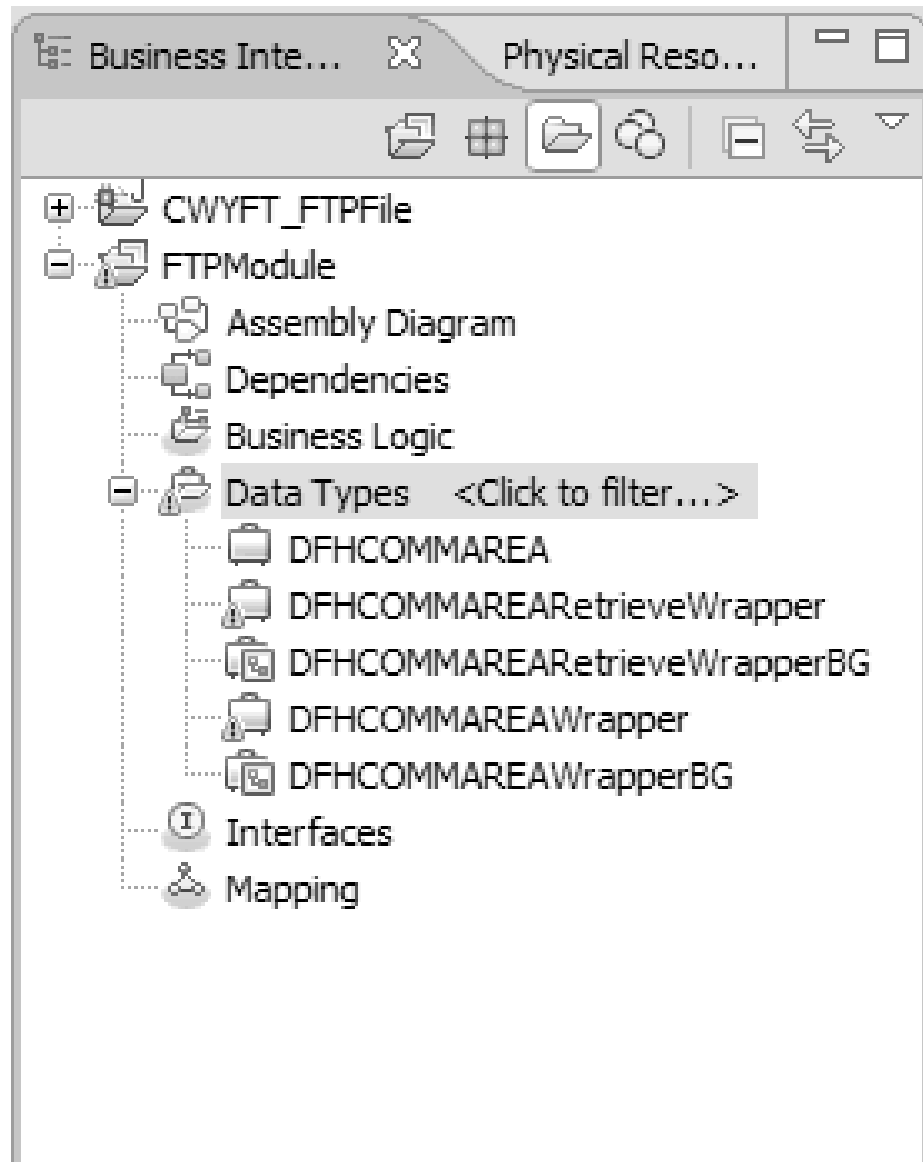


Figure 18. The wrapper business object and the business graph listed in the Business Integration window

3. Generate the required artifacts for the COBOL copybook outbound module. This example shows the configuration for a Create operation.
 - a. In the Business Integration section of the window, right-click on the module and select **New** → **External Service**.
 - b. Select **Adapters** and click **Next**.
 - c. In the Select an Adapter window, select the FTP adapter connector project and click **Next**.

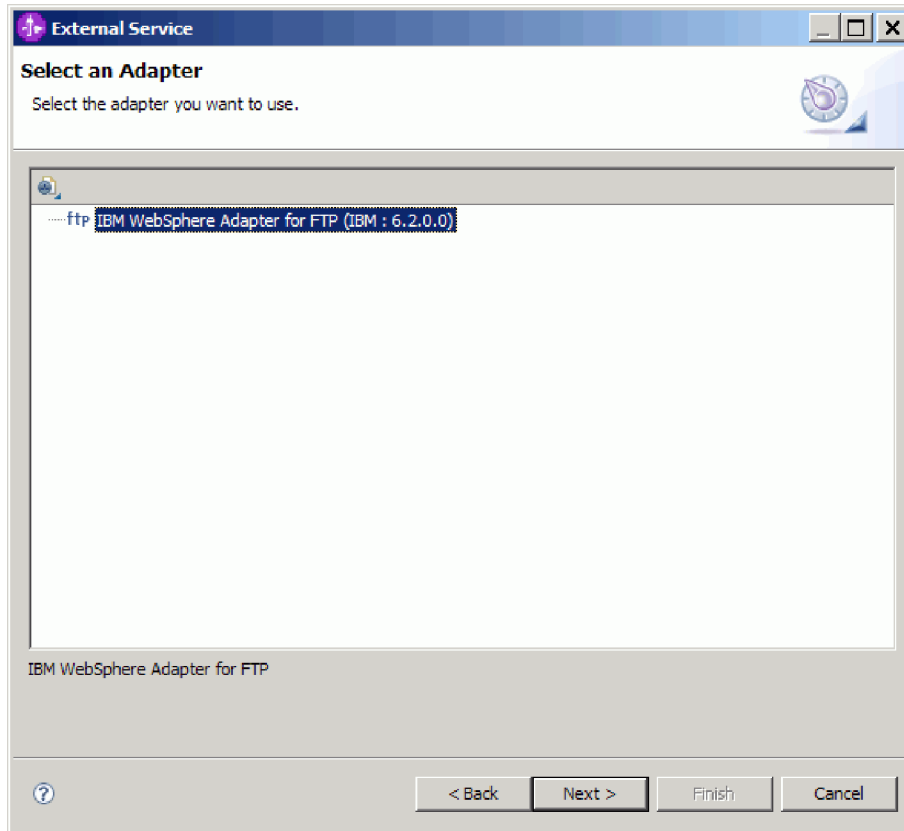


Figure 19. The Select an Adapter window

- d. In the Processing Direction window, select **Outbound**.
- e. Click **Next**.
- f. In the Service Configuration Properties window, in the **Data format options** list, select **Use COBOL, C or PL/I data binding**.

Note: This is not a data binding, but a data binding generator. The tool generates the appropriate data binding code for you in the current module.

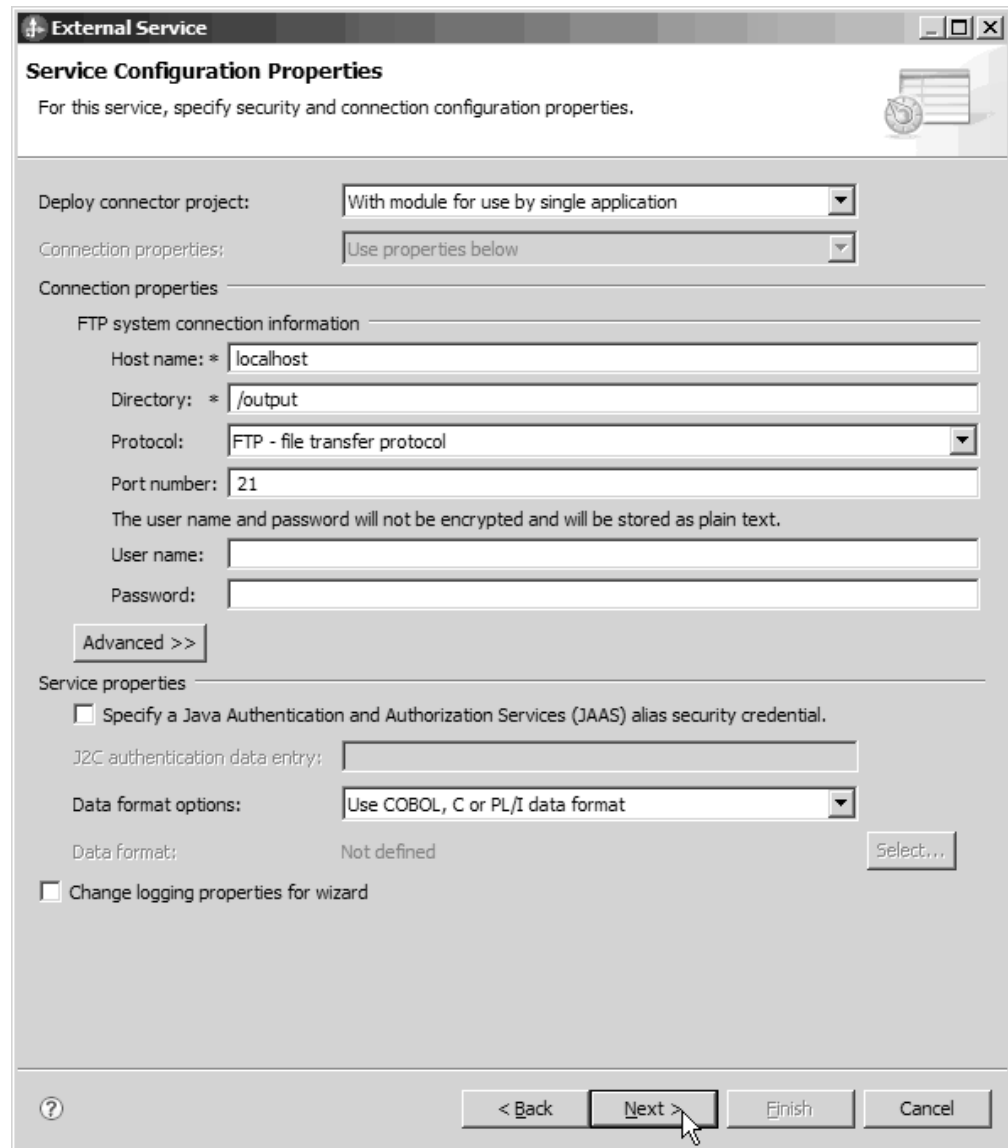


Figure 20. The Service Configuration Properties window

- g. Specify the other required properties for the outbound operation, and click **Next**.
- h. In the Operations window, click **Add** and then **Create**. For the retrieve operation select **Retrieve**. Select **User defined type** from the list for the data type and click **Next**.
- i. Browse for the input type (either DFHCOMMAREA, DFHCOMMAREAWrapper, or DFHCOMMAREAWrapperBG) and click **OK**. For the **Retrieve** operation, browse for the appropriate output type (either DFHCOMMAREA, DFHCOMMAREARetrieveWrapper, or DFHCOMMAREARetrieveWrapperBG).



Figure 21. The Data Type Selection window

j. Click **Next**.

The data bindings used by the COBOL copybook, WSDL files, import files, and other artifacts are generated. See the Project Explorer window for the generated data binding classes.

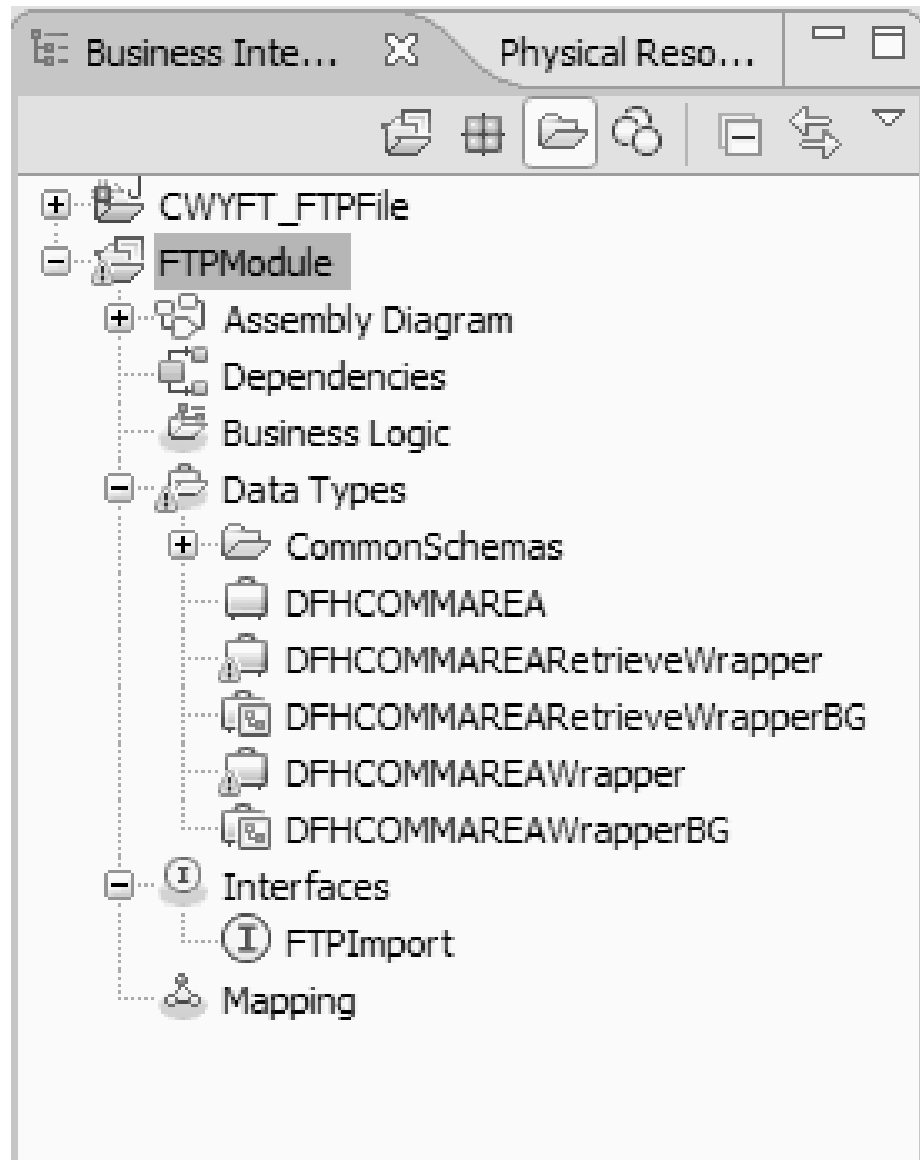


Figure 22. Data bindings used by the COBOL copybook, WSDL files, import files, and other artifacts

Results

A business object, a wrapper business object, and a business graph are created for the COBOL program source file for the outbound module. Artifacts are generated for an outbound Create operation that uses the COBOL copybook data binding. This module can be deployed on WebSphere Process Server and tested for the Create operation.

Note: To generate artifacts for other supported operations (Append and Overwrite), follow the same steps, beginning with Step 3h.

What to do next

Deploy the module.

Converting COBOL copybook files to business objects during inbound processing

Use the external data wizard in WebSphere Integration Developer to generate business object definitions from a COBOL program source file. These business object definitions are used during inbound processing.

Before you begin

Before you perform this task, make sure that:

1. You have created a module in WebSphere Integration Developer.
2. The COBOL program source file (.ccp file) is in a local directory on your workstation.
3. You have created a local event directory.
4. If you are going to generate a wrapper business object definition, you must have imported the adapter RAR file into your workspace.

About this task

Use the external data wizard to generate a business object definition for a COBOL program source file. After you have generated the business object definition, you can optionally rerun the external data wizard to generate a wrapper business object definition from the generated business object.

Procedure

1. Generate the business object for the COBOL program source file.
 - a. In the Business Integration section of the window, right-click on the module and select **New** → **Business Object From External Data**.
 - b. In the Business Object From External Data window, expand **Languages** and select **COBOL**.

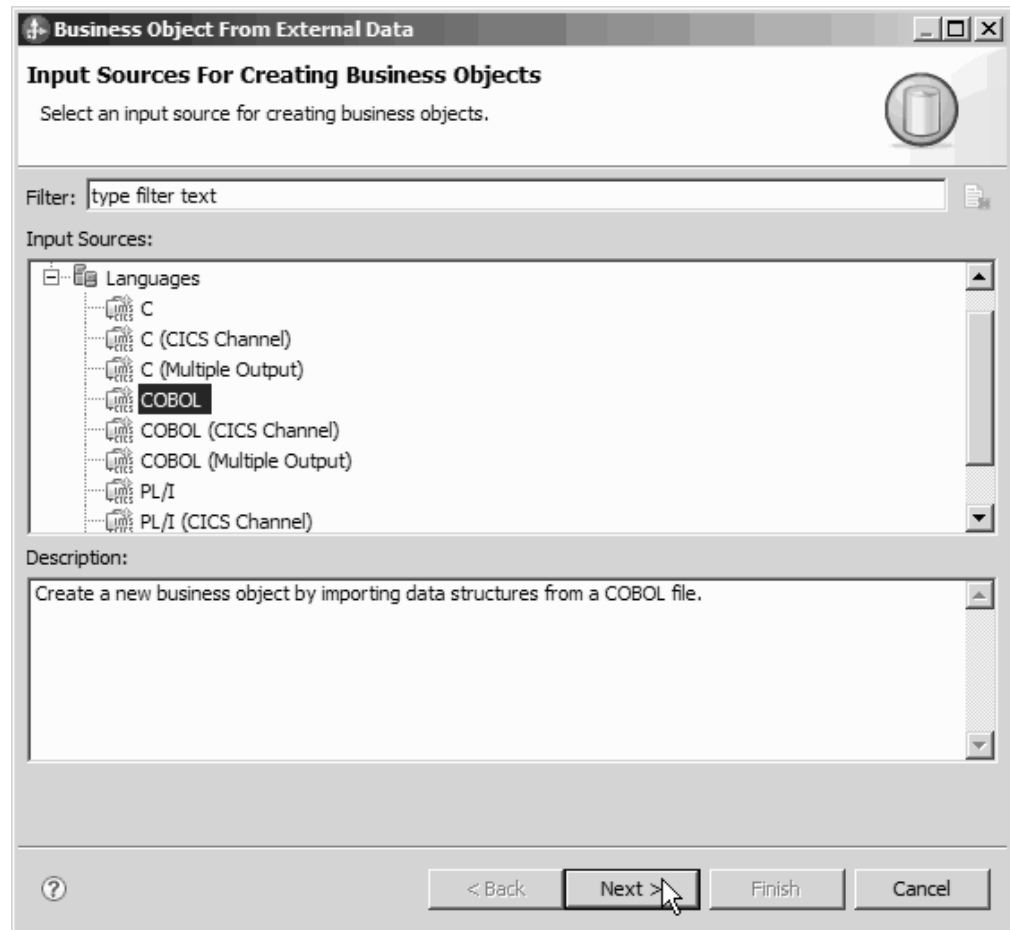


Figure 23. The Business Object From External Data window

- c. Click **Next**.
- d. In the Business Object Mapping Details window, click **Browse** and select the .ccp file (in this example, taderc99.ccp).

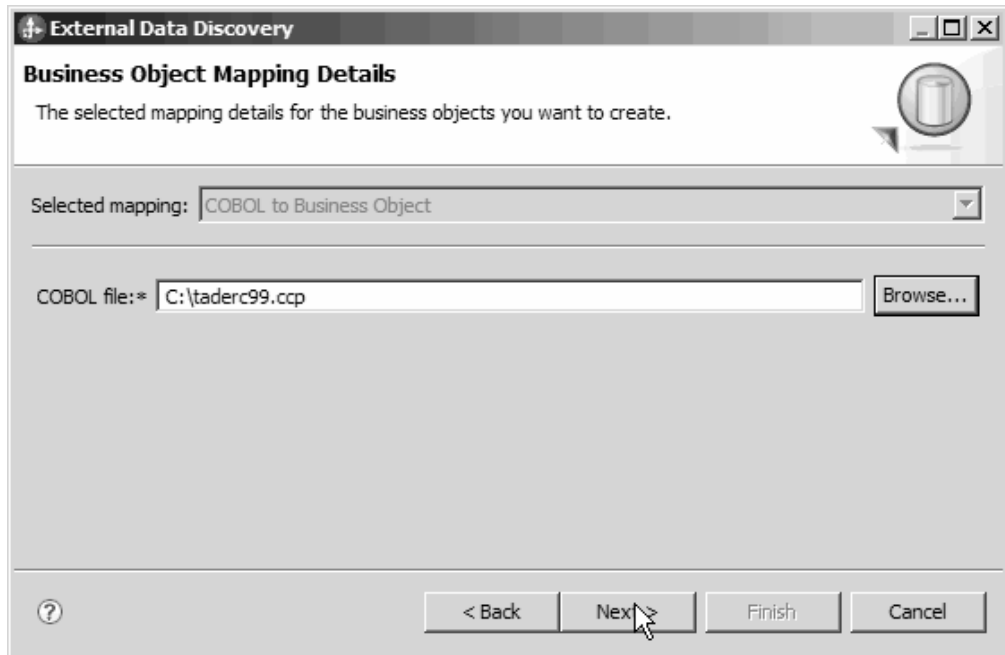


Figure 24. The Business Object Mapping Details window

- e. Click **Next**.
- f. In the Select Data Structures window, click **Find**. The new business object called DFHCOMMAREA is displayed.

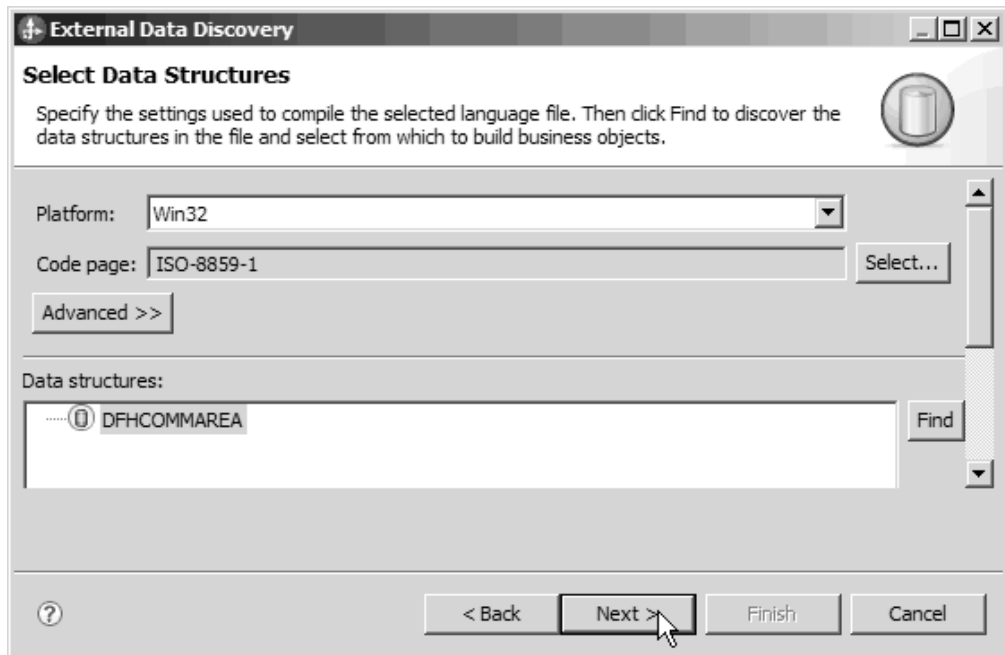


Figure 25. The Select Data Structures window

- g. Select DFHCOMMAREA and click **Next**.
- h. Click **Finish**.

A business object called DFHCOMMAREA is created in the module.

2. Optional: Generate a wrapper business object definition. Wrapper business object definitions wrap existing business object definitions with additional function. The generation of wrapper business object definitions is optional. The option to generate wrapper business object definitions is displayed only if the adapter RAR file is imported into the workspace.

Note: If you want to generate wrapper business object definitions, then you must generate them before running the external service wizard.

- a. In the Business Integration section of the window, right-click on the module and select **New** → **Business Object From External Data**.
- b. In the Business Object From External Data window, expand **Adapters**, select the adapter connector project for which you want to generate the wrapper business object .

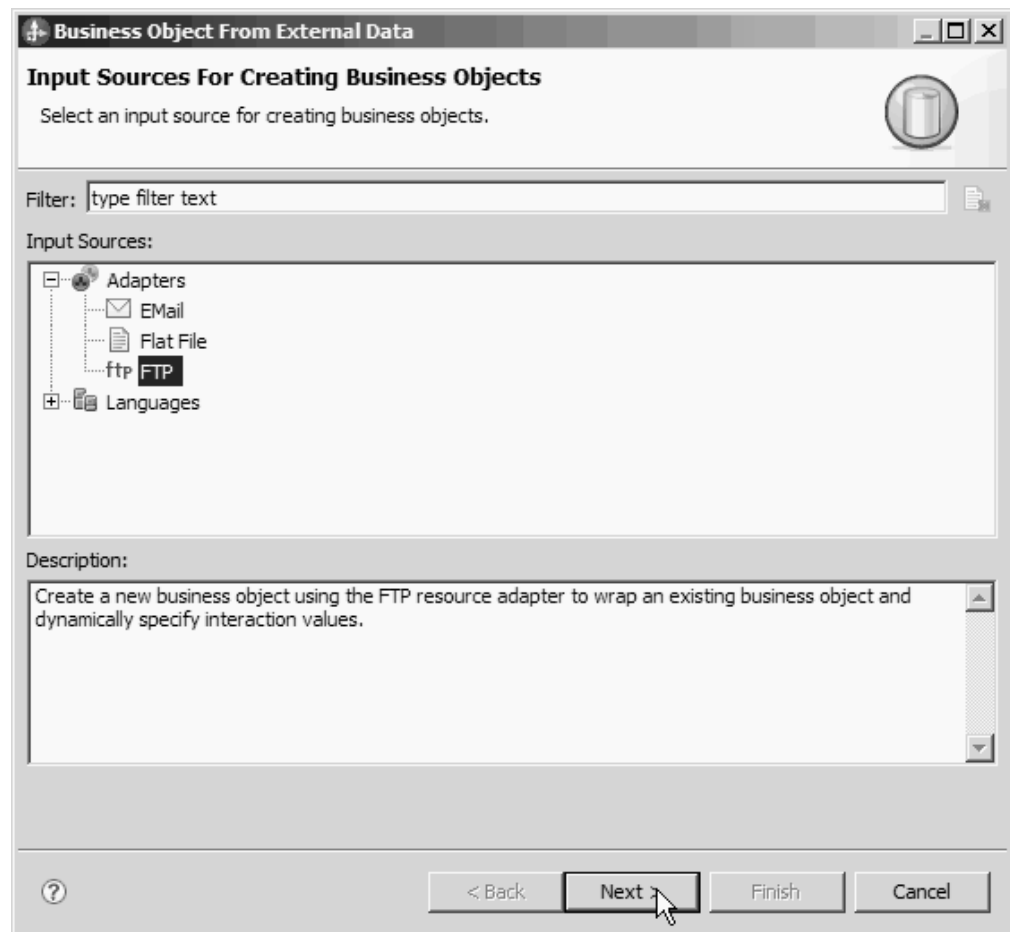


Figure 26. The Business Object From External Data window

- c. Click **Next**.
- d. In the Business Object Properties window, click **Browse** and select DFHCOMMAREA for the data type.
- e. To generate a business graph, select the **Generate business graph for each business object** checkbox.

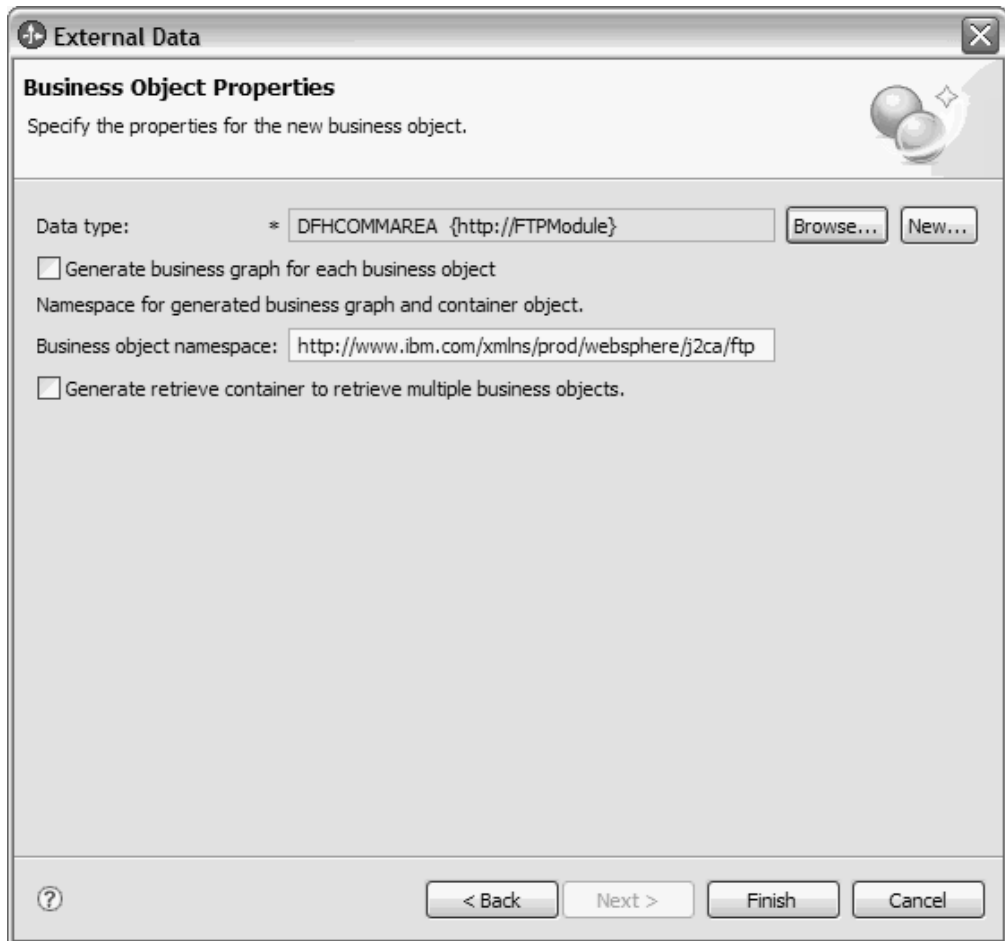


Figure 27. The Business Object Properties window

Note: Generate retrieve container to retrieve multiple business objects is applicable only for outbound retrieve operation.

f. Click **Finish**.

A wrapper business object called DFHCOMMAREAWrapper and a business graph called DFHCOMMAREAWrapperBG are listed for the current module in the Business Integration window.

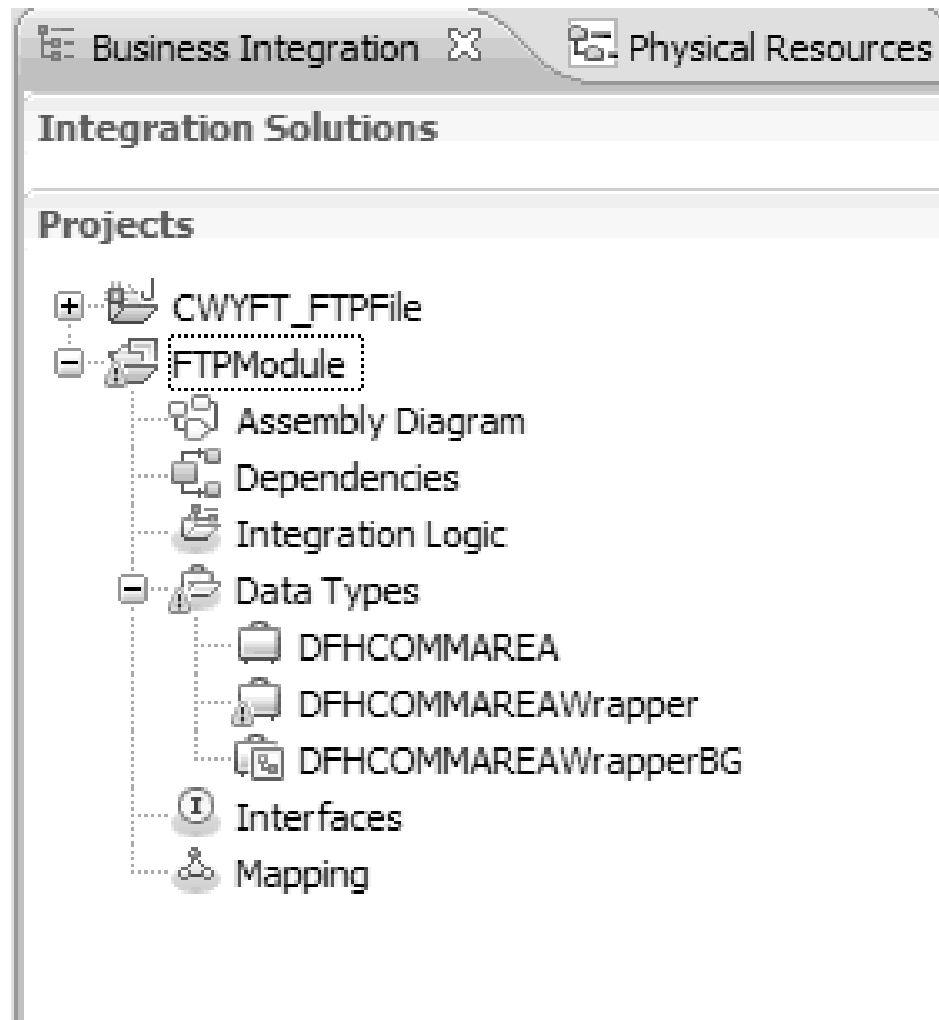


Figure 28. The wrapper business object and the business graph listed in the Business Integration window

3. Generate the required artifacts for the COBOL copybook inbound module.
 - a. In the Business Integration section of the window, right-click on the module and select **New** → **External Service**.
 - b. Select **Adapters** and click **Next**.
 - c. In the Select an Adapter window, select the FTP adapter connector project and click **Next**.

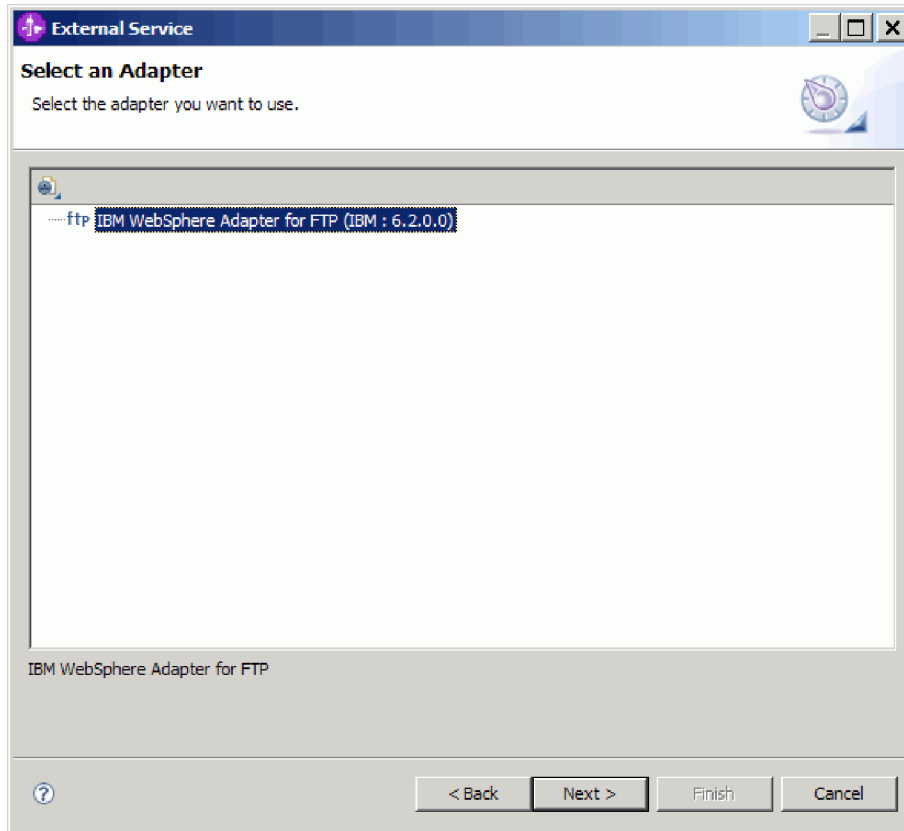


Figure 29. The Select an Adapter window

- d. In the Processing Direction window, select **Inbound** and click **Next**.
- e. Click **Browse** and select the event directory.
- f. For the **Function selector**, choose the default value.
- g. In the **Data format options** list, select **Use COBOL, C or PL/I data binding option**.

Note: This is not a data binding, but a data binding generator. The tool generates the appropriate data binding code for you in the current module.

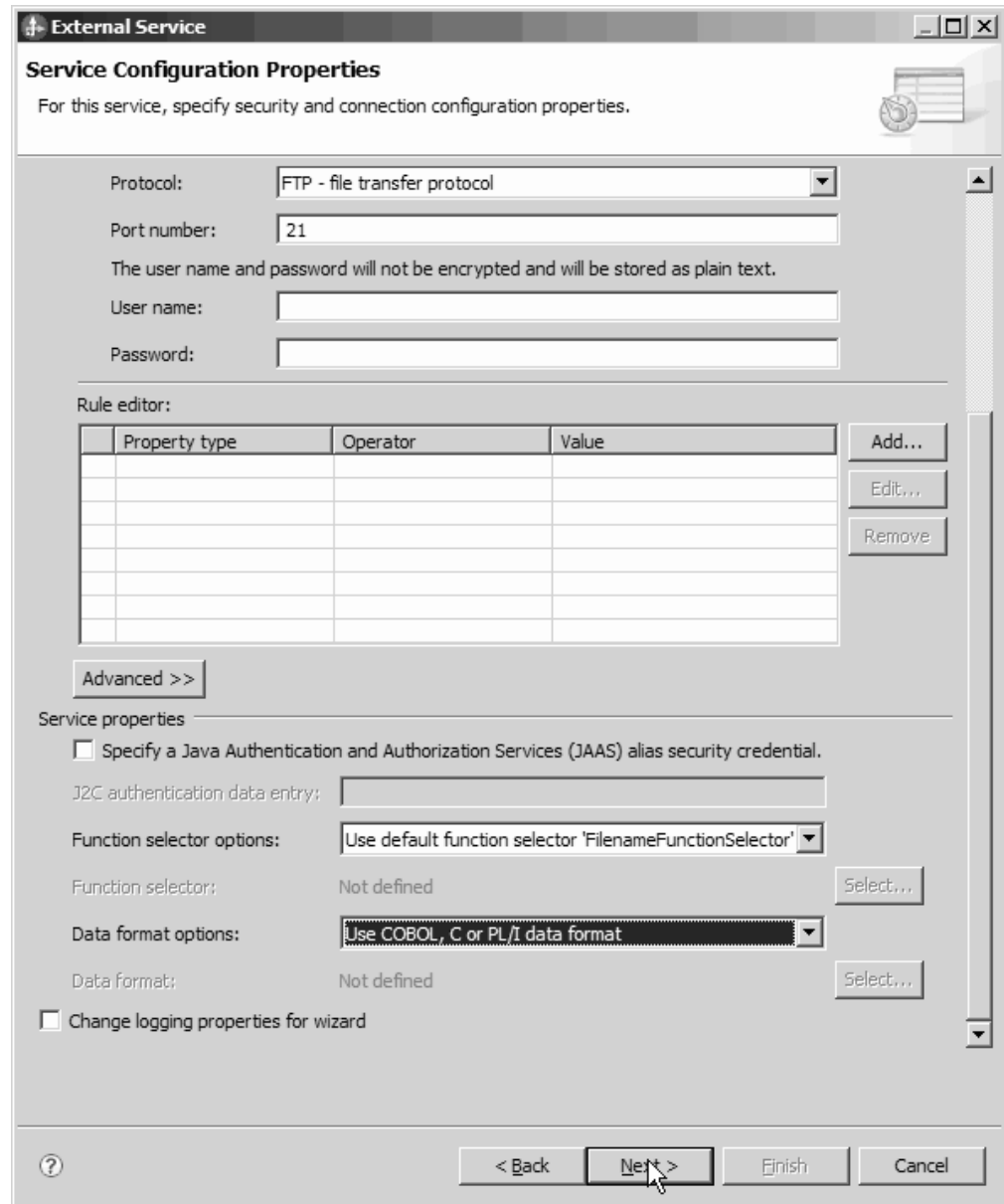


Figure 30. The Service Configuration Properties window

- h. Optional: If the input file contains multiple COBOL program source files, you can enable file splitting by size or by delimiter. To enable file splitting, click **Advanced** and then click **Advanced properties**. To enable file splitting by size, you must provide the correct length of each COBOL program source file. You can either open the business object in a text editor and add up the maximum lengths, or look for the content size of DFHCOMMAREA at the top of the file. See “Specify criteria to split file content property (SplitCriteria)” on page 209.
- i. Click **Next**.
- j. In the Operations window, click **Add**.
- k. In the Operation window, select **User defined type** for the data type. Click **Next**.

- l. For the input type, click **Browse** and select the generated business object (DFHCOMMAREA). Click **OK**.

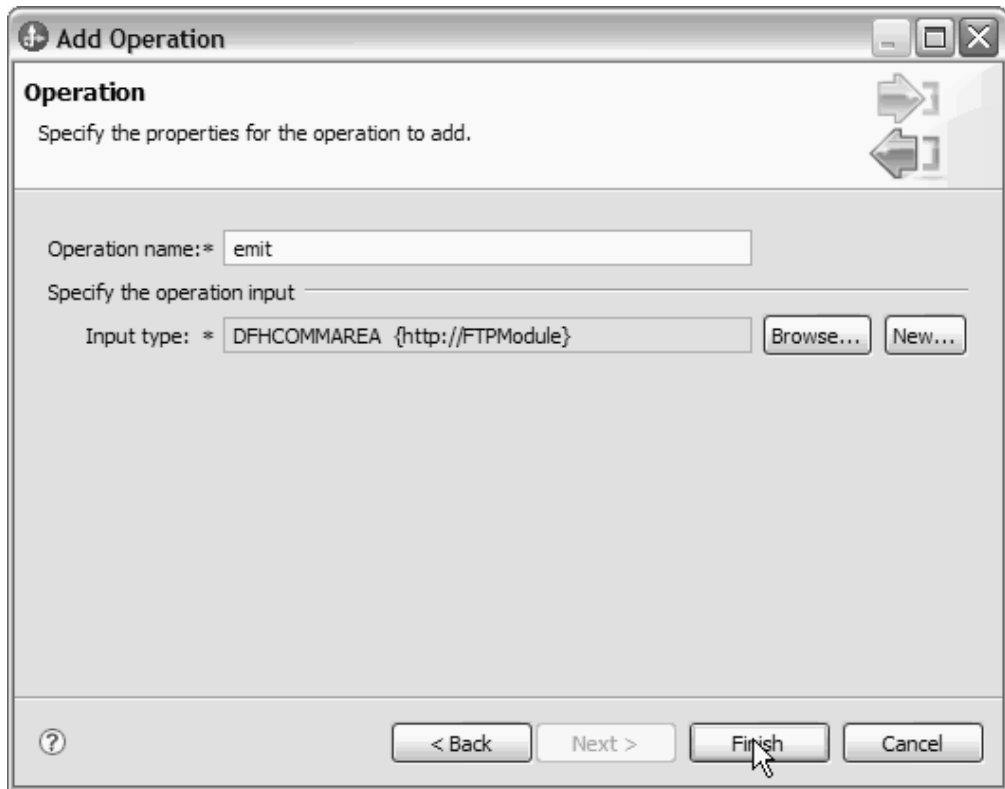


Figure 31. Selecting the input type in the Data Type Selection window

- m. Click **Next** and then **Finish**.

The data bindings used by COBOL copybook, WSDL files, export files, and other artifacts are generated. See the Project Explorer window for the generated data binding classes.

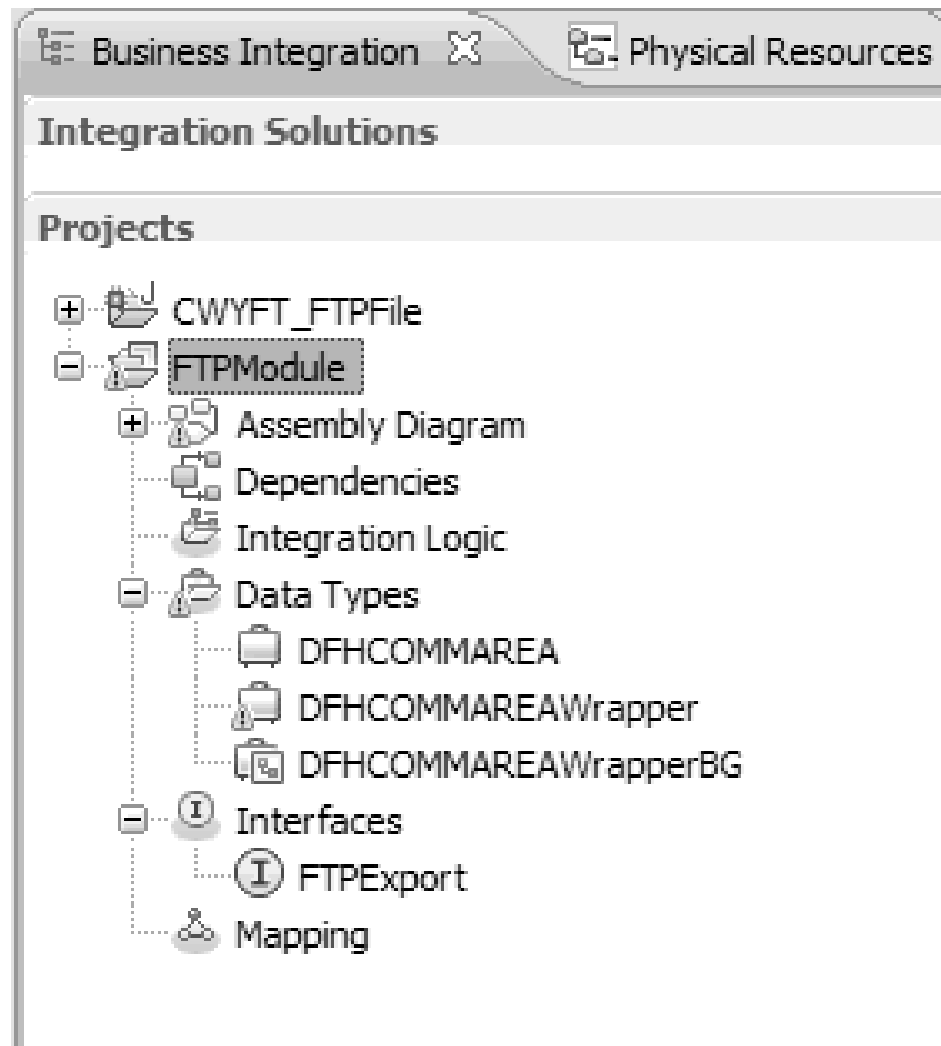


Figure 32. Data bindings used by COBOL copybook, WSDL files, export files, and other artifacts

Results

A business object, a wrapper business object, and a business graph are created for the COBOL program source file for the inbound module. Artifacts are generated for an inbound operation that uses COBOL copybook data binding. This module can be deployed on WebSphere Process Server and tested for an inbound operation.

What to do next

Deploy the module.

Defining WebSphere Application Server environment variables

Use the administrative console of the runtime environment to define WebSphere Application Server environment variables.

Before you begin

About this task

To define a WebSphere Application Server environment variable, use the following procedure.

Procedure

1. Start the administrative console of the server.
2. In the left menu, select **Environment** → **WebSphere Variables**.
3. Select the scope for the environment variable. The scope specifies the level at which the resource definition is visible on the administrative console panel. The possible values are server, node, and cell. In this example, we're choosing Cell=widCell.

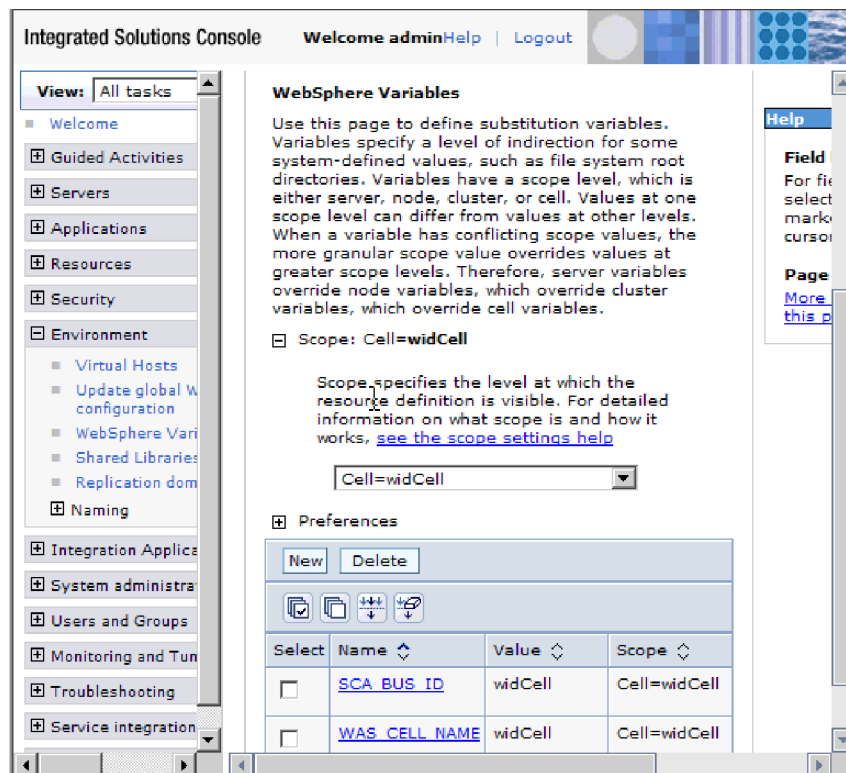


Figure 33. Setting the scope for the environment variable

4. Click **New** and provide a name and a value for the environment variable. The name is the symbolic name that represents a physical path. The value is the absolute path that the variable represents. In this example, the name is `EVENT_DIRECTORY` and the value is `/home/user/event`. You can use the **Description** field, which is optional, to describe the purpose of the variable.

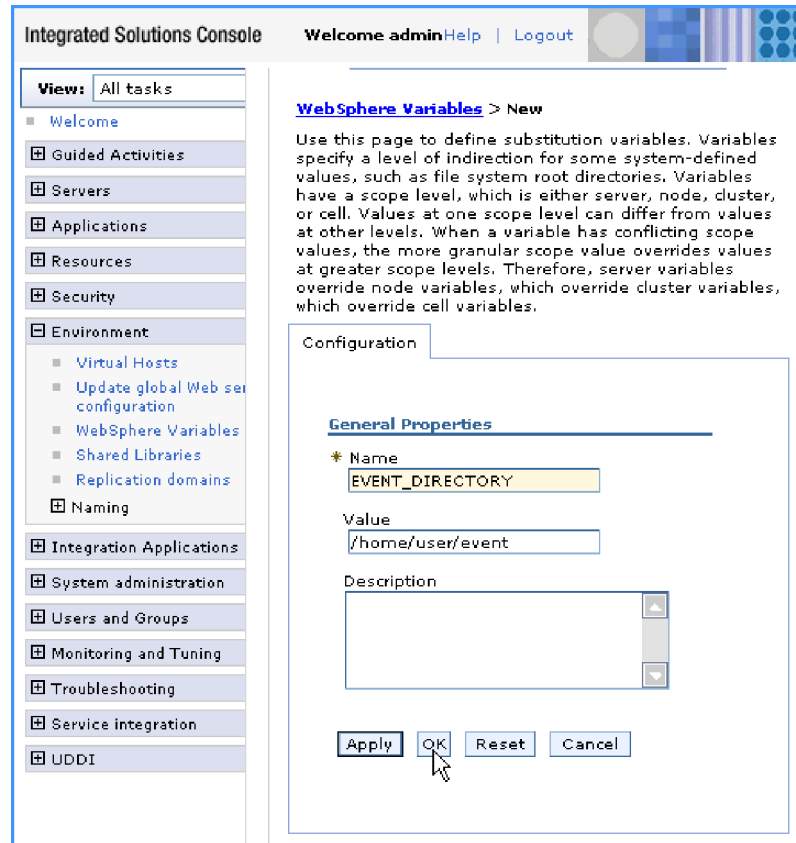


Figure 34. Providing a name and a value for the environment variable

5. Click **OK** and save the changes.

Results

An environment variable called `EVENT_DIRECTORY` is created, with the value `/home/user/event` and a scope of `Cell=widCell`. You can now use it in the external service wizard whenever you need to specify the event directory.



Figure 35. The new environment variable `EVENT_DIRECTORY` displayed in the WebSphere Variables window

What to do next

Create a project, which is used to organize the files associated with the adapter.

Creating a simple service with the adapter pattern wizard

Adapter patterns provide a quick and easy way of creating a simple service with an adapter.

Before you begin

A module has already been created called `RetrieveAFileModule` and a business object called `Customer` has already been created. If you are using WebSphere Application Server environment variables to specify local files and directories, you have defined them using the WebSphere Process Server administrative console.

About this task

The following adapter patterns are available for the adapter for FTP:

Table 6.

Adapter pattern	Description
Inbound FTP pattern	The FTP inbound pattern creates a service that retrieves a file in a specific directory on an FTP server. If the file is not in an XML format, you can specify a data handler that will transform from the file content format to business objects. The file content can be split if the content contains multiple copies of the data structure for processing.
Outbound FTP pattern	The FTP outbound pattern creates a service that stores data in a file in a specific directory on an FTP server. If the required output format is not an XML format, you can specify a data handler that will transform the business object to the file content format.

In this example, you create a FTP inbound service that receives a file from the file system for processing. The completed service in this example will read in a file and split the contents into separate files based on a delimiter.

Complete the following steps to create a service with the adapter pattern wizard:

Procedure

1. Right-click **RetrieveAFileModule** within the **Business Integration** section of the WebSphere Integration Developer window and select **New** → **From Patterns**. The New from Pattern window opens.
2. Select **Create an inbound FTP service to read from a remote file** and click **Next**.

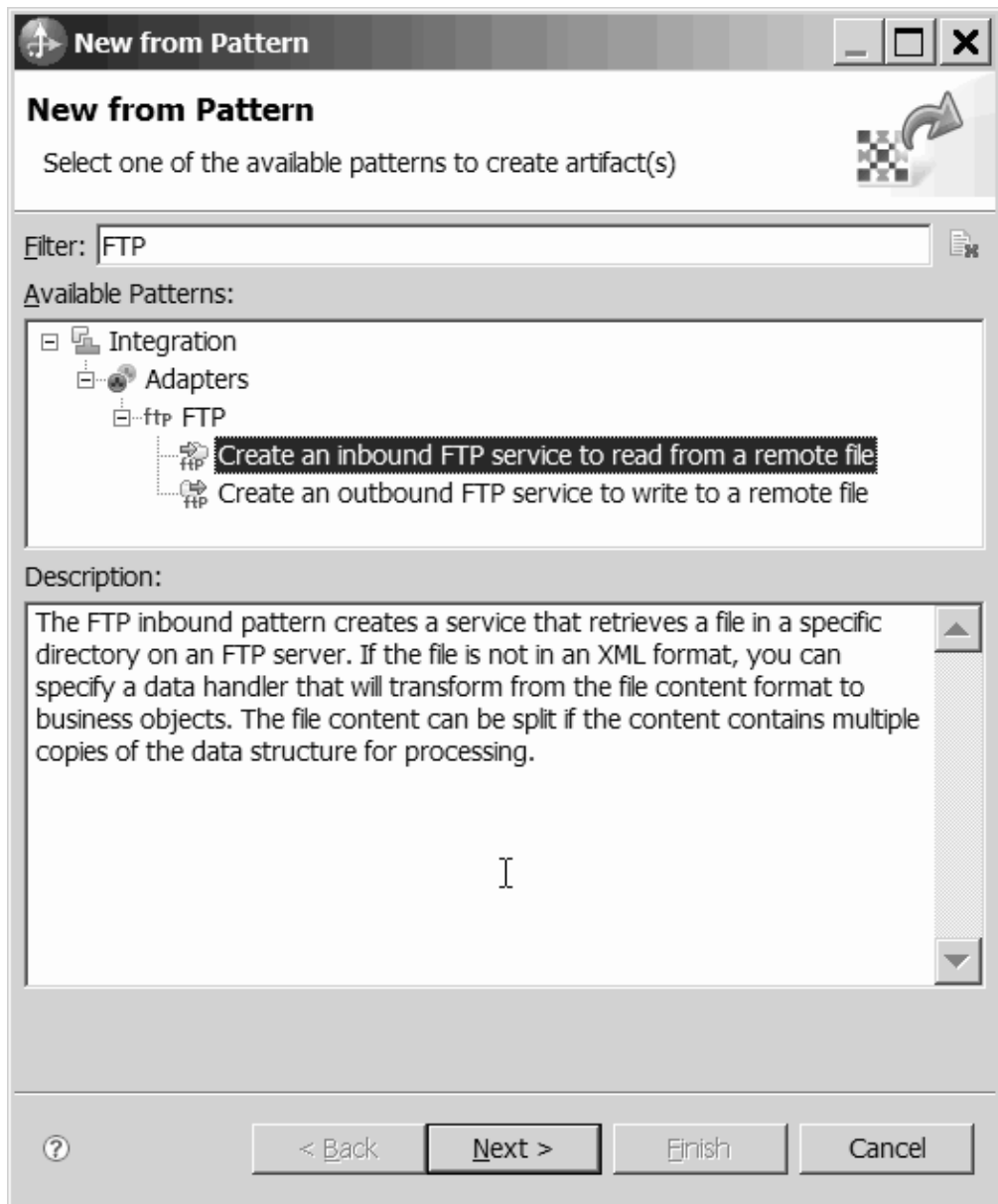


Figure 36. New from Pattern window

3. In the New Inbound FTP Service window, change the name to something meaningful such as FTPInboundInterface and click **Next**.

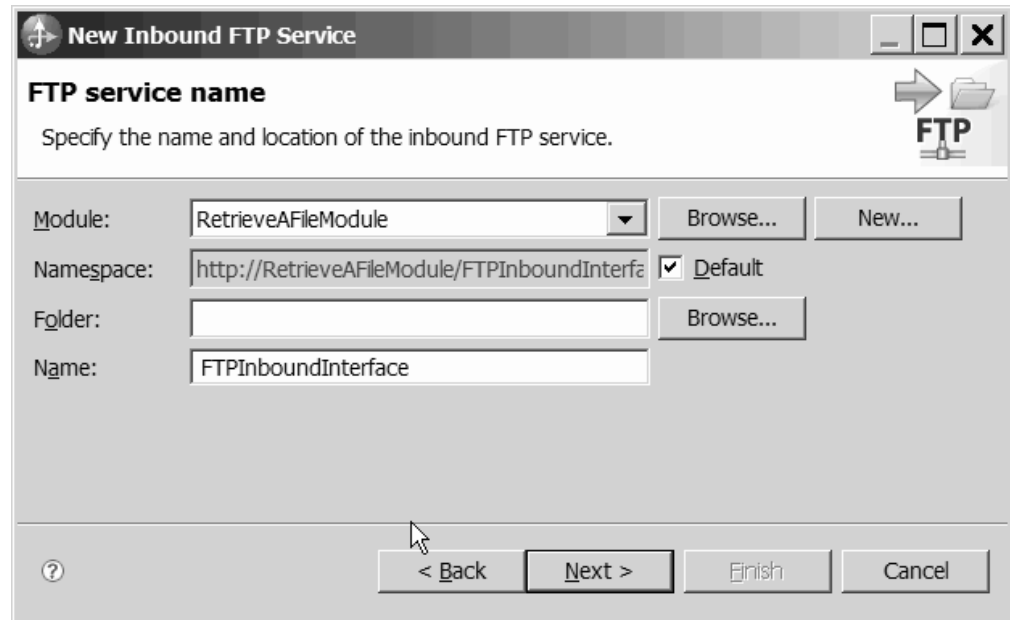


Figure 37. FTP service name window

4. In the Business object and location window, click **Browse** and navigate to the **Customer** business object.
5. Specify the directory where you placed the input file, in this case the `/home/user/event` directory, and click **Next**. To use a WebSphere Application Server environment variable for this value, specify the name of the variable in braces, preceded by a `$` symbol. For example: `${FTPINBOUNDEVENTS}`.

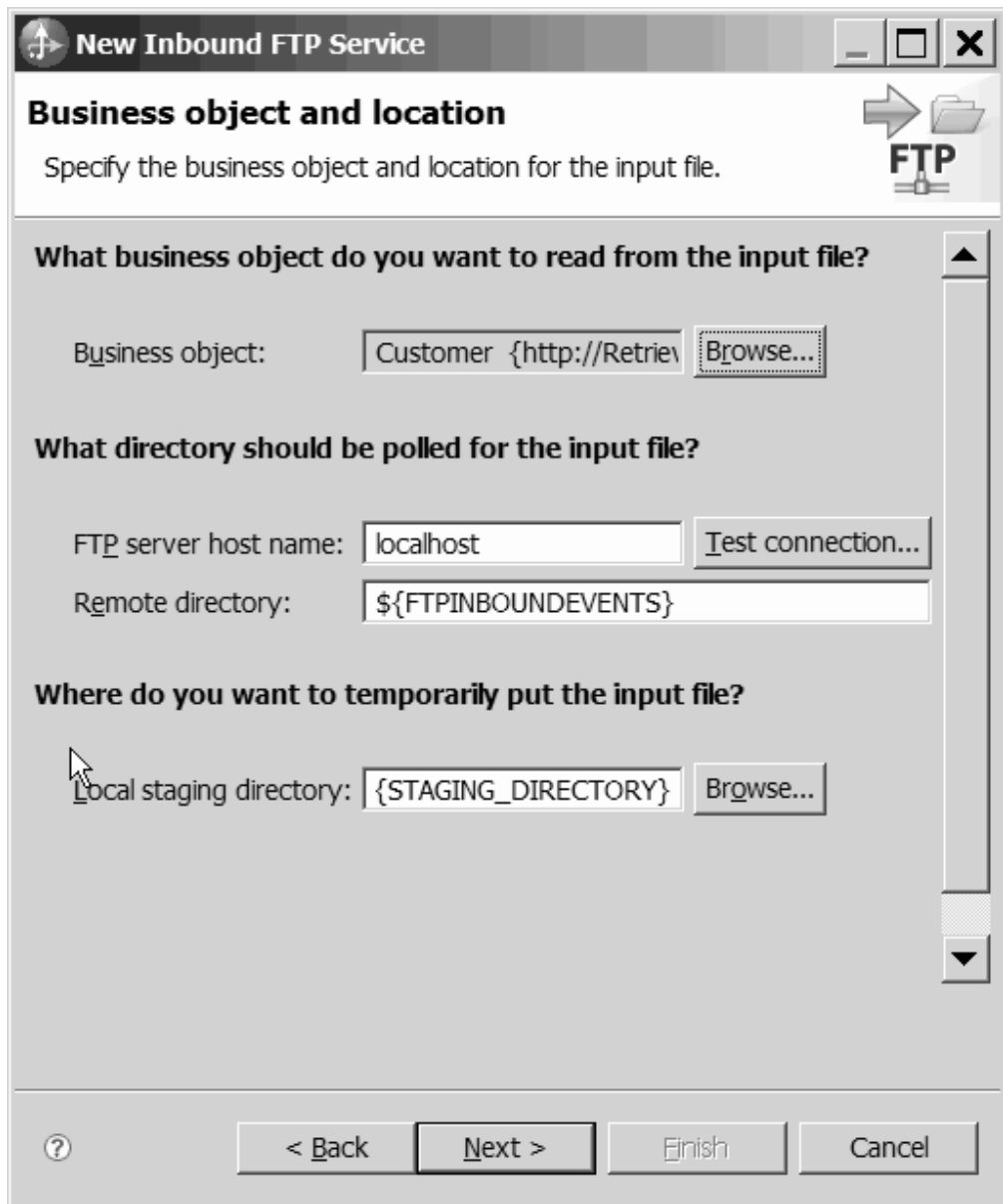


Figure 38. Business object and location window

6. In the FTP server security credential window, select either **Using an existing JAAS alias** or **Using user name and password** and click **Next**.

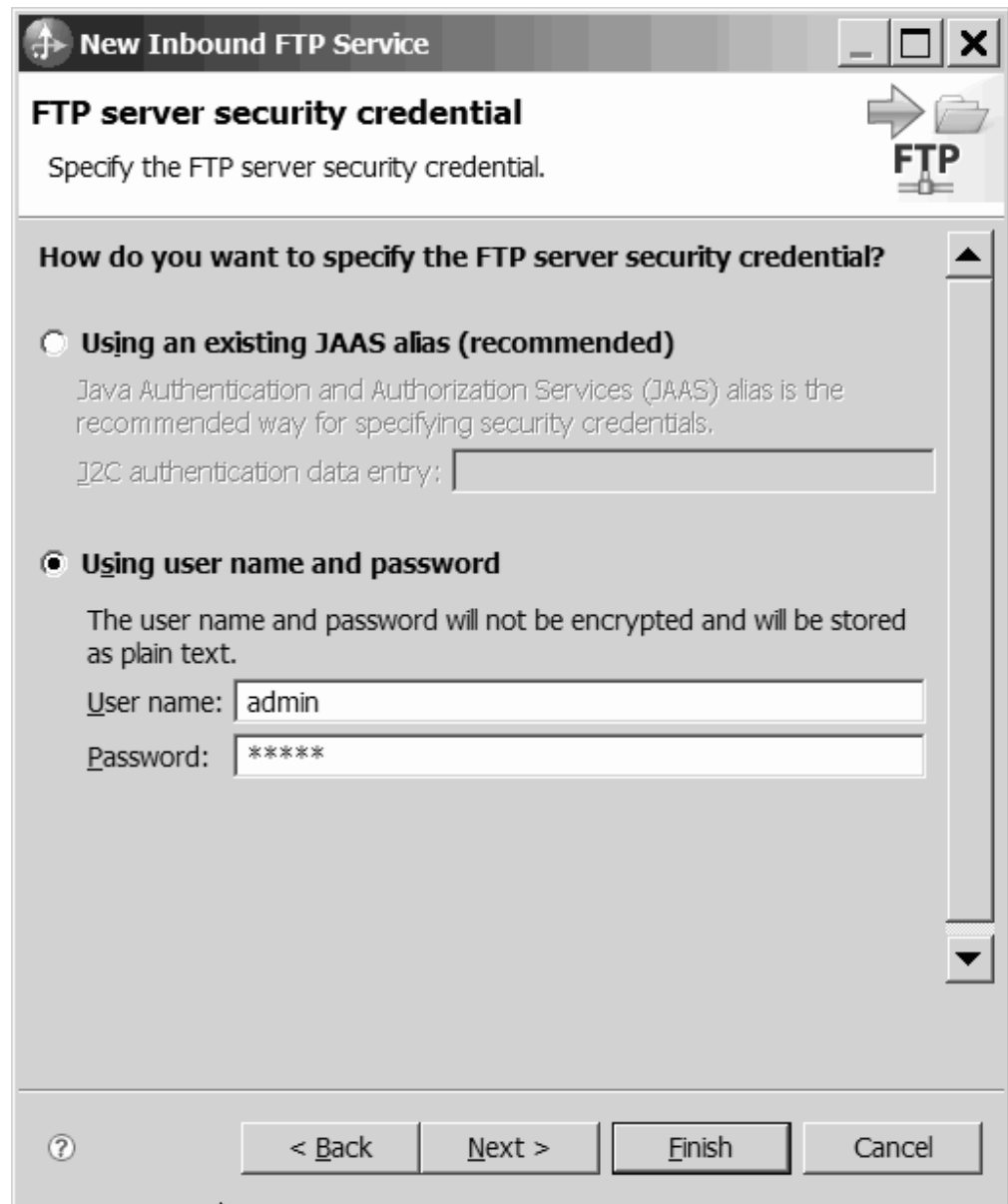


Figure 39. FTP server security credential window

7. In the Input file format and file content split option window, accept the default XML input file format or select **Other** and specify a data handler to transform the data from your native format to the business object format.
8. Select **Split file content by delimiter** and enter your delimiter, which is `###;\n` in this example. Click Next.

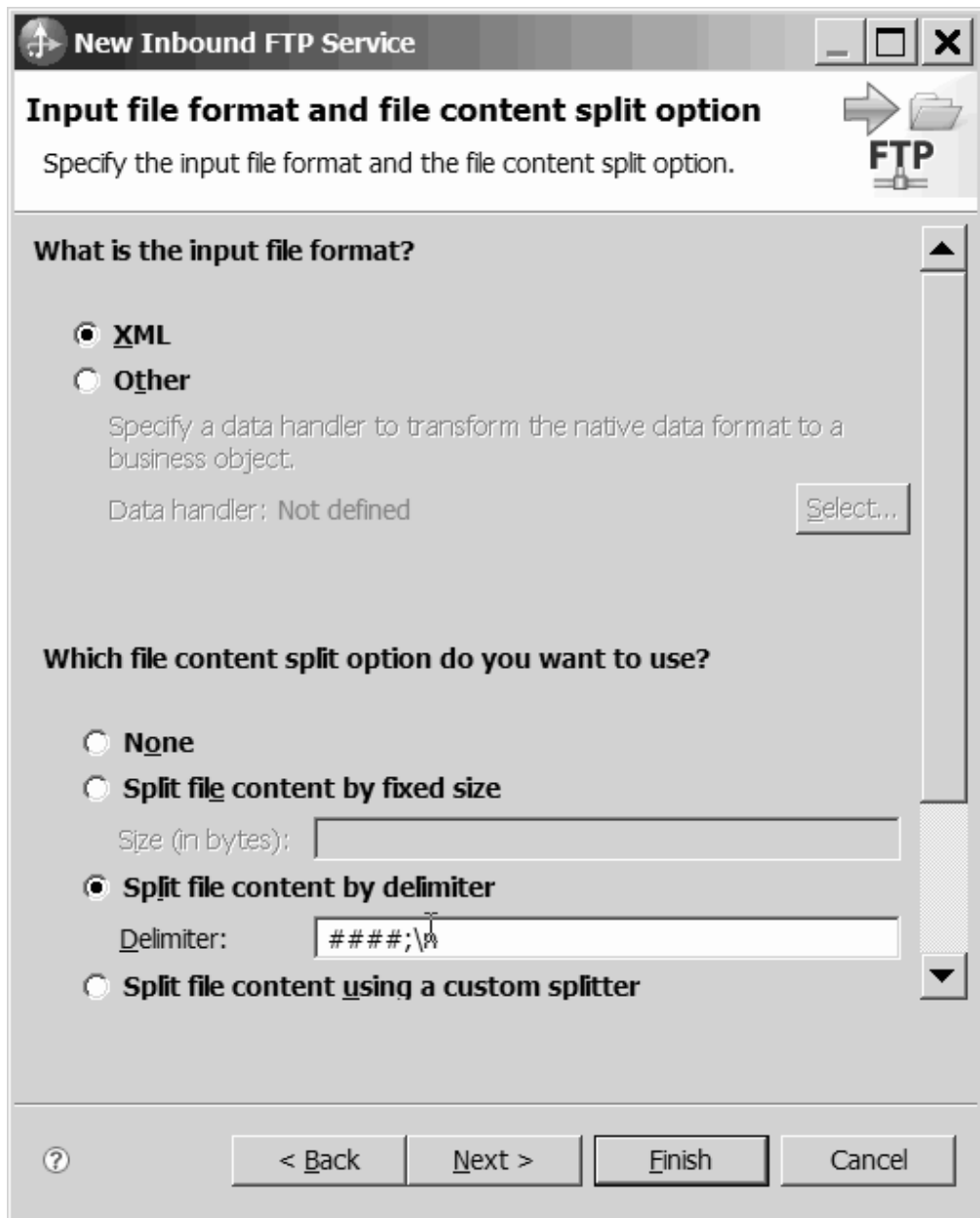


Figure 40. Input file format and file content split option window

9. In the Archive directory and wrapper business object window, specify the **Local archive directory**, which is FTP\inboundarchive in this example. To use a WebSphere Application Server environment variable for this value, specify the name of the variable in braces, preceded by a \$ symbol. For example: `${FTPINBOUNDARCHIVE}`. Select **Use a wrapper business object to contain additional input file information** if you want to include the adapter-specific information. Click **Finish**.

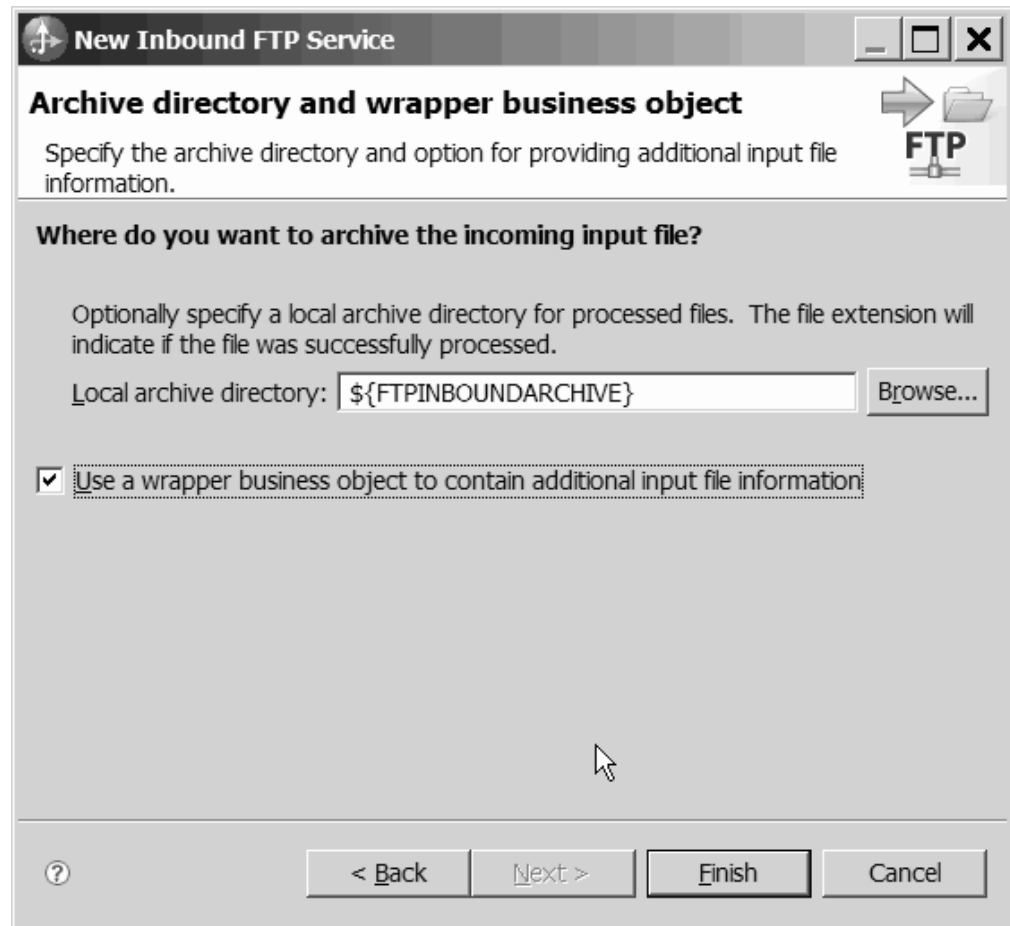


Figure 41. Archive directory and wrapper business object window

Results

The inbound service is created, which includes the following artifacts:

Table 7.

Artifact	Name	Description
Export	FTPInboundInterface	The export exposes the module externally, in this case, to the WebSphere Adapter for FTP.
Business objects	Customer, CustomerWrapper	The Customer business object contains the fields for customer data such as name, address, city and state. The CustomerWrapper business object contains additional fields for adapter-specific information.
Interface	FTPInboundInterface	This interface contains the operation that can be invoked.

Table 7. (continued)

Artifact	Name	Description
Operation	emitCustomerInput	emitCustomerInput is the only operation in the interface.

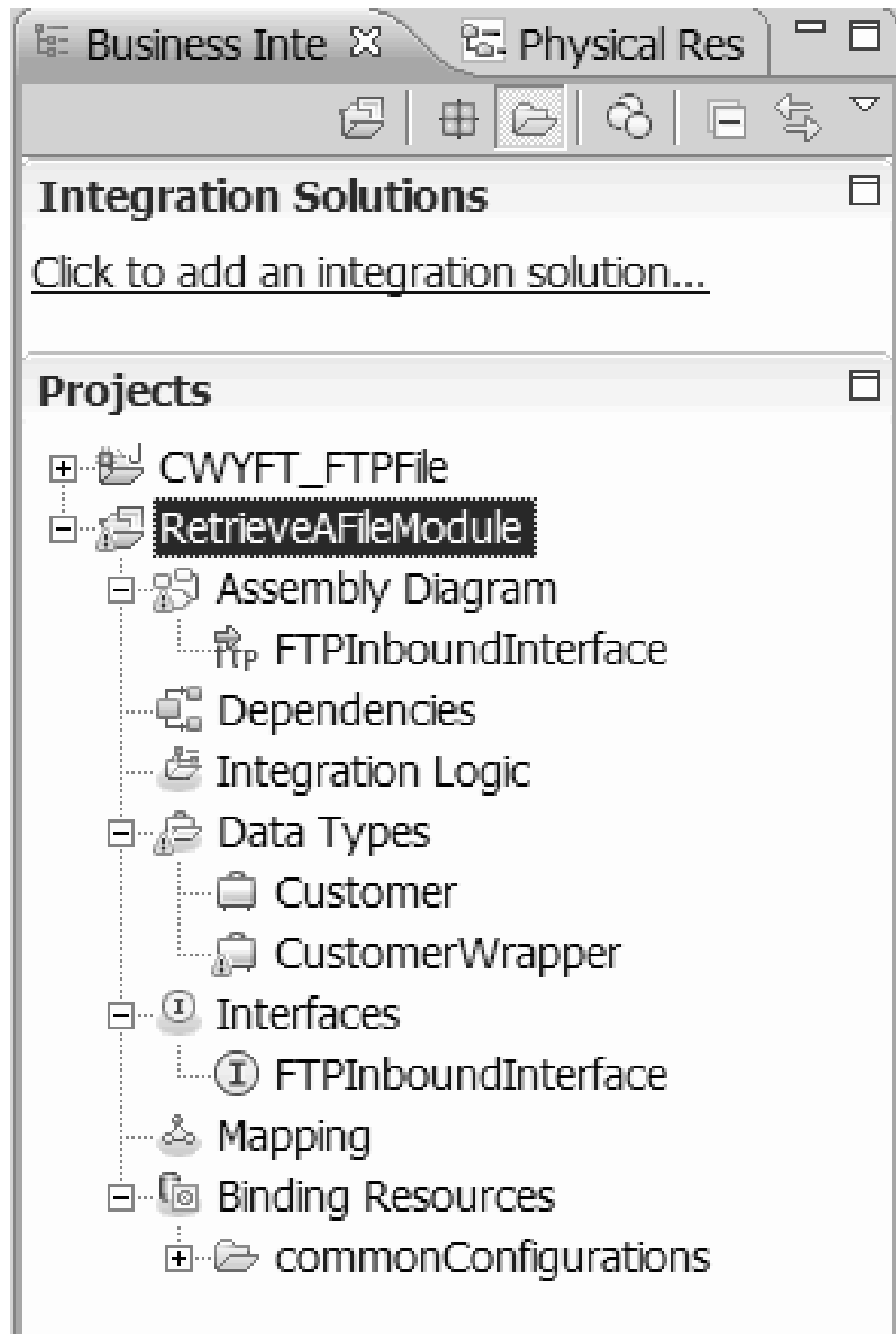


Figure 42. The **Business Integration** section of the WebSphere Integration Developer window with the new artifacts

Starting the external service wizard

To begin the process of creating and deploying a module, you start the external service wizard in WebSphere Integration Developer. The wizard creates a project that is used to organize the files associated with the module.

Before you begin

Make sure you have gathered the information you need to establish a connection to the FTP server. For example, you need the name or IP address of the FTP server and the user ID and password needed to access it.

About this task

Start the external service wizard to create a project for the adapter in WebSphere Integration Developer. If you have an existing project, you can select it instead of having the wizard create one.

To start the external service wizard and create a project, use the following procedure.

Procedure

1. To start the external service wizard, go to the Business Integration perspective of WebSphere Integration Developer, and then click **File** → **New** → **External Service**.
2. In the External service window, expand **Adapters** and select **FTP**, which is the type of adapter you want to create. Then click **Next**.
3. In the Select an Adapter window, select the adapter name to create a new project, or select an existing project to reuse it.
 - To create a project, perform the following steps:
 - a. Select **IBM WebSphere Adapter for FTP (IBM : 6.2.0.0)** and click **Next**.
 - b. In the Adapter Import window, provide details about the project you want to create.
 - 1) In the **Connector project** field, optionally specify a different name for the project.
 - 2) In the **Target runtime** field, select the server (for example, **WebSphere Process Server v6.2**).
 - 3) Click **Next**.
 - To select an existing project, select the project folder under **IBM WebSphere Adapter for FTP (IBM : 6.2.0.0)** and then click **Next**.

Results

For a new project, the project is created and is listed in the Business Integration perspective. The wizard will create adapter artifacts in the specified project.

Configuring the module for outbound processing

To configure a module to use the adapter for outbound processing, use the external service wizard in WebSphere Integration Developer to build business services, specify data transformation processing, and generate business object definitions and related artifacts.

Setting deployment and runtime properties

Specify deployment and runtime properties that the external service wizard uses to connect to the FTP server.

Before you begin

Before you can specify the connection properties, you must have started the external service wizard.

About this task

The external service wizard requires this information to connect to the database for discovery and for creating the service description.

Procedure

1. In the Processing Direction window, select **Outbound** and click **Next**.
2. In the **Deploy connector project** field, specify whether to include the adapter files in the module. Choose one of the following options:
 - **With module for use by single application**

With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or when multiple modules need to run different versions of the adapter. By using an embedded adapter, you can upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
 - **On server for use by multiple applications**

If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
3. Define the following FTP system connection information for your module. For more information, refer to the Managed (J2C) connection factory properties topic.
 - **Host name** - Specifies the host name of the FTP server.
 - **Directory** - Specifies the output directory on the FTP server.
 - **Port number** - Specifies the port number of the FTP server.
 - **Protocol** - Specifies either normal FTP, secure FTP (FTPS), or FTP over SSH (SFTP). If you select SFTP, the port number will default to 22.
 - **User name** - Specifies the name of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see “User name property (UserName)” on page 211.
 - **Password** - Specifies the password of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see “Password property (Password)” on page 202.

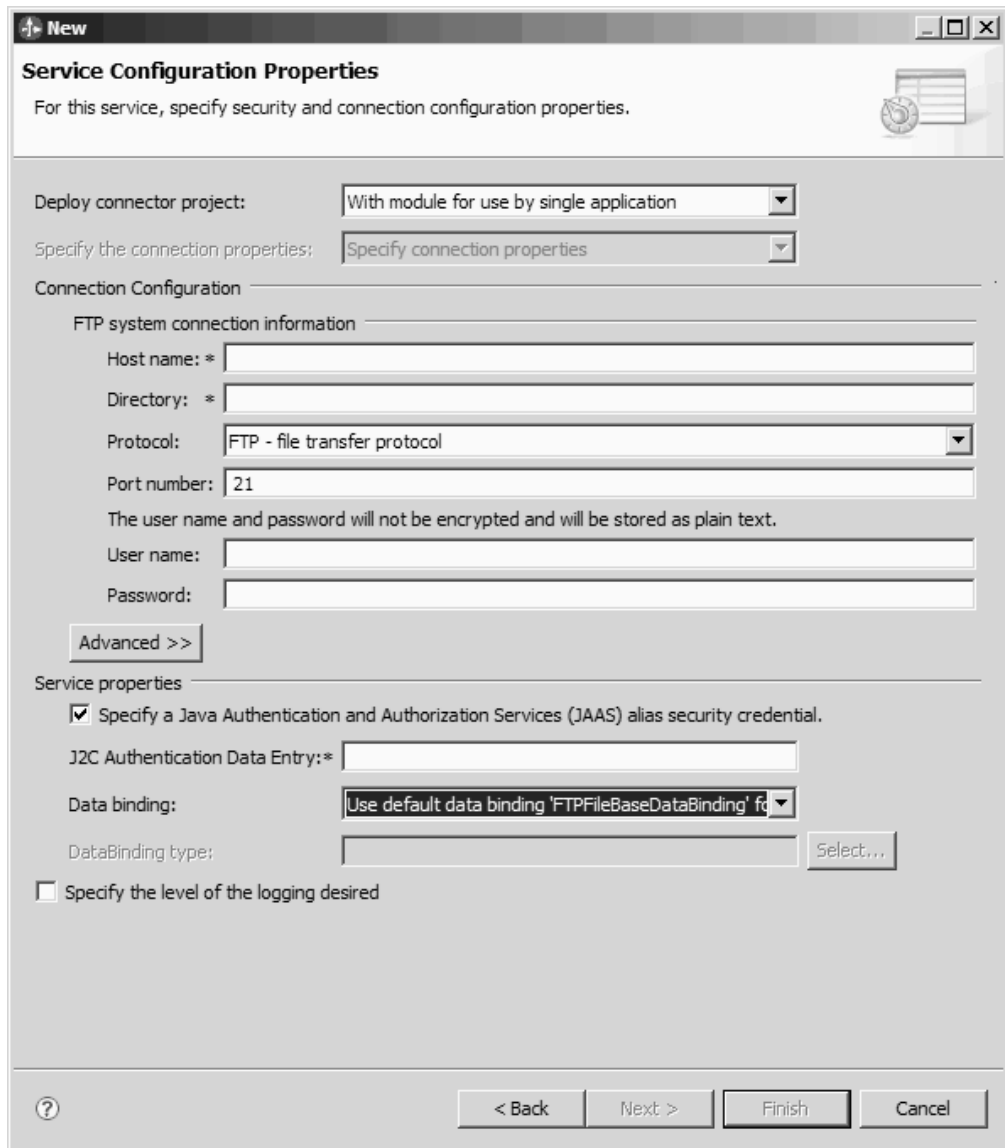


Figure 43. The Service Configuration Properties window

4. Optional: Click **Advanced** to specify additional properties, such as those that control working with a second FTP server, bidi formatting, a staging directory, logging and tracing, secure connection and sequence file selection. For more information, refer to the Managed (J2C) connection factory properties topic.
 - a. If you have multiple instances of the adapter, expand **Logging and tracing** and set **Adapter ID** to a value that is unique for this instance. For more information about this property, see “Resource adapter properties” on page 158.
5. Optional: In the Service properties section of the window, specify a Java Authentication and Authorization Services (JAAS) alias for the adapter to use at run time. This is the authentication alias that you set up on the FTP server. The name is case-sensitive. For more details, see “Creating the authentication alias”.
6. In the **Data binding** field, select one of the following:
 - **Use default data binding 'FTPFileBaseDataBinding' for all operations**
A non-configured data binding for all of the operations used in the service.

- **Use a data binding configuration for all operations**
A configured data binding that will be used for all of the operations used in the service.
 - **Specify a data binding for each operation**
No default binding is specified. You will select a specific data binding for each operation used in the service.
7. Optional: Select the **Specify the level of the logging desired** check box if you want to specify the log file output location or define the level of logging for this module. For information about logging levels, see the section on configuring logging properties in the Troubleshooting and support topic.

Results

The external service wizard now has the information it needs to connect to the FTP server.

If you selected either the **Use default data binding 'FTPFileBaseDataBinding' for all operations** or **Specify a data binding for each operation** Data binding configuration options, click **Next** to continue to work in the wizard to select a data type and to name the operation associated with the data type.

If you selected the **Use a data binding configuration for all operations** Data binding configuration option, proceed to “Configuring the data binding” on page 83.

Selecting a data type and operation name

Use the external service wizard to select a data type and to name the operation associated with the data type. For outbound communications, the external service wizard gives you the choice of three different data types: user defined type, generic FTP business object, and generic FTP business object with business graph. Each data type corresponds to a business object structure.

Before you begin

Before you can perform the following steps, you must have specified the connection properties for the adapter to connect to the FTP server.

About this task

To select a data type and name the operation associated with it, follow this procedure.

Procedure

1. In the Operations window, click **Add**.

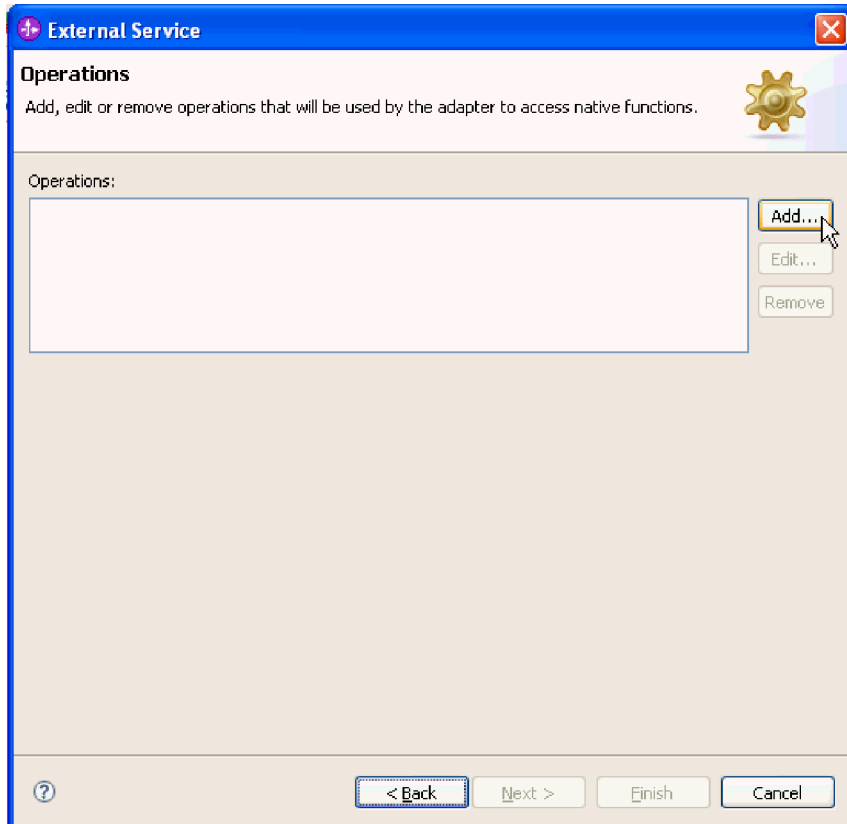


Figure 44. Adding an operation

2. In the Add Operations window, select an operation from the **Operation kind** list and a data type from **The data type for the operation input** list, and click **Next**. If you select **User defined type**, you must provide a user-defined data binding to support it. The data bindings provided by the **Generic FTP business object** only support generic input types for the supported operations.

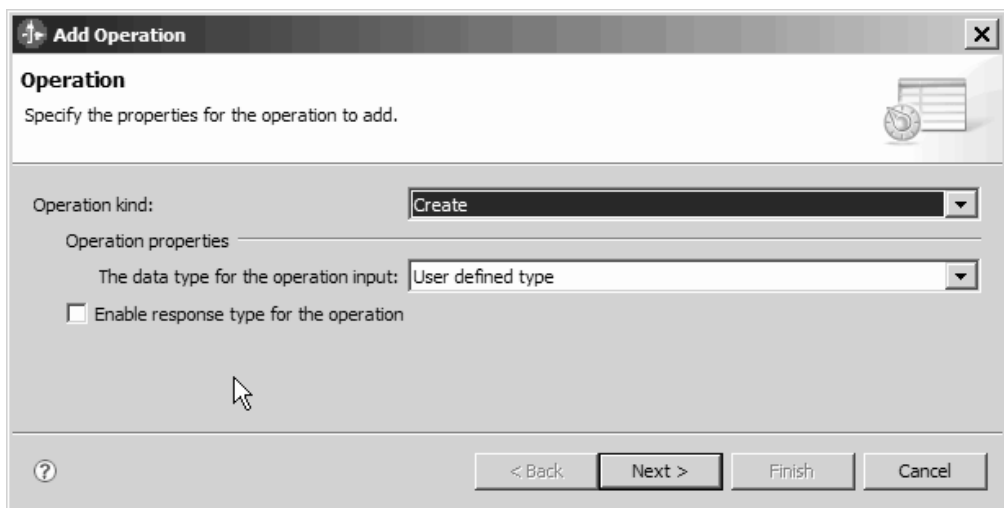


Figure 45. Selecting a data type

3. Optional: To have the file name returned or to have True or False returned during Delete and ServerToServer operations, select the **Enable response type**

for the **operation** check box. For Exists, List and Retrieve operations, a response type is required, and by default the **Enable response type for the operation** check box is selected. Click **Next**.

4. In the Operation window, type a name for the operation in the **Operation name** field. Name the operation something meaningful. If this module is going to be used to create a new customer record, name it something like createCustomer. For more information on the types of operations the adapter can perform, see the Table 1 on page 4 topic.

Note: Names cannot contain spaces.

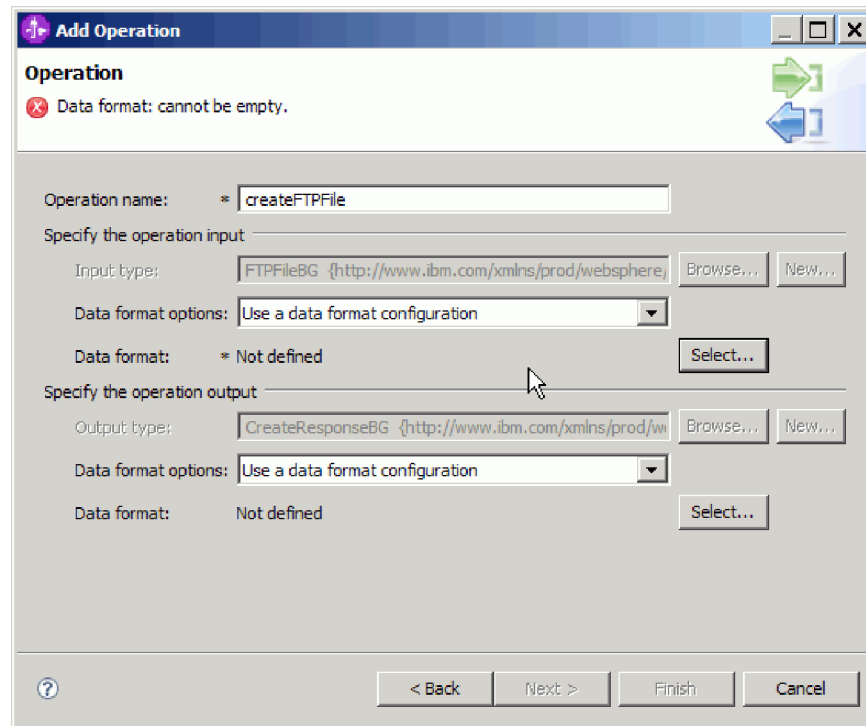


Figure 46. Naming the operation

Results

A data type is defined for the module and the operation associated with this data type is named.

If you chose to use a configured data binding, continue to work in the current external service wizard window, to add and configure a data binding to be used with the module.

If you chose to use the default data binding, proceed to “Setting interaction specification properties and generating the service” on page 89.

Configuring the data binding

Each data type has an equivalent data binding that is used to read the fields in a business object and fill the corresponding fields in a file. In the external service wizard, you add a data binding to your module and configure it to correspond with your data type. This way, the adapter knows how to populate the fields in a file with information it receives in the business object.

Before you begin

You must have selected a data type and chosen a configuration name to be associated with the data type.

Note: Data bindings can be configured prior to running the external service wizard using WebSphere Integration Developer. To do this, select **New** → **Resource configuration** in WebSphere Integration Developer and complete the data binding windows described in this documentation.

About this task

To add and configure a data binding for the module, follow this procedure.

Procedure

1. In the **Specify the operation input** area, select **Use a data format configuration** from the **Data format options** drop down list.
2. Click **Select**.

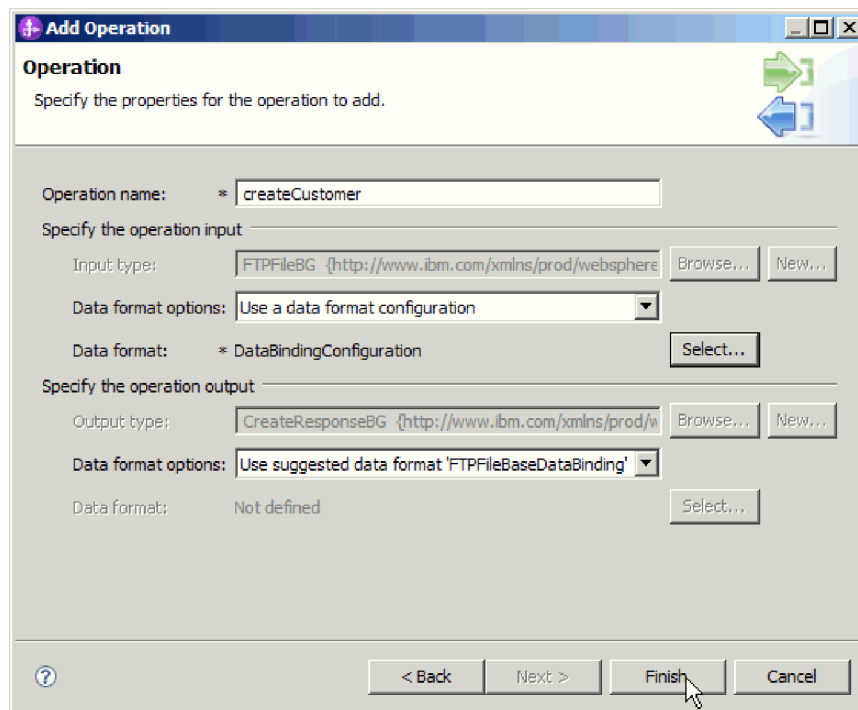


Figure 47. Operation window

3. In the **Specify the operation output** area, select **FTPFileBaseDataBinding** from the **Data format options** drop down list.
4. Click **Next**.
5. Specify the Configuration name for the Data Transformation Configuration property and click **Finish**.

Results

A data binding is configured for use with the module.

What to do next

From the current external service wizard window, you will proceed to select a data handler for the module.

Configuring data handlers

When you select a data type that contains business objects, you need to specify the data handler that will perform the conversions between a business object and a native format.

Before you begin

You must have created a data binding before specifying the data handlers for the module.

About this task

To specify data handlers, follow this procedure.

Note: Data handlers can be configured prior to running the external service wizard using WebSphere Integration Developer. To do this, select **New** → **Binding Resource Configuration** in WebSphere Integration Developer and complete the data handler windows described in this documentation.

Procedure

1. In the New Data Transformation configuration window, click **New**.
2. Enter the name for the data handler configuration in the **Name** field (this example uses `DataBindingConfiguration`). You do this the first time when you set the data handler. To use the same data handler later, click **Browse** and select it.

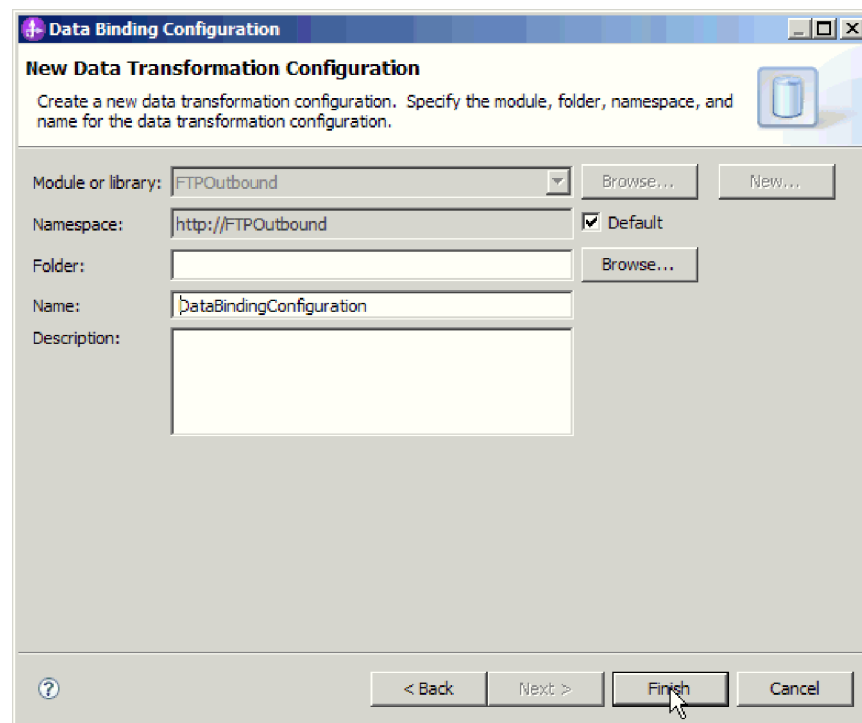


Figure 48. Specifying a name for the data handler configuration

3. Click **Next**.
4. In the Data Transformation Properties window, click the drop down list next to the **Binding Type** property. Two choices are provided: DataBinding and DataHandler. To use a data binding developed for an earlier version of the adapter, select DataBinding. To configure a new data handler, select DataHandler. Click **Select** to create a new data handler configuration.

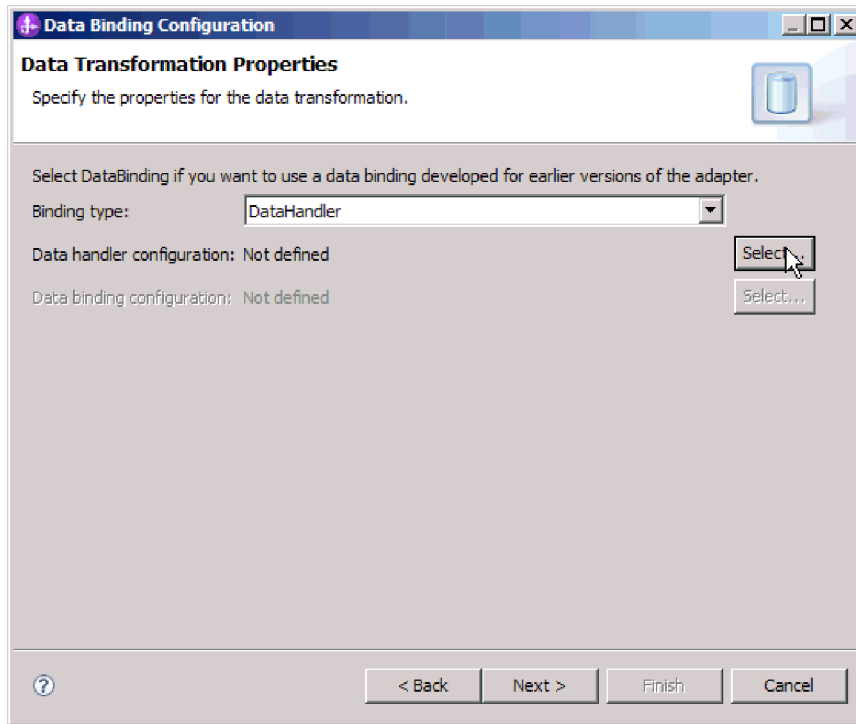


Figure 49. Specifying the Binding Type in the Data Transformation window

5. Click **Next**.
6. Click **New** on the New Data Transformation Configuration window; specify the Module, Namespace, Folder, and Name for the data handler configuration. Click **Next**.

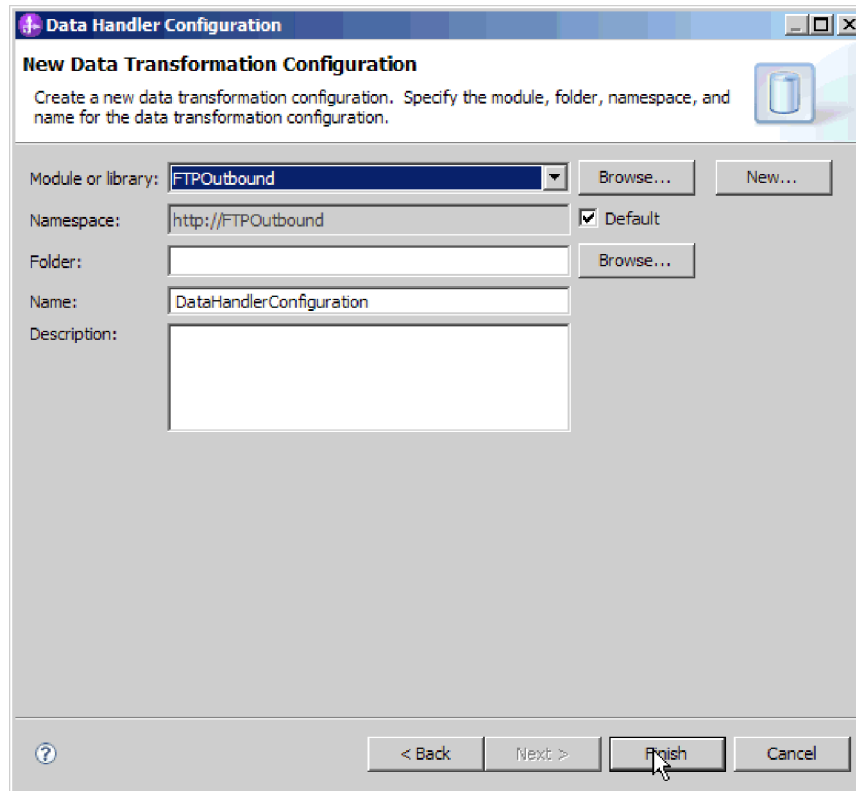


Figure 50. Specifying the details for the Data handler configuration

7. Choose the class name for the data handler. In the Select Data Format Transformation window, click **Use existing data format transformation** from the list option. A list of available data handler classes is displayed. Select the data handler class (this example uses XML data handler). Click **Next**.

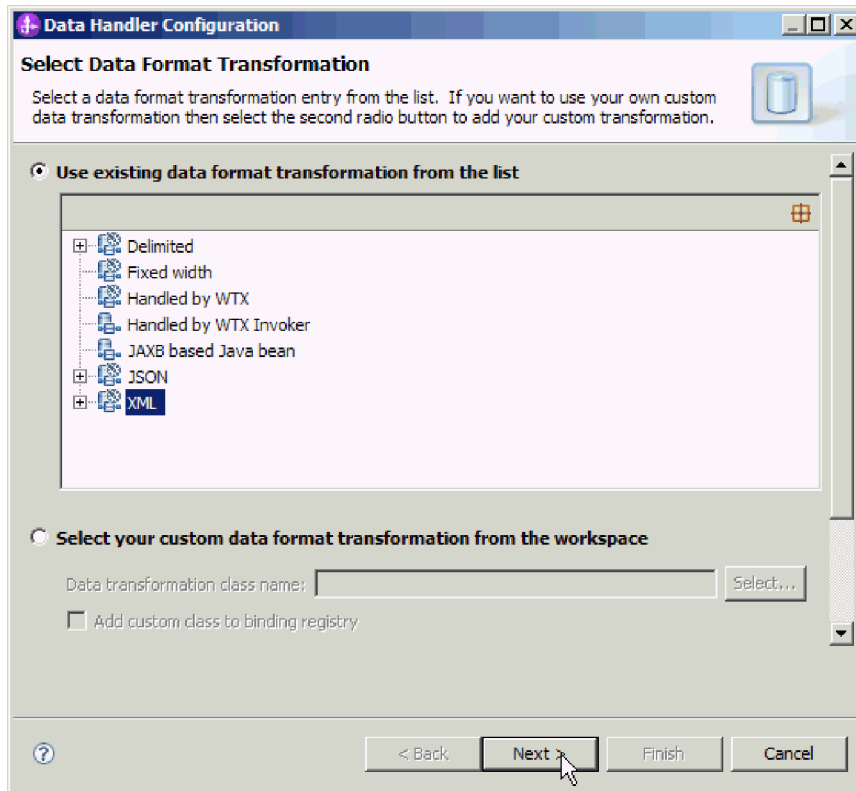


Figure 51. Creating a new data handler configuration

8. In the Data Transformation Properties window, specify the encoding. The default is UTF-8.

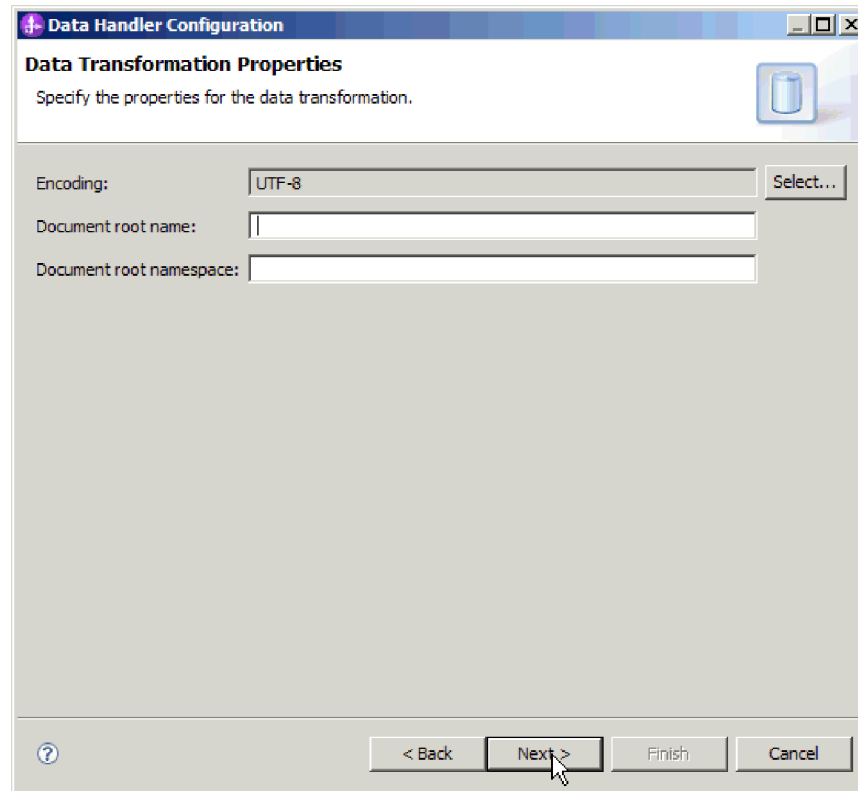


Figure 52. Specifying the encoding for the data handler configuration

9. Click **Next**.
10. Choose the data binding configuration for the operation output. In the Operation window, click **Browse** for the output **Data binding configuration** field. Because the adapter provides only one data binding and this was configured when the operation input DataBinding type was set, you select the same data binding type (DBConfig) for the operation output DataBinding type.
11. Click **Finish**. The next screen shows the Create operation that has been added, with the interaction specification properties.
12. Click **Finish** on the Operations window.

Results

Data handlers are created.

What to do next

In the wizard, continue to specify interaction specification properties and generate artifacts for the module.

Setting interaction specification properties and generating the service

Interaction specification properties are optional. If you choose to set them, the values you specify will appear as defaults in all parent FTP business objects generated by the external service wizard. Interaction specification properties

control the interaction for an operation. While creating artifacts for the module, the adapter generates an import file. The import file contains the operation for the top level business object.

About this task

To set interaction specification properties and generate artifacts, follow this procedure. For more information about interaction specification properties, see the topic about it in this documentation.

Procedure

1. Optional: To set interaction specification properties, populate the fields in the Operations window. You can also click **Advanced** to add additional property details.
 - a. Type values for any fields you want to set as defaults.
 - b. Click **Next**.

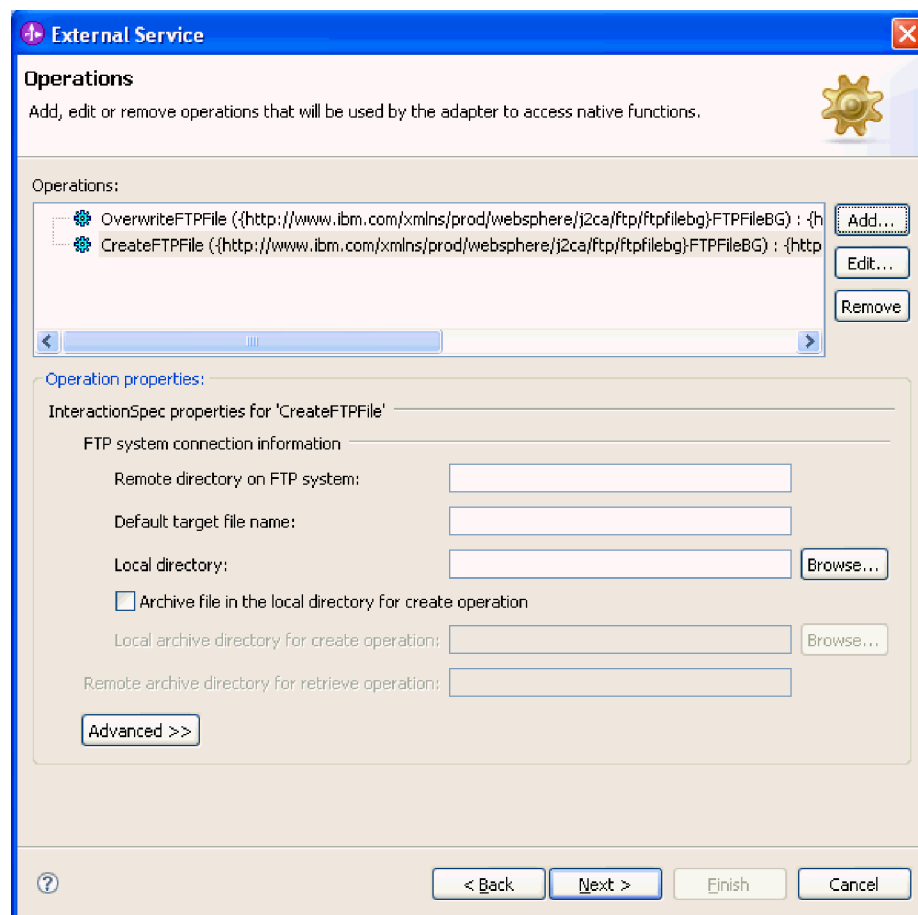


Figure 53. Interaction specification properties

2. On the Generate Service window, supply a name for the interface. This is the name that will display in the WebSphere Integration Developer assembly diagram.

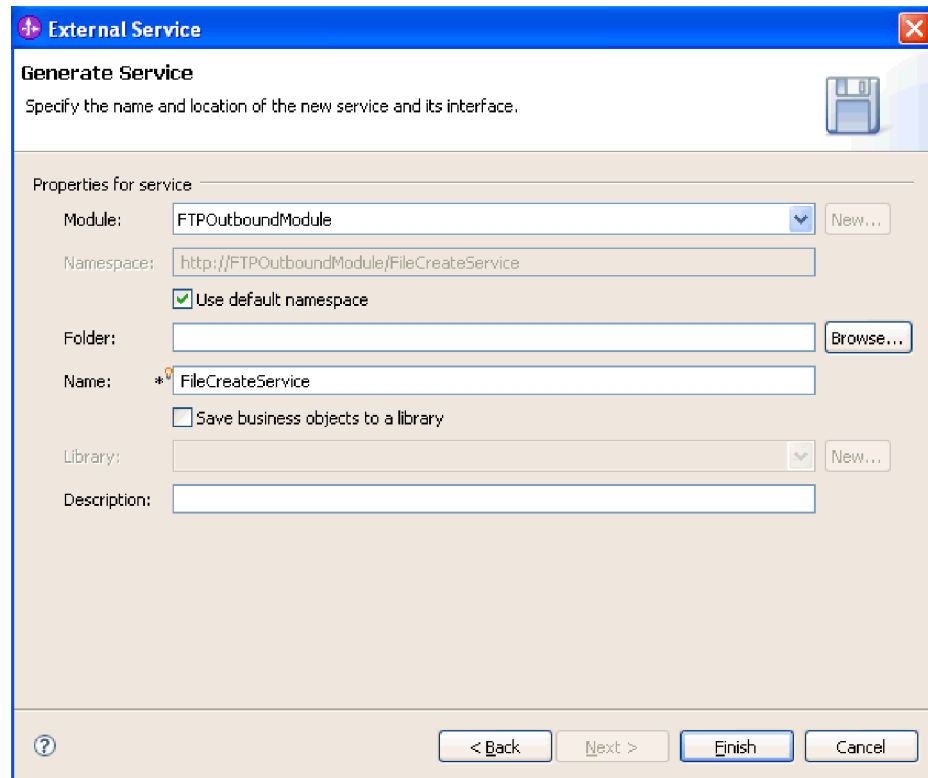


Figure 54. Naming the interface

3. Click **Finish**. The WebSphere Integration Developer assembly diagram opens and the interface you created is displayed.

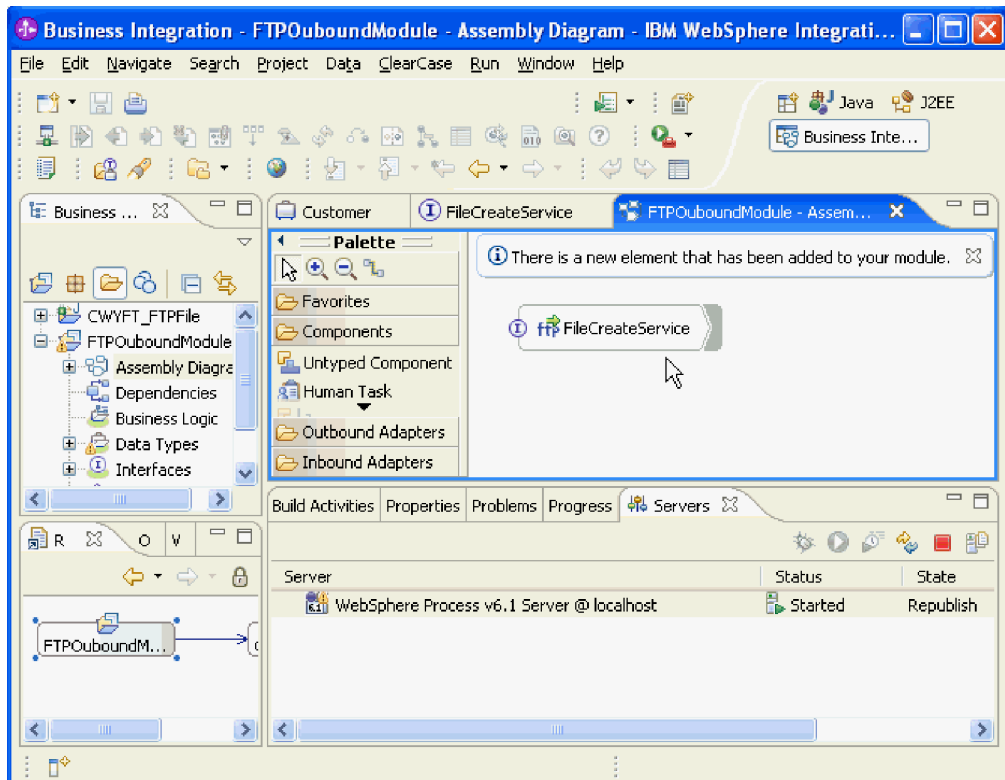


Figure 55. Interface in WebSphere Integration Developer

- Optional: Repeat the previous steps to add all other required operations, including the bindings, data handlers, and interaction specifications.

Results

WebSphere Integration Developer generates the artifacts and the import. The outbound artifacts that are created are visible in the WebSphere Integration Developer Project Explorer under your module.

What to do next

Deploy the module to the server.

Configuring the module for inbound processing

To configure a module to use the adapter for inbound processing, use the external service wizard in WebSphere Integration Developer to build business services, specify data transformation processing, and generate business object definitions and related artifacts.

Setting deployment and runtime properties

Specify deployment and runtime properties that the external service wizard uses to connect to the FTP server.

Before you begin

Before CWYFT can specify the connection properties, you must have started the external service wizard.

About this task

The external service wizard requires this information to connect to the database for discovery and for creating the service description.

Procedure

1. In the Processing Direction window, select **Inbound** and click **Next**.
2. In the **Deploy connector project** field, specify whether to include the adapter files in the module. Choose one of the following options:
 - **With module for use by single application**

With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or when multiple modules need to run different versions of the adapter. By using an embedded adapter, you can upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
 - **On server for use by multiple applications**

If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
3. Define the following FTP system connection information for your module. For more details about the properties in this window, see the topic about managed connection factory properties in this documentation.
 - **Host name** - Specifies the host name of the FTP server.
 - **Remote directory** - Specifies the directory on the FTP server where the adapter polls and picks up files.
 - **Local directory** - Specifies the directory on the adapter workstation where the event files are downloaded from the FTP server.
 - **Port number** - Specifies the port number of the FTP server.
 - **Protocol** - Specifies either normal FTP, secure FTP (FTPS), or FTP over SSH (SFTP). If you select SFTP, the port number will default to 22.
 - **User name** - Specifies the name of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see “User name property (UserName)” on page 211.
 - **Password** - Specifies the password of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see “Password property (Password)” on page 202.

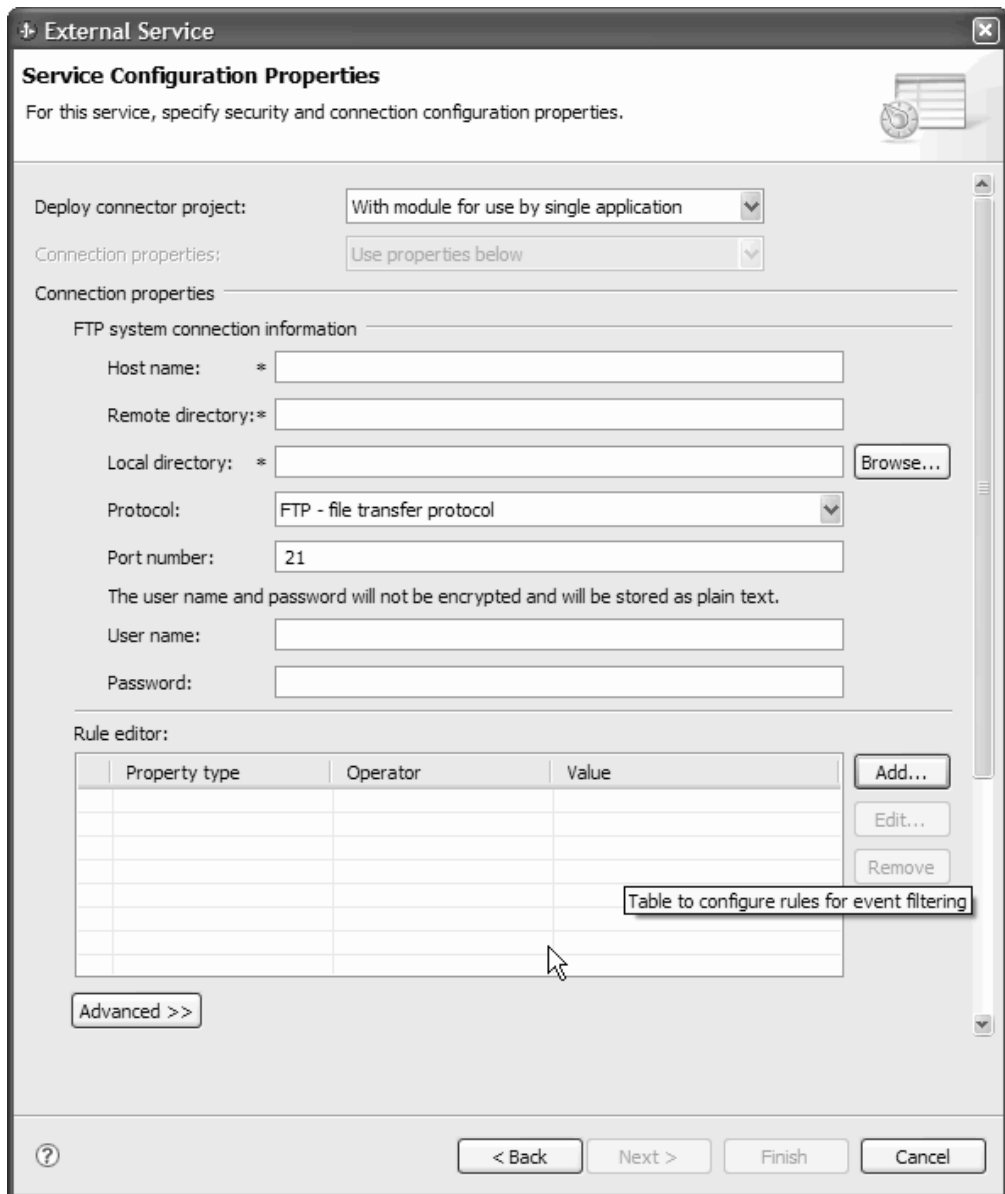


Figure 56. The Service Configuration Properties window

4. To filter the inbound event file by configuring rules, click **Add** or **Edit** in the Rule Editor table. The rule constitutes three parameters: Property Type, Operator and Value.

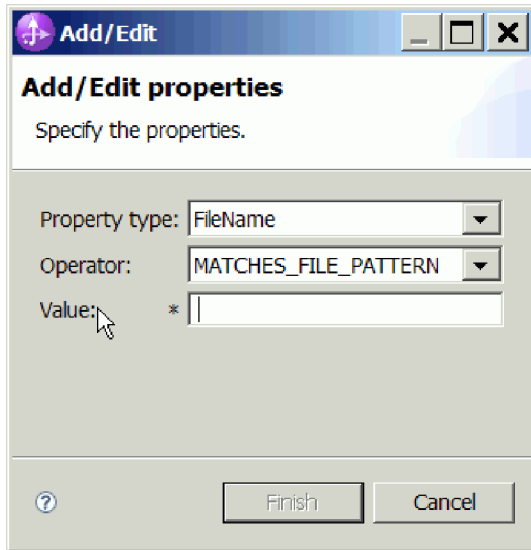


Figure 57. Adding or editing a rule

- a. Select any of the following metadata filtering property types from **Property Type** list.
 - FileName
 - FileSize
 - LastModified
- b. Select the operator for the property type from the **Operator** list. Each of the property type metadata has its own operators. FileName contains MATCHES_FILE_PATTERN (matches pattern) and MATCHES_REGEX (matches regular expression) as its operators. FileSize contains GT (greater than), LT (less than), GE (greater than or equal to), LE (less than or equal to), EQ (equal to), NE (not equal to) in Bytes as its operators. LastModified contains GT (greater than), LT (less than), GE (greater than or equal to), LE (less than or equal to), EQ (equal to), NE (not equal to) (in Day of the week or Time) as its operators. The day-of-the-week value will accept only EQ operator.
- c. Type the value for filtering the event file in the **Value** column. You must enter a valid Java regular expression in value for MATCHES_REGEX operator.

To configure multiple rules, select **END-OF-RULE** option for each rule from the **Property Type** list.

Note: Each expression is grouped by the logical **OR** operator, unless **END-OF-RULE** is selected in the property field. Because you need to select **END-OF-RULE** in a property field to end a rule, all rules will be grouped by the logical **AND** operator, and all expressions within a rule will be grouped by the logical **OR** operator.

Important: Do not select **END-OF-RULE** if you want to configure only one rule, otherwise the rule becomes invalid during run time. If you select **END-OF-RULE** from the **Property Type** list and do not select the second rule, the adapter throws an error.

For more information see, “Rule based event filtering (ruleTable)” on page 212.

5. Optionally, you can click **Advanced** to specify advanced properties. Expand each of the following group sections to review the advanced properties:

- Event polling configuration
- Event delivery configuration
- Event persistence configuration
- Advanced connection configuration
- FTP archiving configuration
- Socks proxy server connection information
- Secure configuration
- Bidi properties
- Logging and tracing properties

The following sections describe the options that are available in these Advanced property groups.

- **Event polling configuration**
 - a. In **Interval between polling periods**, type the number of milliseconds that the adapter waits between polling periods. For more information, see “Interval between polling periods (PollPeriod)” on page 203.
 - b. In **Maximum events in polling period**, type the number of events to deliver in each polling period. For more information, see “Maximum events in polling period (PollQuantity)” on page 203.
 - c. In **Retry interval if connection fails**, type the number of milliseconds to wait before trying to connect after a connection failure during polling. For more information, see “Retry interval if connection fails (RetryInterval)” on page 206.
 - d. In **Number of times to retry the system connection**, type the number of times to retry the connection before reporting a polling error. For more information, see “Number of times to retry the system connection (RetryLimit)” on page 207.
 - e. If you want the adapter to stop if polling errors occur, select **Stop the adapter when an error is encountered while polling**. If you do not select this option, the adapter logs an exception but continues to run. For more information, see “Stop the adapter when an error is encountered while polling (StopPollingOnError)” on page 211.
 - f. Select **Retry EIS connection on startup** if you want the adapter to retry a failed connection when starting. For more information, see “Retry EIS connection on startup (RetryConnectionOnStartup)” on page 206.
- **Event delivery configuration**
 - a. In **Type of delivery**, select the delivery method. The methods are described in “Type of delivery (DeliveryType)” on page 195.
 - b. If you want to ensure that events are delivered only once and to only one export, select **Ensure once-only delivery**. This option might reduce performance but does not result in duplicate or missing event delivery. For more information, see “Ensure assured-only event delivery (AssuredOnceDelivery)” on page 193.
 - c. Select **Do not process events that have a timestamp in the future** if you want the adapter to filter out future events by comparing the timestamp on each event with the system time. For more information, see “Do not process events that have a timestamp in the future (FilterFutureEvents)” on page 198.

- d. In **Event types to process**, type a comma-separated list of the business objects for which you want events delivered. Leave this field blank to receive events for all business object types.
For example, if you want to receive events only when the Customer and Order tables, but not other tables, are changed in the enterprise information system, set this field to `Customer,Order`.
For more information, see “Event types to process (EventTypeFilter)” on page 196.
- e. In **Retry limit for failed events** specify the number of times that the adapter will attempt to redeliver an event before marking it as failed. For more information, see “Retry limit for failed events (FailedEventRetryLimit)” on page 199.
- **Event persistence configuration**
 - a. Select **Auto event create table** if you want the adapter to create the Event Persistence table. For more information, see “Auto create event table property (EP_CreateTable)” on page 193.
 - b. In **Event recovery table name**, specify the name of the table that the adapter uses for event persistence. For more information, see “Event recovery table name property (EP_EventTableName)” on page 196
 - c. In **Event recovery data source (JNDI) name**, specify the JNDI name of the data source that event persistence uses to get the JDBC database connection. For more information, see “Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)” on page 196
 - d. In **User name used to connect to event data source**, specify the user name that event persistence uses to get the database connection from the data source. For more information, see “User name used to connect to event data source property (EP_UserName)” on page 212
 - e. In **Password used to connect to event data source**, specify the password used during event persistence to get the database connection from the data source. For more information, see “Password used to connect to event data source property (EP_Password)” on page 202.
 - f. In **Database schema name**, specify the schema name of the database that event persistence uses. For more information, see “Database schema name property (EP_SchemaName)” on page 194.
- **Advanced connection configuration**
 - a. In **Retrieve files with this pattern**, specify the filter for the event files. For more information, see “Retrieve files with this pattern property (EventFileMask)” on page 206.
 - b. In **Sort event files**, specify the determination for the sorting order of event files being polled. For more information, see “Sort event files property (SortEventFiles)” on page 209.
 - c. In **Encoding used by FTP server**, specify the encoding of the FTP server. For more information, see “Encoding used by FTP server property (EISEncoding)” on page 195.
 - d. In **File content encoding**, specify the encoding used to read the event files. For more information, see “File content encoding property (FileContentEncoding)” on page 197.
 - e. In **FTP server connection mode**, specify the data connection mode used by the FTP server during file transfers. For more information, see “FTP server connection mode property (DataConnectionMode)” on page 195.

- f. In **File transfer type**, specify the file transfer type used during inbound processing. For more information, see “File transfer type property (FileTransferType)” on page 198.
- g. In **Number of files to get at a time**, specify the number of files retrieved from the remote FTP URL. For more information, see “Number of files to get at a time property (ftpGetQuantity)” on page 198.
- h. In **Number of poll periods between downloads**, specify how frequently the adapter polls the FTP server. For more information, see “Number of poll periods between downloads property (ftpPollFrequency)” on page 199.
- i. In **Custom parser class name**, specify the fully qualified class name of the custom parser that is used to parse the ls output. For more information, see “Custom parser class name property (CustomParserClassName)” on page 194.
- j. Select **Pass only file name and directory, not the content** to specify that the file content of the event file is not sent to the export. For more information, see “Pass only file name and directory, not the content property (FilePassByReference)” on page 197.
- k. Select **Include business object delimiter in the file content** to specify that the delimiter will be sent with the business object content for further processing. For more information, see “Include business object delimiter in the file content property (IncludeEndBODelimiter)” on page 200.
- l. Select **Split file content based on the size (bytes) or delimiter** to use the size in bytes or the delimiter to split the file content. For more information, see “Splitting function class name property” on page 210.
- m. In **Specify criteria to split file content**, specify that different values will be taken, based on the value of the SplittingFunctionClassName property. For more information, see “Specify criteria to split file content property (SplitCriteria)” on page 209.
- n. In **Split function class name**, specify the fully qualified class name of the class file to be used to enable file splitting. For more information, see “Splitting function class name property” on page 210.
- o. In **Run FTP script file before downloading files**, specify the path of the script file that is run before every inbound poll cycle. For more information, see “Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)” on page 200.
- p. In **Run FTP script file after downloading files**, specify the path of the script file that is run after every inbound poll cycle. For more information, see “Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)” on page 199.
- **FTP archiving configuration**
 - a. In **Local archive directory**, specify the absolute path of the local Archive directory. For more information, see “Local archive directory property (LocalArchiveDirectory)” on page 200.
 - b. In **File extension for local archive**, specify the file extension used to archive the original event file. For more information, see “File extension for local archive property (originalArchiveExt)” on page 202.
 - c. In **Success file extension for local archive**, specify the file extension used to archive all of the successfully processed business objects. For more information, see “Success file extension for local archive property (SuccessArchiveExt)” on page 211.

- d. In **Failure file extension for local archive**, specify the file extension used to archive business objects in the event file that are not successfully processed. For more information, see “Failure file extension for local archive property (FailedArchiveExt)” on page 197.
 - e. In **Remote archive directory**. For more information, see “Remote archive directory property (ftpArchiveDirectory)” on page 204.
 - f. In **File extension for remote archive**, specify the file extension or suffix that the adapter uses to rename the remote FTP file. For more information, see “File extension for remote archive property (ftpRenameExt)” on page 197.
- **Socks proxy server connection information**
 - a. In **Host name**, specify the host name of the machine used as a proxy server through which the adapter requests are routed to the FTP server. For more information, see “Host name property (SocksProxyHost)” on page 208.
 - b. In **Port number**, specify the port number of the proxy server through which the adapter requests are routed to the FTP server. For more information, see “Port number property (SocksProxyPort)” on page 208.
 - c. In **User name**, specify the user name for authenticating the proxy server. For more information, see “User name property (SocksProxyUserName)” on page 208.
 - d. In **Password**, specify the password used to authenticate the proxy server. For more information, see “Password property (SocksProxyPassword)” on page 208.
 - **Secure configuration**

If you want to compare the host key of the SFTP server with the host keys known to the adapter:

 - a. Select the **Enable remote server authentication for SFTP protocol** check box. The host key file has to be available with the host keys of the trusted server before the first attempt to connect to a SFTP server is made. For more information, see Enable server verification property (EnableServerVerification).
 - b. In the **Host key file** field, specify the absolute file path to the host key file. The host key file is created by the administrator and contains the host keys of all the trusted servers. The Host key file property points to the file on the adapter workstation. For more information, see Host key file property (HostKeyFile).

If you want to enable public key authentication, specify these properties:

 - 1) In the **Private key file** field, specify the private key used to authenticate to the secure shell server. For more information, see “Private key file property (PrivateKeyFilePath)” on page 204.
 - 2) In the **Passphrase** field, specify the phrase used for enhanced security by encrypting the private key. For more information, see Passphrase property (Passphrase).
 - **Bidi properties**
 - **Logging and tracing**
 - a. If you have multiple instances of the adapter, expand and set Adapter ID to a value that is unique for this instance. For more information about this property, see “Resource adapter properties” on page 158.
 - b. Select **Disguise user data as 'XXX' in log and trace files** if you want to prevent sensitive user data from being written to log and trace files. For

more information, see “Disguise user data as “XXX” in log and trace files (HideConfidentialTrace) property” on page 160.

6. In the Service properties section of the window, select one of the following options from the **Function selector** field. A function selector assigns incoming messages or requests to the correct operation on the service.
 - **Use default function selector ‘FilenameFunctionSelector’**
If choosing to use this option, click **Next**.
 - **Use a function selector configuration**
If choosing this option, complete the following steps:
 - a. Click **Select** next to the **Function Selector** field.

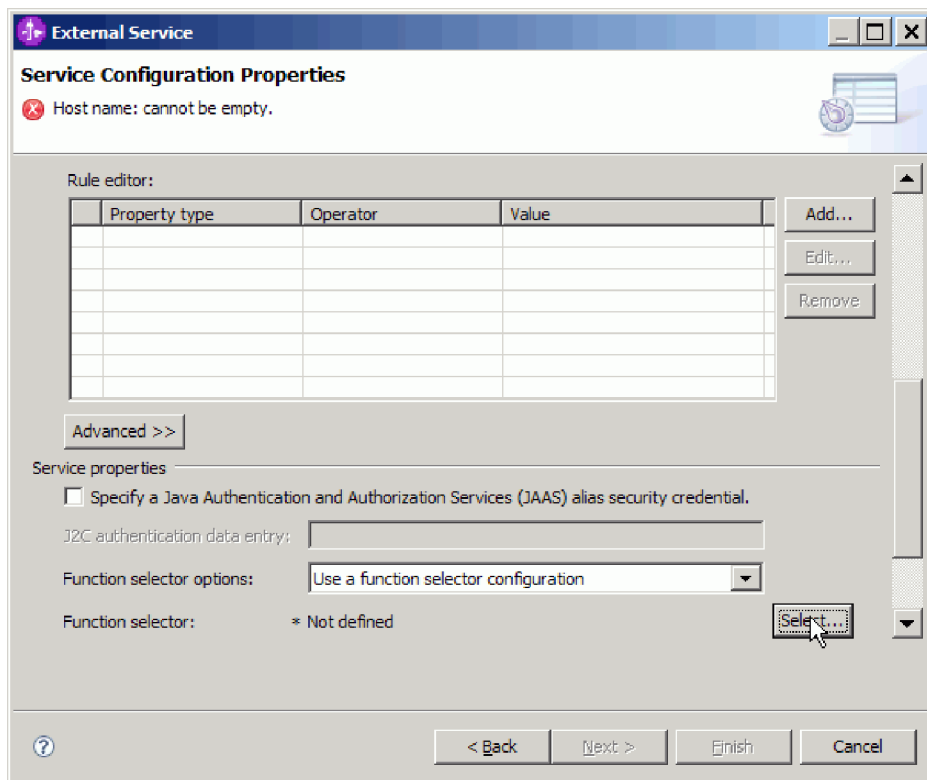


Figure 58. The Service Configuration Properties window

- b. In the Select Function Selector window, select the **FilenameFunctionSelector** properties. Click **Next**

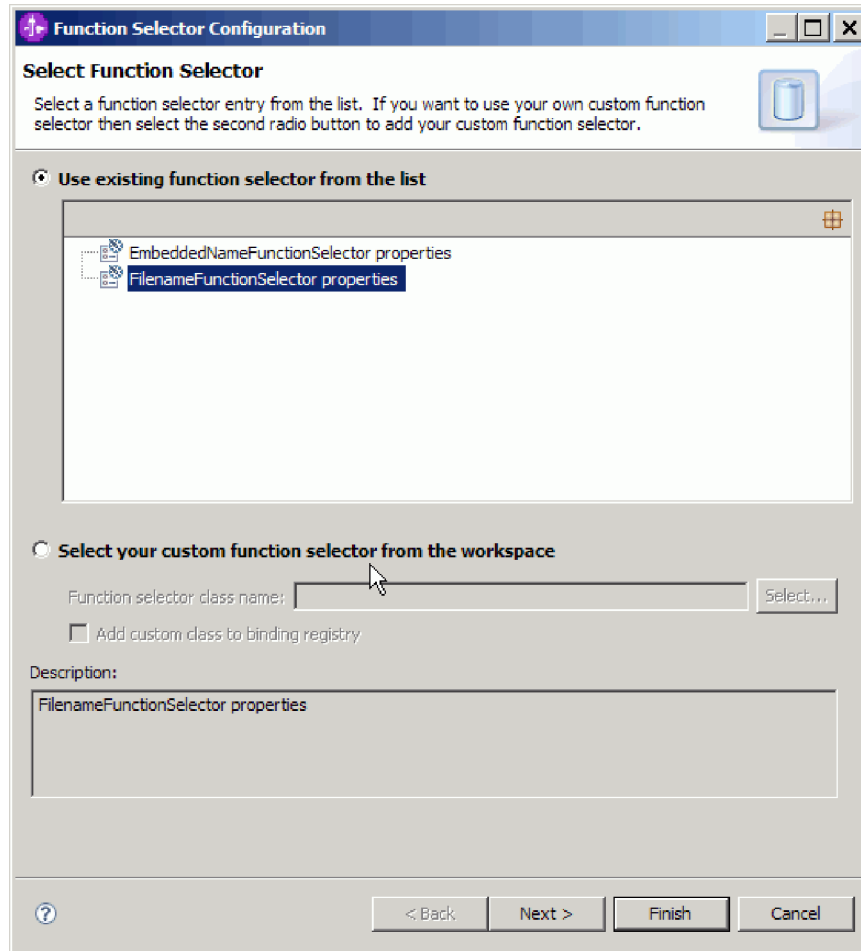


Figure 59. The Select Function Selector window

Note: The EIS function name is not available in the external service wizard. If you want to specify a value other than the default that is generated by the adapter, you can edit it using the assembly editor.

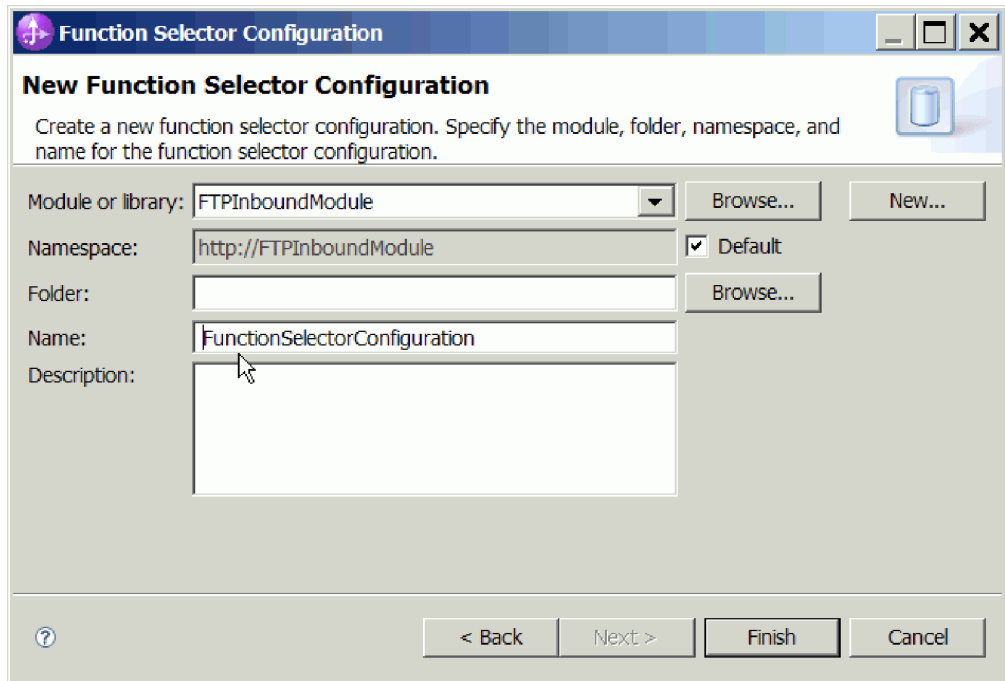


Figure 60. The New Function Selector Configuration window

7. Click **Finish** in the Function Selector Configuration window.

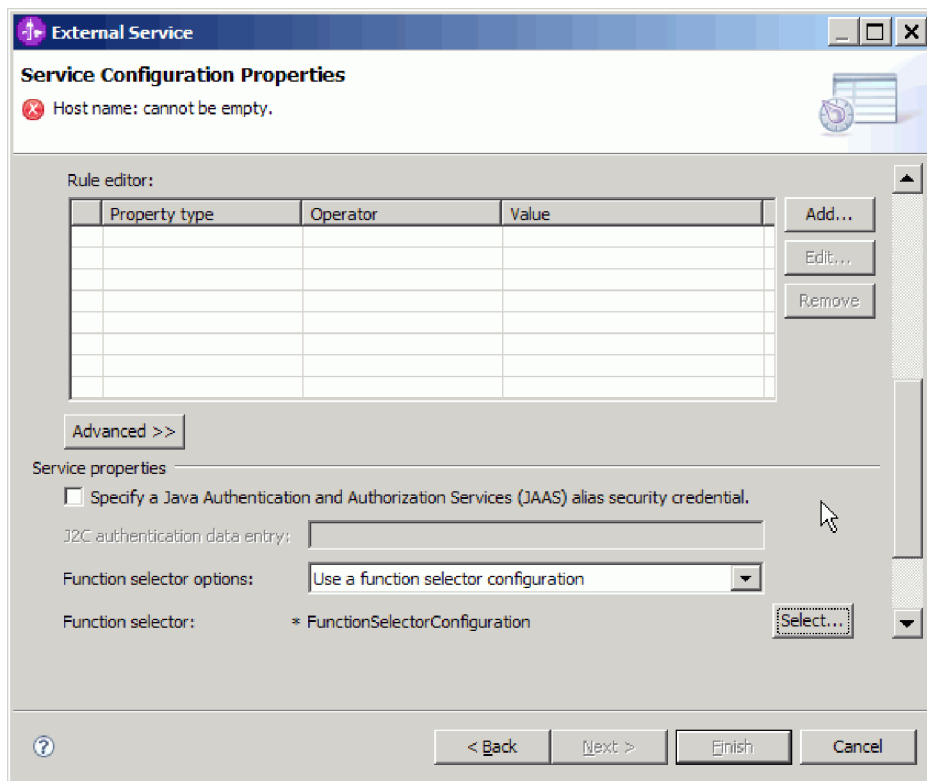


Figure 61. The Function Selector Selection window

8. Click **Next** in the Service Configuration Properties window.

Results

The external service wizard now has the information it needs to connect to the FTP server.

If you selected either the **Use default data binding 'FTPFileBaseDataBinding' for all operations** or **Specify a data binding for each operation** Data binding configuration options, click **Next** to continue to work in the wizard to select a data type and to name the operation associated with the data type.

If you selected the **Use a data binding configuration for all operations** Data binding configuration option, proceed to “Configuring the data binding” on page 83.

Selecting a data type and operation name

Use the external service wizard to select a data type and to name the operation associated with the data type. For inbound communications, the external service wizard gives you the choice of three different data types: user defined type, generic FTP business object, and generic FTP business object with business graph. Each data type corresponds to a business object structure.

Before you begin

You must have specified the connection properties for the adapter to connect to the FTP server before you can complete the following steps.

About this task

To select a data type and name the operation associated with it, follow this procedure.

Procedure

1. In the Operations window, click **Add**.



Figure 62. Adding an operation

2. In the Add Operations window, select **The data type for the operation input**, and click **Next**. If you select **User defined type**, you must provide a user-defined data binding to support it. The **Generic FTP business object** provided data binding only supports generic input types for the supported operations.
3. In the Operation window, type a name in the **Operation name** field or keep the default emitFTPFile name.

Note: Names cannot contain spaces.

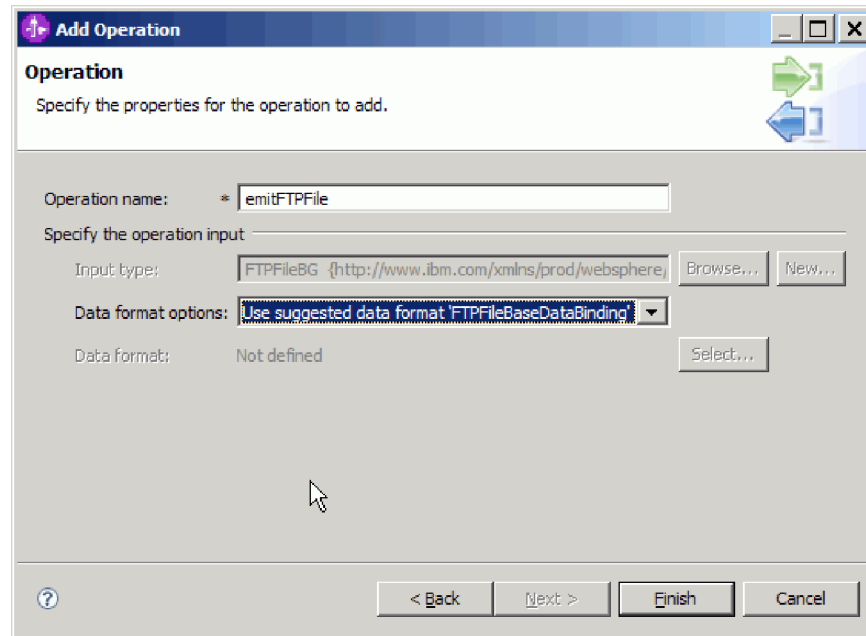


Figure 63. Naming the operation

Results

A data type is defined for the module and the operation associated with the data type is named.

If you choose to use a configured data binding, continue to work from the current external service wizard window, to add and configure a data binding to be used with the module.

If you choose to use the default data binding, proceed to “Generating the service” on page 111.

Configuring the data binding

Each data type has an equivalent data binding that is used to read the fields in a business object and fill in the corresponding fields in a file. In the external service wizard, you add a data binding to your module and configure it to correspond with your data type. This way, the adapter knows how to populate the fields in a file with information it receives in the business object.

Before you begin

You must have selected a data type and chosen a configuration name to be associated with the data type.

About this task

To add and configure a data binding for the module, follow this procedure.

Note: Data bindings can be configured prior to running the external service wizard using WebSphere Integration Developer. To do this, select **New** → **Resource configuration** in WebSphere Integration Developer and complete the data binding windows described in this documentation.

Procedure

1. On the Operation window, select **New** to enter the **Input type** in the **Specify the operation input** area. You do this the first time you set the data binding. To use the same data binding configuration later, click **Browse** and select it.

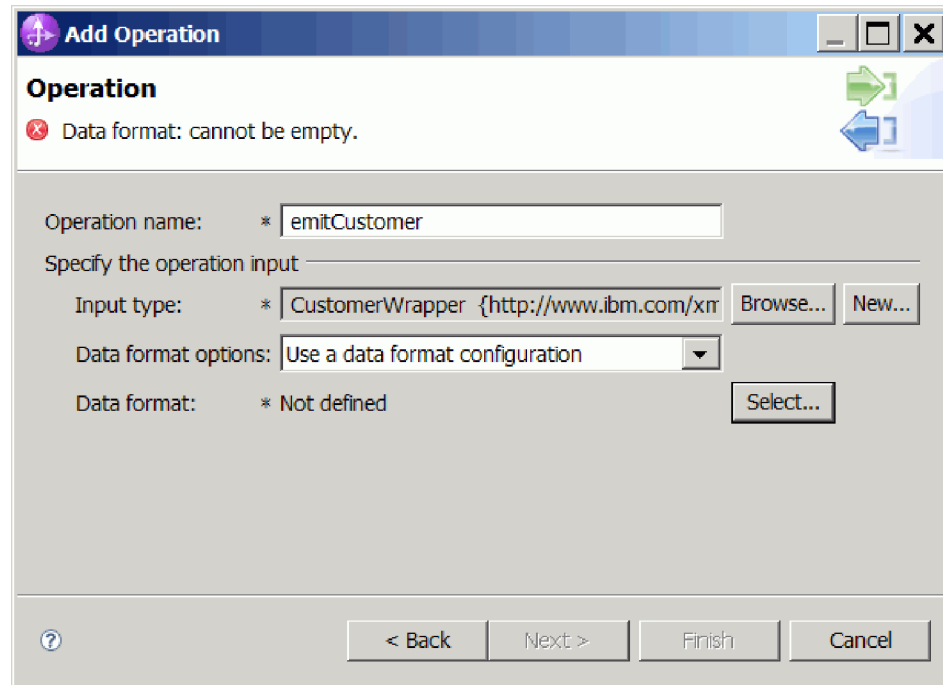


Figure 64. Operation window

2. Optional: On the **New Data Binding Configuration** window, the Module defaults to the module name you typed earlier in the wizard. If this is not the module that you want to create a data binding for, choose **New** to create a new module.
3. Optional: If you want to choose a new folder for the artifact, click **Browse** and select a new folder location. If you do not browse for a new folder location, the artifacts will be created in the root directory for the module.
4. Select Use a data format configuration from the **Data format options** drop down list.
5. Type the Configuration name for the data binding configuration (this example uses DataBindingConfiguration).
6. Click **Next**.

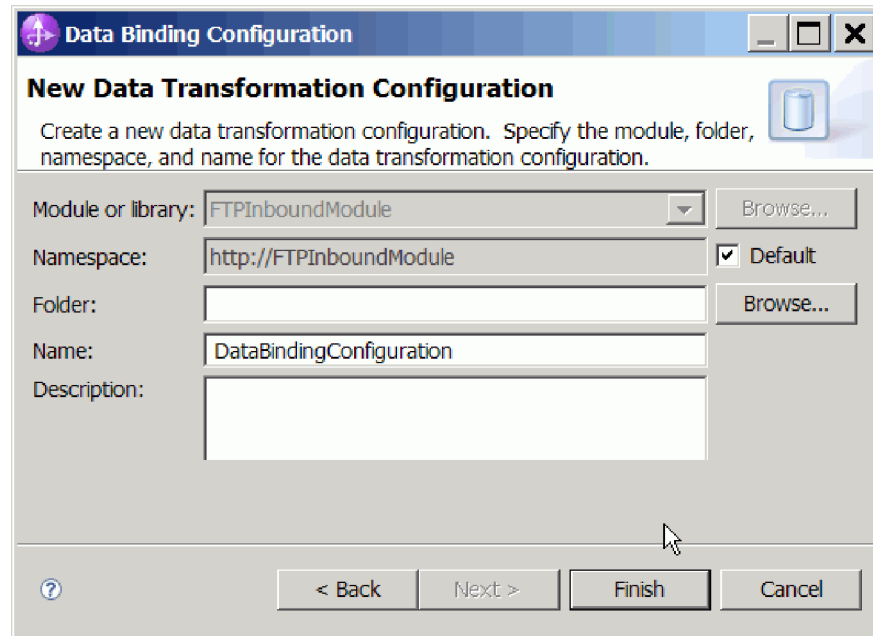


Figure 65. Naming the data binding configuration

7. Click Next.

Results

A data binding is configured for use with the module.

What to do next

From the current external service wizard window, you will proceed to select a data handler for the module.

Configuring data handlers

When you select a data type that contains business objects, you need to specify the data handler that will perform the conversions between a business object and a native format.

Before you begin

You must have created a data binding before specifying the data handlers for the module.

About this task

To specify data handlers, follow this procedure.

Note: Data handlers can be configured prior to running the external service wizard using WebSphere Integration Developer. To do this, select **New** → **Binding Resource Configuration** in WebSphere Integration Developer and complete the data handler windows described in this documentation.

Procedure

1. In the New Data Transformation configuration window, click **New**.

2. Enter the name for the data handler configuration in the **Name** field (this example uses DataBindingConfiguration). You do this the first time when you set the data handler. To use the same data handler later, click **Browse** and select it.
3. Click **Next**.
4. In the Data Transformation Properties window, click the drop down list next to the **Binding Type** property. Two choices are provided: DataBinding and DataHandler. To use a data binding developed for an earlier version of the adapter, select DataBinding. To configure a new data handler, select DataHandler. Click **Select** to create a new data handler configuration.

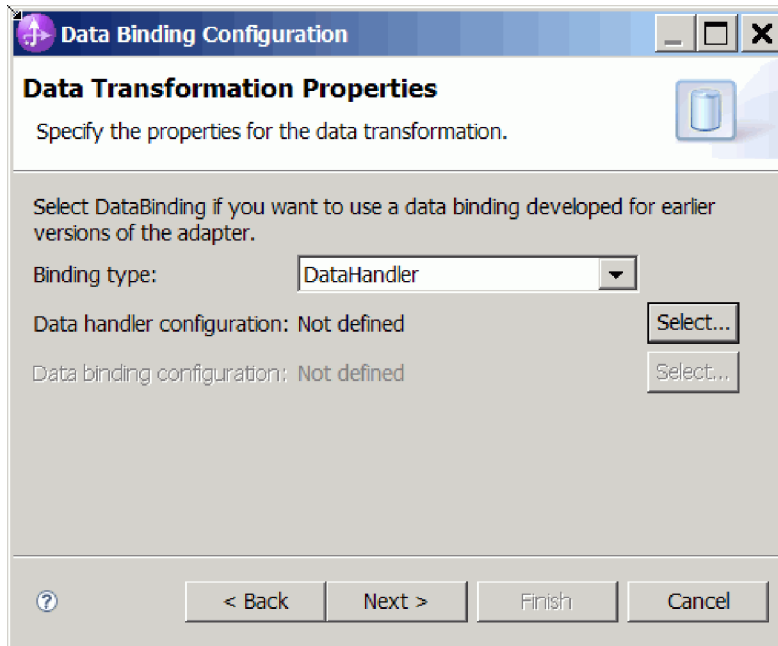


Figure 66. Specifying the Binding Type on the Data Transformation window

5. Click **Next**.
6. Choose the class name for the data handler. In the Select Data Format Transformation window, click **Use existing data format transformation** from the list option. A list of available data handler classes is displayed. Select the data handler class (this example uses XML data handler). Click **Next**.

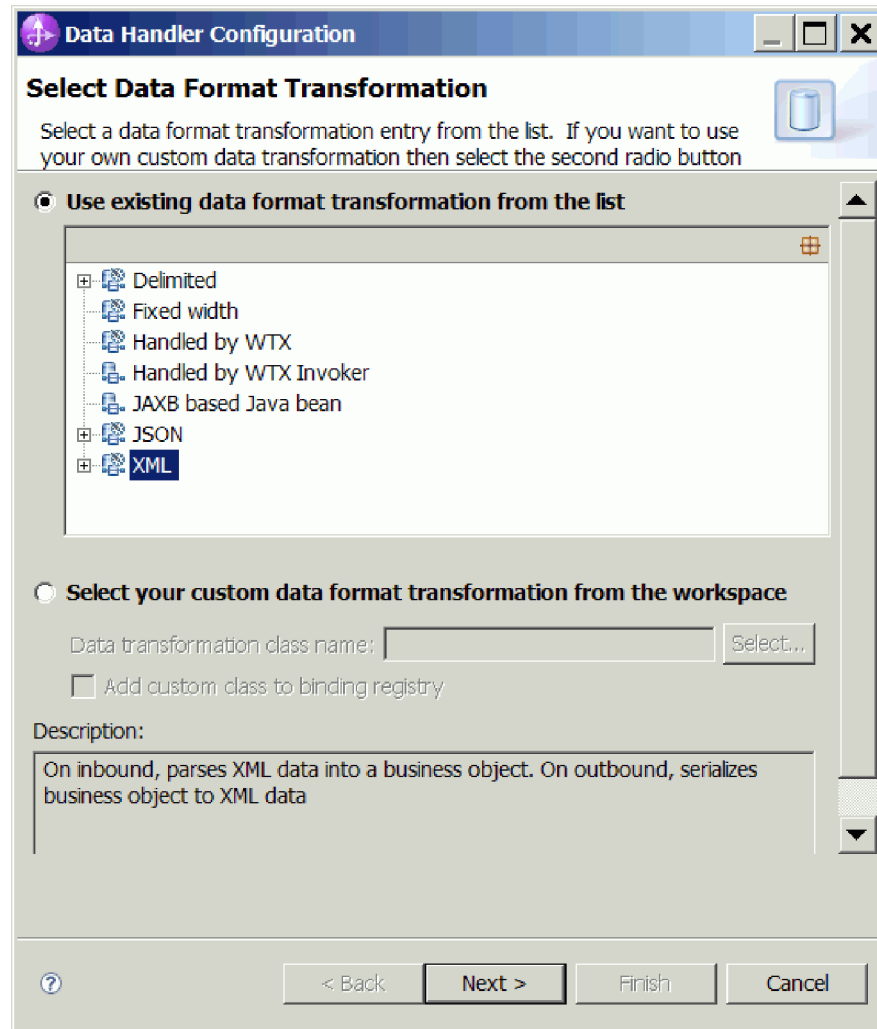


Figure 67. Creating a new data handler configuration

7. In the Data Transformation Properties window, specify the encoding. The default is UTF-8.

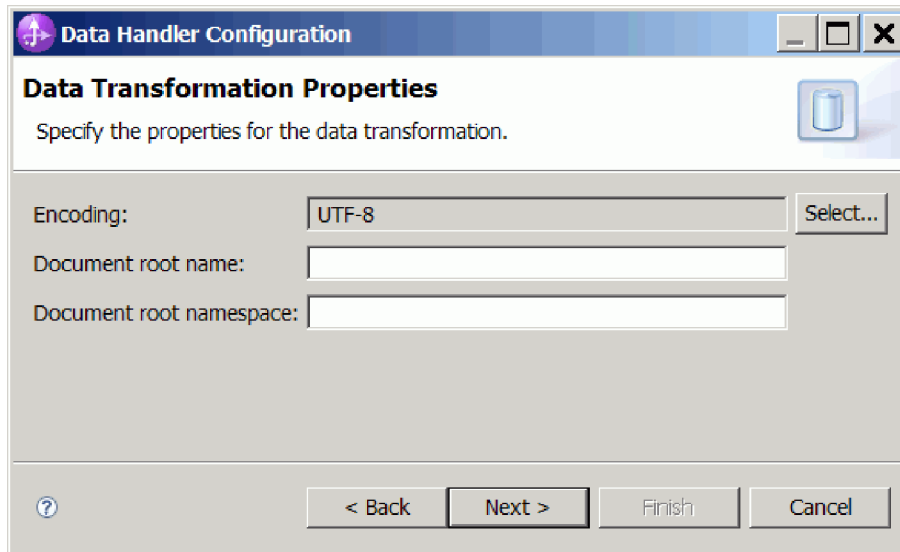


Figure 68. Specifying the encoding for the data handler configuration

8. Click **Next**.
9. In the **New Data Transformation Configuration** window, click **New** and provide a name for the data handler configuration (example uses `DataHandlerConfiguration`). You click **New** the first time you set the data handler. To use this data handler later, click **Browse**.

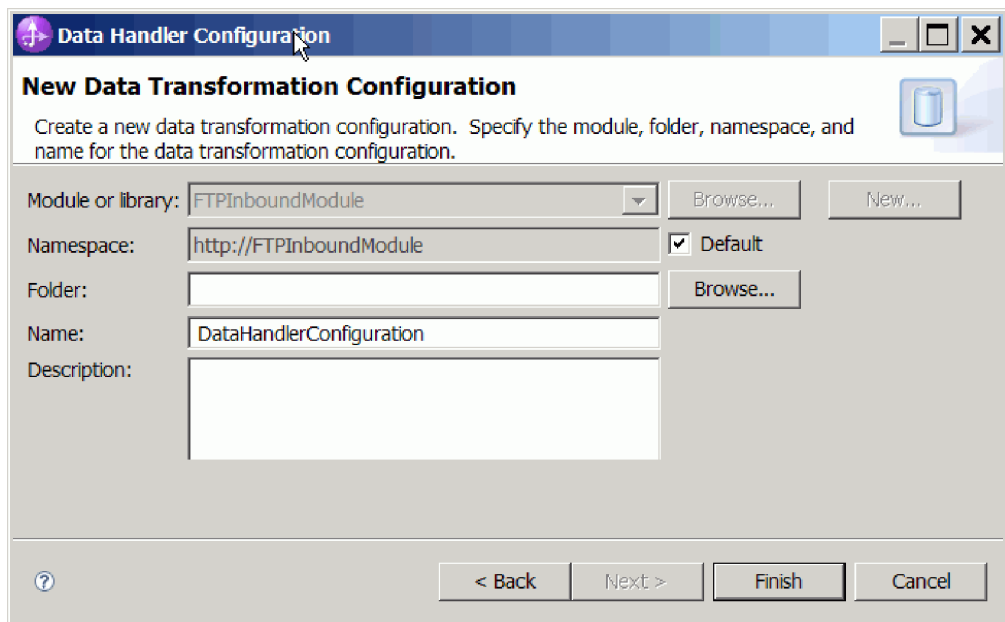


Figure 69. Specifying a name for the data handler configuration

10. Click **Finish**.
11. The **Data Transformation Properties** window is displayed.

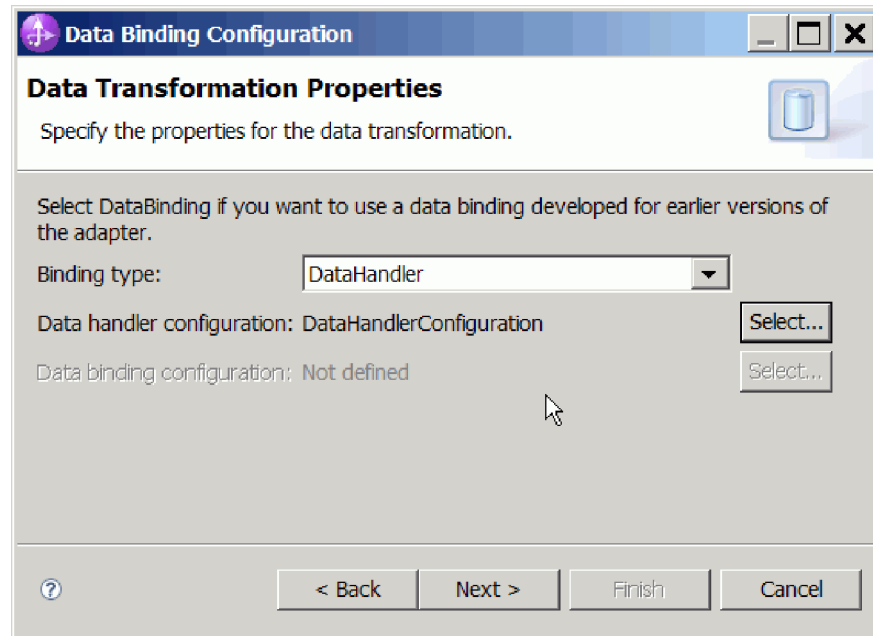


Figure 70. Data Transformation Properties window

Results

Data handlers are created.

What to do next

Continue in the wizard to specify interaction specification properties and generate artifacts for the module.

Generating the service

While creating artifacts for the module, the adapter generates an export file. The export file contains the operation for the top level business object.

About this task

To generate artifacts, follow this procedure.

Procedure

1. Click **Next** in the Operations window.
2. In the Generate Service window, supply a name for the interface. This is the name that will display in the WebSphere Integration Developer assembly diagram.
3. Click **Finish**. The WebSphere Integration Developer assembly diagram opens and the interface you created is displayed.

Results

The WebSphere Integration Developer generates the artifacts and an export. The inbound artifacts that are created are visible in the WebSphere Integration Developer Project Explorer under your module.

What to do next

Deploy the module to the server.

Chapter 5. Changing interaction specification properties using the assembly editor

To change interaction specification properties for your adapter module after generating the service, use the assembly editor in WebSphere Integration Developer.

Before you begin

You must have used the external service wizard to generate a service for the adapter.

About this task

You might want to change interaction specification properties after you have generated a service for the adapter. Interaction specification properties, which are optional, are set at the method level, for a specific operation on a specific business object. The values you specify will appear as defaults in all parent business objects generated by the external service wizard. You can change these properties before you export the EAR file. You cannot change these properties after you deploy the application.

To change the interaction specification properties, use the following procedure.

Procedure

1. From the Business Integration perspective of WebSphere Integration Developer, expand the module name.
2. Expand **Assembly Diagram** and double-click the interface.
3. Click the interface in the assembly editor. (It shows the module properties if you don't do the extra click.)
4. Click the **Properties** tab. (You can also right-click the interface in the diagram and click **Show in Properties**.)
5. Under **Binding**, click **Method bindings**. The methods for the interface are displayed, one for each combination of business object and operation.
6. Select the method whose interaction specification property you want to change.
7. Click **Advanced** and change the property in the **Generic** tab. Repeat this step for each method whose interaction specification property you want to change.

Results

The interaction specification properties associated with your adapter module are changed.

What to do next

Deploy the module.

Chapter 6. Deploying the module

Deploy a module to place the files that make up your module and adapter into an operational environment for production or testing. In WebSphere Integration Developer, the integrated test environment features runtime support for WebSphere Process Server or WebSphere Enterprise Service Bus, or both, depending on the test environment profiles that you selected during installation.

Deployment environments

There are test and production environments into which you can deploy modules and adapters.

In WebSphere Integration Developer, you can deploy your modules to one or more servers in the test environment. This is typically the most common practice for running and testing business integration modules. However, you can also export modules for server deployment on WebSphere Process Server or WebSphere Enterprise Service Bus as EAR files using the administrative console or command-line tools.

Deploying the module for testing

In WebSphere Integration Developer, you can deploy a module that includes an embedded adapter to the test environment and work with server tools that enable you to perform such tasks as editing server configurations, starting and stopping servers and testing the module code for errors. The testing is generally performed on the interface operations of your components, which enables you to determine whether the components are correctly implemented and the references are correctly wired.

About this task

Generating and wiring a target component for testing inbound processing

Before deploying to the test environment a module that includes an adapter for inbound processing, you must first generate and wire a target component. This target component serves as the *destination* to which the adapter sends events.

Before you begin

You must have generated an export module, using the external service wizard.

About this task

Generating and wiring a target component for inbound processing is required in a testing environment only. It is not necessary when deploying the adapter in a production environment.

The target component receives events. You *wire* the export to the target component (connecting the two components) using the assembly editor in WebSphere Integration Developer. The adapter uses the wire to pass event data (from the export to the target component).

Procedure

1. Create the target component
 - a. From the Business Integration perspective of WebSphere Integration Developer, expand **Assembly Diagram** and double-click the export component. If you did not change the default value, the name of the export component is the name of your adapter + **InboundInterface**.

An interface specifies the operations that can be called and the data that is passed, such as input arguments, returned values, and exceptions. The **InboundInterface** contains the operations required by the adapter to support inbound processing and is created when you run the external service wizard.
 - b. Create a new component by expanding **Components**, selecting **Untyped Component**, and dragging the component to the Assembly Diagram. The cursor changes to the placement icon.
 - c. Click the component to have it displayed in the Assembly Diagram.
2. Wire the components.
 - a. Click and drag the export component to the new component. This draws a wire from the export component to the new component, as shown in the following figure:
 - b. Save the assembly diagram. Click **File** → **Save**
3. Generate an implementation for the new component.
 - a. Right-click on the new component and select **Generate implementation**.

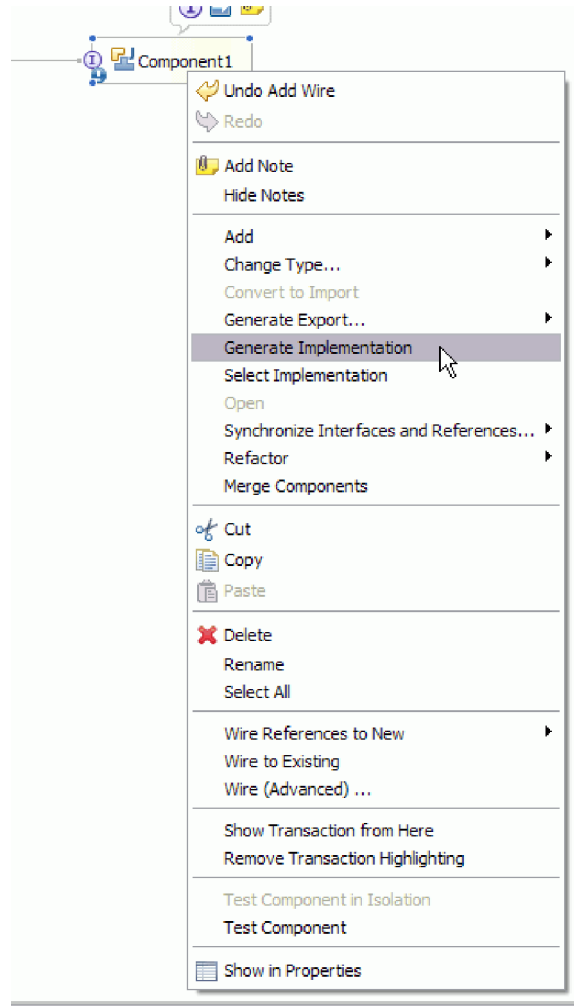


Figure 71. Generating a Java implementation

- b. Select **(default package)** and click **OK**. This creates an endpoint for the inbound module.
The Java implementation is displayed in a separate tab.
- c. **Optional:** Add print statements to print the data object received at the endpoint for each of the endpoint methods.
- d. Click **File** → **Save** to save the changes.

What to do next

Continue deploying the module for testing.

Adding the module to the server

In WebSphere Integration Developer, you can add modules to one or more servers in the test environment.

Before you begin

If the module you are testing uses an adapter to perform inbound processing, you need to generate and wire a *target component* to which the adapter will send events.

About this task

In order to test your module and its use of the adapter, you need to add the module to the server.

Procedure

1. *Conditional:* If there are no servers in the **Servers view**, add and define a new server by performing the following steps:
 - a. Place your cursor in the **Servers view**, right click and select **New** → **server**
 - b. From the Define a New Server window, select the server type.
 - c. Configure server's settings.
 - d. Click **Finish** to publish the server.
2. Add the module to the server
 - a. Switch to the servers view. In WebSphere Integration Developer, select **Windows** → **Show View** → **Servers**
 - a. Start the server. In the Servers tab in the lower-right pane of the WebSphere Integration Developer screen, right-click on the server, and then select **Start**.
3. When the server status is *Started*, right-click on the server, and select **Add and remove projects**.
4. In the Add and Remove Projects screen, select your project and click **Add**. The project moves from the **Available projects** list to the **Configured projects** list.
5. Click **Finish**. This deploys the module on the server.

The Console tab in the lower-right pane displays a log while the module is being added to the server.

What to do next

Test the functionality of your module and the adapter.

Testing the module for outbound processing using the test client

Test the assembled module and adapter for outbound processing using the WebSphere Integration Developer integration test client.

Before you begin

You need to add the module to the server first.

About this task

Testing a module is generally performed on the interface operations of your components, which enables you to determine whether the components are correctly implemented and the references are correctly wired.

Procedure

1. Select the module you want to test, right-click on it, and select **Test** → **Test Module**.
2. For information on testing a module using the test client, see the *Testing modules and components* topic in the WebSphere Integration Developer information center.

What to do next

If you are satisfied with the results of testing your module and adapter, you can deploy the module and adapter to the production environment.

Deploying the module for production

Deploying a module created with the external service wizard to WebSphere Process Server or WebSphere Enterprise Service Bus in a production environment is a two-step process. First, you export the module in WebSphere Integration Developer as an enterprise archive (EAR) file. Second, you deploy the EAR file using the WebSphere Process Server or WebSphere Enterprise Service Bus administrative console.

About this task

Installing the RAR file (for modules using stand-alone adapters only)

If you chose not to embed the adapter with your module, but instead choose to make the adapter available to all deployed applications in the server instance, you will need to install the adapter in the form of a RAR file to the application server. A RAR file is a Java archive (JAR) file that is used to package a resource adapter for the Java 2 Connector (J2C) architecture.

Before you begin

You must have set **Deploy connector project** to **On server for use by multiple adapters** in the Service Generation and Deployment Configuration window of the external service wizard.

About this task

Installing the adapter in the form of a RAR file results in the adapter being available to all J2EE application components running in the server runtime.

Procedure

1. Start the administrative console.
2. Click **Resources** → **Resource Adapters** → **Resource adapters**.
3. In the Resource adapters page, click **Install RAR**.

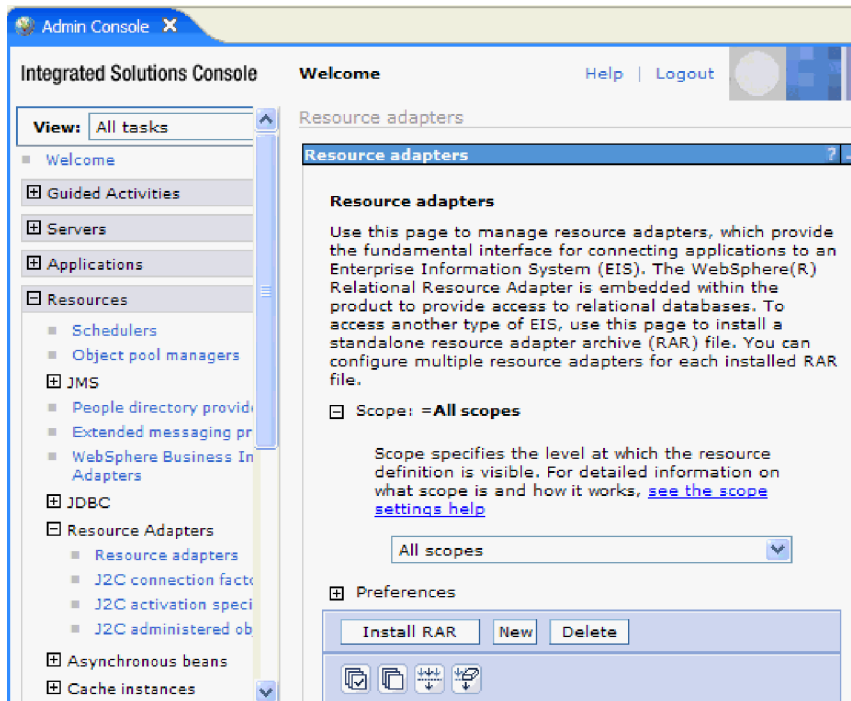


Figure 72. The Install RAR button on the Resource adapters page

4. In the Install RAR file page, click **Browse** and navigate to the RAR file for your adapter.

The RAR files are typically installed in the following path:
`WID_installation_directory/ResourceAdapters/adapter_name/deploy/adapter.rar`

5. Click **Next**.
6. In the Resource adapters page, optionally change the name of the adapter and add a description.
7. Click **OK**.
8. Click **Save** in the **Messages** box at the top of the page.

What to do next

The next step is to export the module as an EAR file that you can deploy on the server.

Exporting the module as an EAR file

Using WebSphere Integration Developer, export your module as an EAR file. By creating an EAR file, you capture all of the contents of your module in a format that can be easily deployed to WebSphere Process Server or WebSphere Enterprise Service Bus.

Before you begin

Before you can export a module as an EAR file, you must have created a module to communicate with your service. The module should be displayed in the WebSphere Integration Developer Business Integration perspective.

About this task

To export the module as an EAR file, perform the following procedure.

Procedure

1. Right-click the module and select **Export**.
2. In the Select window, expand **Java EE**.
3. Select **EAR file** and click **Next**.
4. Optional: Select the correct EAR application. The EAR application is named after your module, but with “App” added to the end of the name.
5. **Browse** for the folder on the local file system where the EAR file will be placed.
6. Optionally, if you want to export the source files, select **Export source files**. This option is provided in case you want to export the source files in addition to the EAR file. Source files include files associated with Java components, data maps, and so on.
7. To overwrite an existing file, click **Overwrite an existing file**.
8. Click **Finish**.

Results

The contents of the module are exported as an EAR file.

Install the module in the administrative console. This deploys the module to WebSphere Process Server or WebSphere Enterprise Service Bus.

Installing the EAR file

Installing the EAR file is the last step of the deployment process. When you install the EAR file on the server and run it, the adapter, which is embedded as part of the EAR file, runs as part of the installed application.

Before you begin

You must have exported your module as an EAR file before you can install it on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

To install the EAR file, perform the following procedure. For more information on clustering adapter module applications, see the <http://www.ibm.com/software/webservers/appserv/was/library/>.

Procedure

1. Open the WebSphere Process Server or WebSphere Enterprise Service Bus administrative console by right-clicking your server instance and selecting **Run administrative console**.
2. In the administrative console window, click **Applications** → **Install New Applications**.

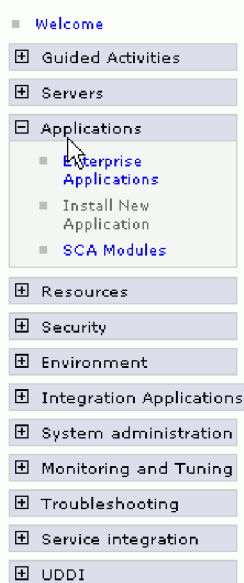


Figure 73. Preparing for the application installation window

3. Click **Browse** to locate your EAR file and click **Next**. The EAR file name is the name of the module followed by "App."
4. Optional: If you are deploying to a clustered environment, complete the following steps.
 - a. On the **Step 2: Mapping modules to servers** window, select the module.
 - b. Select the name of the server cluster.
 - c. Click **Apply**.
5. Click **Next** to open the Summary. Verify that all settings are correct and click **Finish**.
6. Optional: If you are using an authentication alias, complete the following steps:
 - a. Expand **Security** and select **Business Integration Authentication Aliases**.
 - b. Select the authentication alias that you want to configure. You must have administrator or operator authority to make changes to authentication alias configurations.
 - c. Optional: If it is not already filled in, type the **User name**.
 - d. If it is not already filled in, type the **Password**.
 - e. If it is not already filled in, type the password again in the **Confirm Password** field.
 - f. Click **OK**.

Results

The project is now deployed and the Enterprise Applications window is displayed.

What to do next

If you want to set or reset any properties or you would like to cluster adapter project applications, make those changes using the administrative console before configuring troubleshooting tools.

Chapter 7. Administering the adapter module

When you are running the adapter in a stand-alone deployment, use the administrative console of the server to start, stop, monitor, and troubleshoot the adapter module. In an application that uses an embedded adapter, the adapter module starts or stops when the application is started or stopped.

Changing configuration properties for embedded adapters

To change configuration properties after you deploy the adapter as part of a module, you use the administrative console of the runtime environment. You can update resource adapter properties (used for general adapter operation), managed connection factory properties (used for outbound processing), and activation specification properties (used for inbound processing).

Setting resource adapter properties for embedded adapters

To set resource adapter properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

Custom properties are default configuration properties shared by all WebSphere adapters.

To configure properties using the administrative console, use the following procedure.

Procedure

1. Start the administrative console.
2. Under **Applications**, select **Enterprise Applications**.
3. From the **Enterprise Applications** list, click the name of the adapter module whose properties you want to change. The Configuration page is displayed.

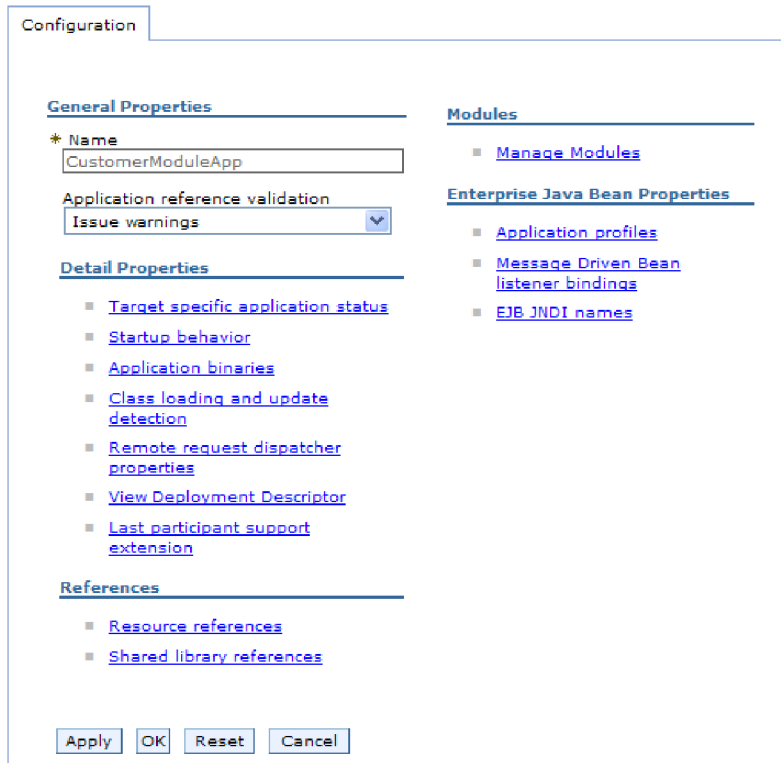


Figure 74. The Manage Modules selection in the Configuration tab

4. Under **Modules**, click **Manage Modules**.
5. Click **IBM WebSphere Adapter for FTP**.
6. From the **Additional Properties** list, click **Resource Adapter**.
7. On the next page, from the **Additional Properties** list, click **Custom properties**.
8. For each property you want to change, perform the following steps.

Note: See “Resource adapter properties” on page 158 for more information about these properties.

- a. Click the name of the property. The **Configuration** page for the selected property is displayed. For example, if you click the **logNumberOfFiles** property, you see the page shown in the figure below.

The screenshot shows a configuration window titled 'Configuration' with a tab labeled 'Configuration'. Under the 'General Properties' section, there are several fields:

- * Scope:** A text box containing 'widNode'.
- Required:** An unchecked checkbox.
- Name:** A text box containing 'logNumberOfFiles'.
- Value:** A text box containing '1'.
- Description:** An empty text area with vertical scrollbars.
- Type:** A dropdown menu showing 'java.lang.String'.

 At the bottom of the form are four buttons: 'Apply', 'OK', 'Reset', and 'Cancel'.

Figure 75. The Configuration tab for the logNumberOfFiles property

- b. Change the contents of the **Value** field or type a value, if the field is empty. You can change the number in the **Value** field and add a description of the property.
 - c. Click **OK**.
9. Click the **Save** link in the **Messages** box at the top of the window.

Results

The resource adapter properties associated with your adapter module are changed.

Setting managed (J2C) connection factory properties for embedded adapters

To set managed connection factory properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You use managed connection factory properties to configure the target FTP server instance.

Note: In the administrative console, the properties are referred to as "J2C connection factory properties."

To configure properties using the administrative console, use the following procedure.

Procedure

1. Start the administrative console.
2. Under **Applications**, select **Enterprise Applications**.
3. In the **Enterprise Applications** list, click the name of the adapter module whose properties you want to change.
4. Under **Modules**, click **Manage Modules**.

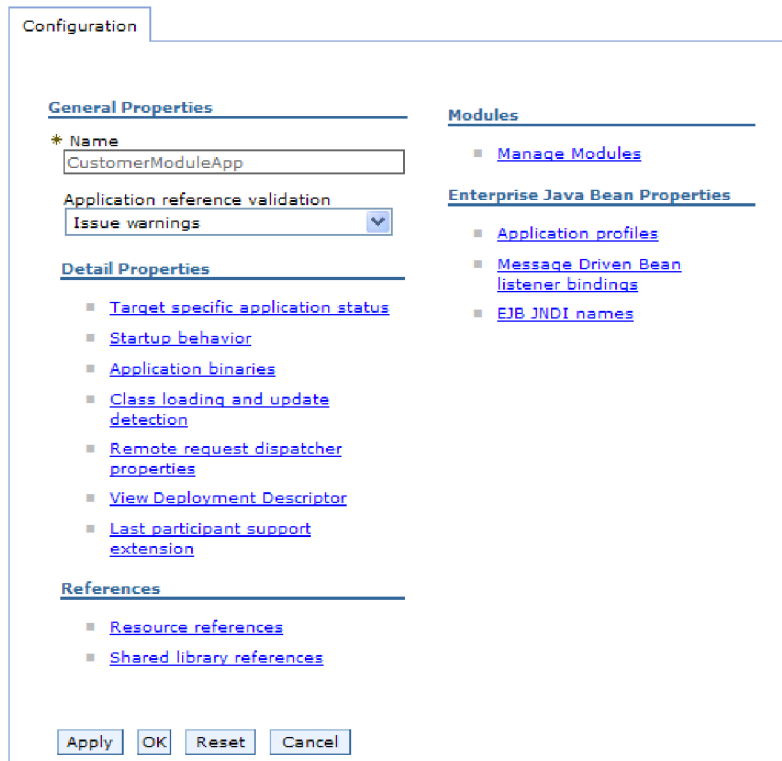


Figure 76. The Manage Modules selection in the Configuration tab

5. Click **IBM WebSphere Adapter for FTP**.
6. In the **Additional Properties** list, click **Resource Adapter**.
7. On the next page, from the **Additional Properties** list, click **J2C connection factories**.
8. Click the name of the connection factory associated with your adapter module.
9. In the **Additional Properties** list, click **Custom properties**.
Custom properties are those J2C connection factory properties that are unique to Adapter for FTP. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
10. For each property you want to change, perform the following steps.

Note: See “Managed (J2C) connection factory properties” on page 162 for more information about these properties.

- a. Click the name of the property.

- b. Change the contents of the **Value** field or type a value, if the field is empty.
 - c. Click **OK**.
11. Click the **Save** link in the **Messages** box at the top of the window.

Results

The managed connection factory properties associated with your adapter module are changed.

Setting activation specification properties for embedded adapters

To set activation specification properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the message endpoint property you want to configure, and then change or set the value.

Before you begin

Your adapter module must be deployed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You use activation specification properties to configure the endpoint for inbound processing.

To configure properties using the administrative console, use the following procedure.

Procedure

1. Start the administrative console.
2. Under **Applications**, select **Enterprise Applications**.
3. From the **Enterprise Applications** list, click the name of the adapter module whose properties you want to change.
4. Under **Modules**, click **Manage Modules**.

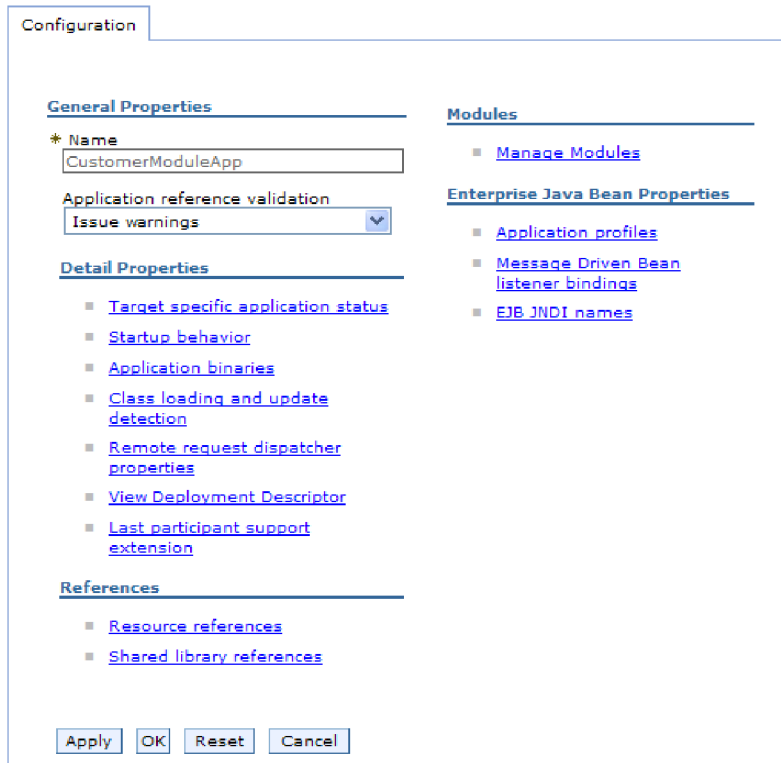


Figure 77. The Manage Modules selection in the Configuration tab

5. Click **IBM WebSphere Adapter for FTP**.
6. From the **Additional Properties** list, click **Resource Adapter**.
7. On the next page, from the **Additional Properties** list, click **J2C activation specifications**.
8. Click the name of the activation specification associated with the adapter module.
9. From the **Additional Properties** list, click **J2C activation specification custom properties**.
10. For each property you want to change, perform the following steps.

Note: See “Activation specification properties” on page 189 for more information about these properties.

- a. Click the name of the property.
 - b. Change the contents of the **Value** field or type a value, if the field is empty.
 - c. Click **OK**.
11. Click the **Save** link in the **Messages** box at the top of the window.

Results

The activation specification properties associated with your adapter module are changed.

Changing configuration properties for stand-alone adapters

To set configuration properties after you install a stand-alone adapter, you use the administrative console of the runtime environment. You provide general information about the adapter and then set resource adapter properties (which are used for general adapter operation). If the adapter will be used for outbound operations, you create a connection factory and then set properties for it. If the adapter will be used for inbound operations, you create an activation specification and then set properties for it.

Setting resource adapter properties for stand-alone adapters

To set resource adapter properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

Custom properties are default configuration properties shared by all WebSphere adapters.

To configure properties using the administrative console, use the following procedure.

Procedure

1. Start the administrative console.
2. Click **Resources** → **Resource Adapters** → **Resource adapters**.
3. In the Resource adapters page, click **IBM WebSphere Adapter for FTP**.
4. In the **Additional Properties** list, click **Custom properties**.
5. For each property you want to change, perform the following steps.

Note: See “Resource adapter properties” on page 158 for more information about these properties.

- a. Click the name of the property.
- b. Change the contents of the **Value** field or type a value, if the field is empty.

For example, if you click **logNumberOfFiles**, you see the following page:

The image shows a configuration dialog box titled "Configuration" with a tab labeled "Configuration". Under the heading "General Properties", there are several fields:

- * Scope:** A text box containing "widNode".
- Required:** An unchecked checkbox.
- Name:** A text box containing "logNumberOfFiles".
- Value:** A text box containing "1".
- Description:** A large empty text area with scroll bars.
- Type:** A dropdown menu showing "java.lang.String".

 At the bottom of the dialog are four buttons: "Apply", "OK", "Reset", and "Cancel".

Figure 78. The Configuration tab for the logNumberOfFiles property

You can change the number in the **Value** field and add a description of the property.

- c. Click **OK**.
6. Click **Save** in the **Messages** box at the top of the page.

Results

The resource adapter properties associated with your adapter are changed.

Setting managed (J2C) connection factory properties for stand-alone adapters

To set managed connection factory properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You use managed connection factory properties to configure the target FTP server instance.

Note: In the administrative console, the properties are referred to as "J2C connection factory properties."

To configure properties using the administrative console, use the following procedure.

Procedure

1. Start the administrative console.
2. Click **Resources** → **Resource Adapters** → **Resource adapters**.
3. In the Resource adapters page, click **IBM WebSphere Adapter for FTP**.
4. In the **Additional Properties** list, click **J2C connection factories**.
5. If you are going to use an existing connection factory, skip ahead to select from the list of existing connection factories.

Note: If you selected **Use predefined connection properties** when you used the external service wizard to configure the adapter module, you do not need to create a connection factory.

If you are creating a connection factory, perform the following steps:

- a. Click **New**.
- b. In the **General Properties** section of the **Configuration** tab, type a name for the connection factory. For example, you could type AdapterCF.
- c. Type a value for **JNDI name**. For example, you could type com/eis/AdapterCF.
- d. Select an authentication alias from the **Component-managed authentication alias** list.
- e. Click **OK**.
- f. Click **Save** in the **Messages** box at the top of the page.

The newly created connection factory is displayed.

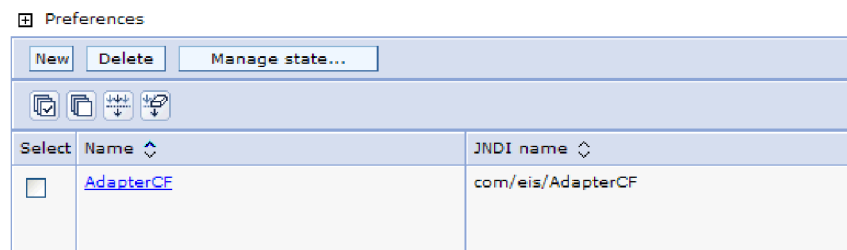


Figure 79. User defined connection factories for use with the resource adapter

6. In the list of connection factories, click the one you want to use.
7. In the **Additional Properties** list, click **Custom properties**.
Custom properties are those J2C connection factory properties that are unique to Adapter for FTP. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
8. For each property you want to change, perform the following steps.

Note: See “Managed (J2C) connection factory properties” on page 162 for more information about these properties.

- a. Click the name of the property.
- b. Change the contents of the **Value** field or type a value, if the field is empty.
- c. Click **OK**.

9. After you have finished setting properties, click **Apply**.
10. Click **Save** in the **Messages** box at the top of the window.

Results

The managed connection factory properties associated with your adapter are set.

Setting activation specification properties for stand-alone adapters

To set activation specification properties for your stand-alone adapter after it has been installed on WebSphere Process Server or WebSphere Enterprise Service Bus, use the administrative console. You select the name of the message endpoint property you want to configure, and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Process Server or WebSphere Enterprise Service Bus.

About this task

You use activation specification properties to configure the endpoint for inbound processing.

To configure properties using the administrative console, use the following procedure.

Procedure

1. Start the administrative console.
2. Click **Resources** → **Resource Adapters** → **Resource adapters**.
3. In the Resource adapters page, click **IBM WebSphere Adapter for FTP**.
4. In the **Additional Properties** list, click **J2C activation specifications**.
5. If you are going to use an existing activation specification, skip ahead to select from an existing list of activation specifications.

Note: If you selected **Use predefined connection properties** when you used the external service wizard to configure the adapter module, you do not need to create an activation specification.

If you are creating an activation specification, perform the following steps:

- a. Click **New**.
 - b. In the **General Properties** section of the **Configuration** tab, type a name for the activation specification. For example, you could type AdapterAS.
 - c. Type a value for **JNDI name**. For example, you could type com/eis/AdapterAS.
 - d. Select an authentication alias from the **Authentication alias** list.
 - e. Select a message listener type.
 - f. Click **OK**.
 - g. Click **Save** in the **Messages** box at the top of the page.
The newly created activation specification is displayed.
6. In the list of activation specifications, click the one you want to use.

7. In the Additional Properties list, click **J2C activation specification custom properties**.
8. For each property you want to set, perform the following steps.

Note: See “Activation specification properties” on page 189 for more information about these properties.

- a. Click the name of the property.
 - b. Change the contents of the **Value** field or type a value, if the field is empty.
 - c. Click **OK**.
9. After you have finished setting properties, click **Apply**.
 10. Click **Save** in the **Messages** box at the top of the page.

Results

The activation specification properties associated with your adapter are set.

Starting the application that uses the adapter

Use the administrative console of the server to start an application that uses the adapter. By default, the application starts automatically when the server starts.

About this task

Use this procedure to start the application, whether it is using an embedded or a stand-alone adapter. For an application that uses an embedded adapter, the adapter starts when the application starts. For an application that uses a stand-alone adapter, the adapter starts when the application server starts.

Procedure

1. On the administrative console, click **Applications** → **Enterprise Applications**.

Note: The administrative console is labeled “Integrated Solutions Console”.

2. Select the check box of the application that you want to start. The application name is the name of the EAR file you installed, without the .EAR file extension.
3. Click **Start**.

Results

The status of the application changes to Started, and a message stating that the application has started displays at the top of the administrative console.

Stopping the application that uses the adapter

Use the administrative console of the server to stop an application that uses the adapter. By default, the application stops automatically when the server stops.

About this task

Use this procedure to stop the application, whether it is using an embedded or a stand-alone adapter. For an application with an embedded adapter, the adapter stops when the application stops. For an application that uses a stand-alone adapter, the adapter stops when the application server stops.

Procedure

1. On the administrative console, click **Applications** → **Enterprise Applications**.

Note: The administrative console is labeled “Integrated Solutions Console”.

2. Select the check box of the application that you want to stop. The application name is the name of the EAR file you installed, without the .EAR file extension.
3. Click **Stop**.

Results

The status of the application changes to Stopped, and a message stating that the application has stopped displays at the top of the administrative console.

Monitoring performance using Performance Monitoring Infrastructure

Performance Monitoring Infrastructure (PMI) is a feature of the administrative console that allows you to dynamically monitor the performance of components in the production environment, including the adapter for FTP. PMI collects adapter performance data, such as average response time and total number of requests, from various components in the server and organizes the data into a tree structure. You can view the data through the Tivoli® Performance Viewer, a graphical monitoring tool that is integrated with the administrative console in WebSphere Process Server.

About this task

You can monitor the performance of your adapter by having PMI collect data at the following points:

- At outbound processing to monitor outbound requests
- At inbound event retrieval to monitor the retrieval of an event from the event table
- At inbound event delivery to monitor the delivery of an event to the endpoint or endpoints

Before you can enable and configure PMI for your adapter, you must first set the level of tracing detail and run some events from which to gather performance data.

To learn more about how PMI can help you monitor and improve the overall performance of your adapter environment, search for PMI on the WebSphere Application Server web site: <http://www.ibm.com/software/webservers/appserv/was/library/>.

Configuring Performance Monitoring Infrastructure

You can configure Performance Monitoring Infrastructure (PMI) to gather adapter performance data, such as average response time and total number of requests. After you configure PMI for your adapter, you can monitor the adapter performance using Tivoli Performance viewer.

Before you begin

Before you can configure PMI for your adapter, you must first set the level of tracing detail and run some events from which to gather performance data.

1. To enable tracing and to receive event data, the trace level must be set to either fine, finer, finest, or all. After *=info, add a colon and a string, for example:

```
*=info: WBILocationMonitor.CEI.ResourceAdapter.  
*=finest: WBILocationMonitor.LOG.ResourceAdapter.*=finest:
```

For instructions on setting the trace level, refer to “Enabling tracing with the Common Event Infrastructure (CEI)” on page 136.

2. Generate at least one outbound request or inbound event to produce performance data that you can configure.

Procedure

1. Enable PMI for your adapter.
 - a. In the administrative console, expand **Monitoring and Tuning**, and then select **Performance Monitoring Infrastructure (PMI)**.
 - b. From the list of servers, click the name of your server.
 - c. Select the Configuration tab, and then select the **Enable Performance Monitoring (PMI)** check box.
 - d. Select **Custom** to selectively enable or disable statistics.

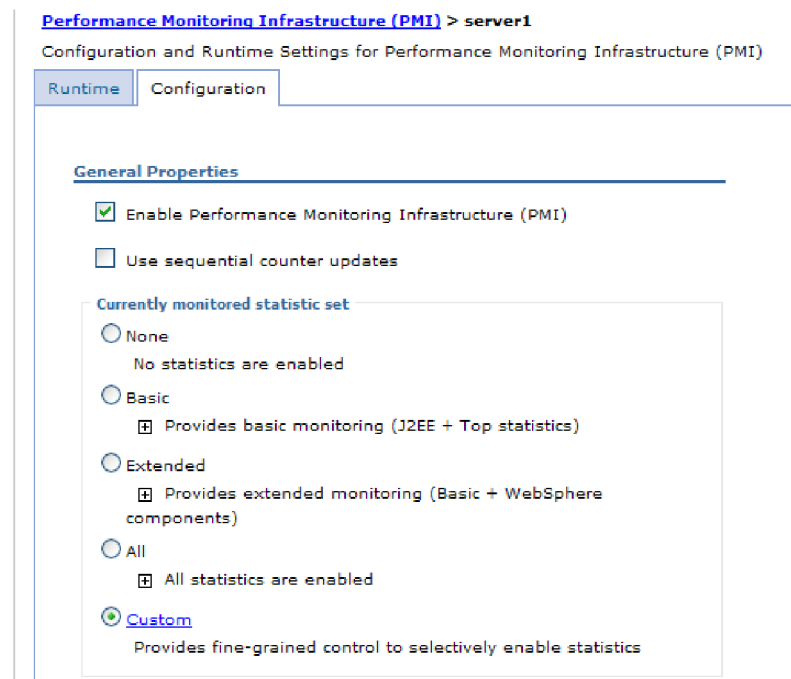


Figure 80. Enabling Performance Monitoring Infrastructure

- e. Click **Apply** or **OK**.
 - f. Click **Save**. PMI is now enabled.
2. Configure PMI for your adapter.
 - a. In the administrative console, expand **Monitoring and Tuning**, and then select **Performance Monitoring Infrastructure (PMI)**.
 - b. From the list of servers, click the name of your server.
 - c. Select **Custom**.
 - d. Select the **Runtime** tab. The following figure shows the Runtime tab.

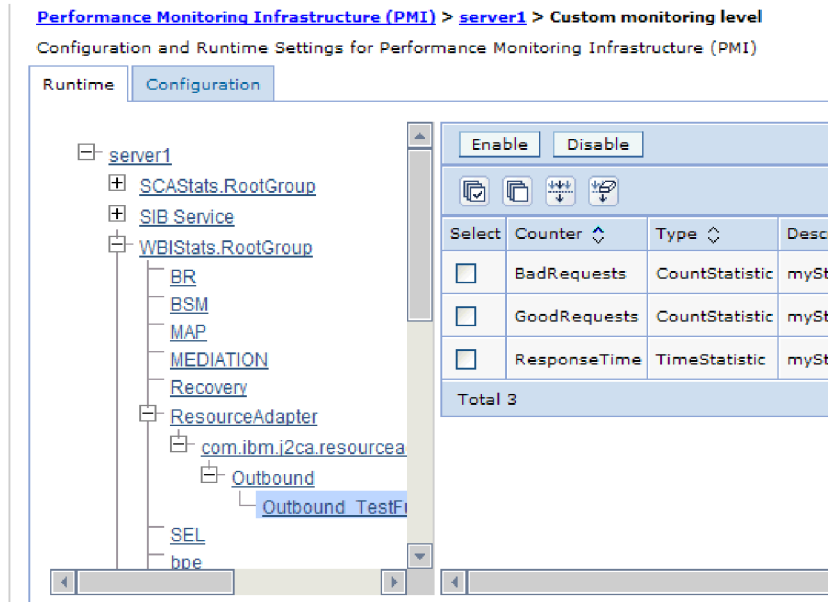


Figure 81. Runtime tab used for configuring PMI

- e. Click **WBISStats.RootGroup**. This is a PMI sub module for data collected in the root group. This example uses the name WBISStats for the root group.
- f. Click **ResourceAdapter**. This is a sub module for the data collected for the JCA adapters.
- g. Click the name of your adapter, and select the processes you want to monitor.
- h. In the right pane, select the check boxes for the statistics you want to gather, and then click **Enable**.

Results

PMI is configured for your adapter.

What to do next

Now you can view the performance statistics for your adapter.

Enabling tracing with the Common Event Infrastructure (CEI)

The adapter can use the Common Event Infrastructure, a component embedded in the server, to report data about critical business events such as the starting or stopping of a poll cycle. Event data can be written to a database or a trace log file depending on configuration settings.

About this task

Procedure

1. In the administrative console, click **Troubleshooting**.
2. Click **Logs and Trace**.
3. In the list of servers, click the name of your server.

4. In the **Change Log Detail Levels** box, click the name of the CEI database (for example, `WBIEventMonitor.CEI.ResourceAdapter.*`) or the trace log file (for example, `WBIEventMonitor.LOG.ResourceAdapter.*`) to which you want the adapter to write event data.
5. Select the level of detail about business events that you want the adapter to write to the database or trace log file, and (optionally) adjust the granularity of detail associated with messages and traces.
 - **No Logging.** Turns off event logging.
 - **Messages Only.** The adapter reports an event.
 - **All Messages and Traces.** The adapter reports details about an event.
 - **Message and Trace Levels.** Settings for controlling the degree of detail the adapter reports about the business object payload associated with an event. If you want to adjust the detail level, choose one of the following:
 - Fine.** The adapter reports the event but none of the business object payload.
 - Finer.** The adapter reports the event and the business object payload description.
 - Finest.** The adapter reports the event and the entire business object payload.
6. Click **OK**.

Results

Event logging is enabled. You can view CEI entries in the trace log file or by using the Common Base Event Browser within the administrative console.

Viewing performance statistics

You can view adapter performance data through the graphical monitoring tool, Tivoli Performance Viewer. Tivoli Performance Viewer is integrated with the administrative console in WebSphere Process Server.

Before you begin

Configure Performance Monitoring Infrastructure for your adapter.

Procedure

1. In the administrative console, expand **Monitoring and Tuning**, expand **Performance Viewer**, then select **Current Activity**.
2. In the list of servers, click the name of your server.
3. Under your server name, expand **Performance Modules**.
4. Click **WBISStatsRootGroup**.
5. Click **ResourceAdapter** and the name of your adapter module.
6. If there is more than one process, select the check boxes for the processes whose statistics you want to view.

Results

The statistics are displayed in the right panel. You can click **View Graph** to view a graph of the data, or **View Table** to see the statistics in a table format. The following figure shows adapter performance statistics as a graph.

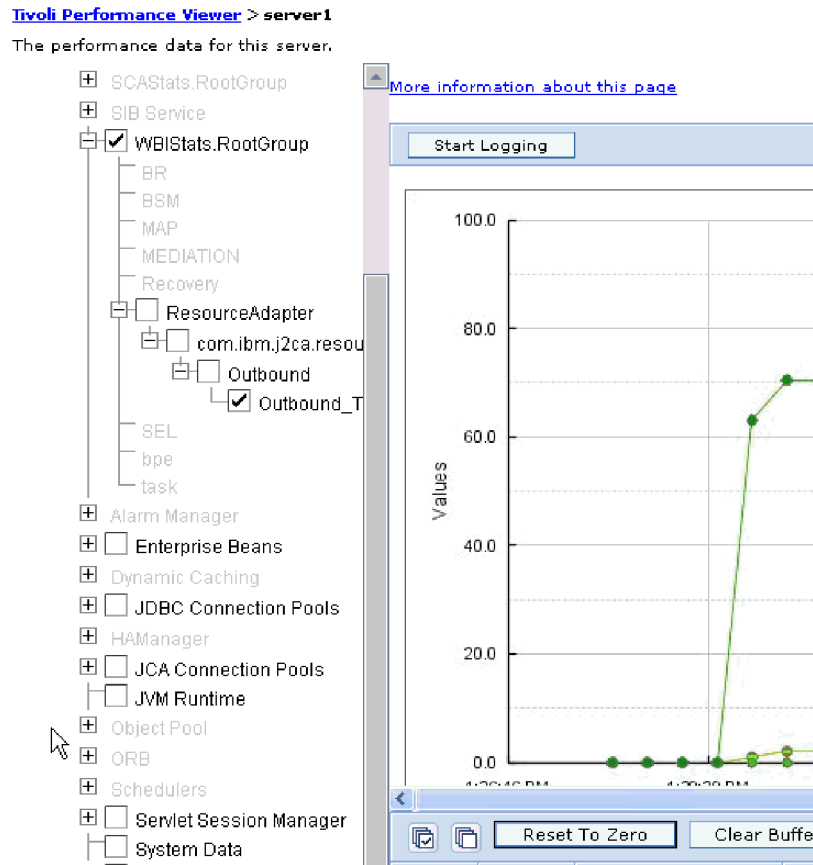


Figure 82. Adapter performance statistics, using graph view

Chapter 8. Troubleshooting and support

Common troubleshooting techniques and self-help information help you identify and solve problems quickly.

Support for the Log and Trace Analyzer

The adapter creates log and trace files that can be viewed with the Log and Trace Analyzer.

The Log and Trace Analyzer can filter log and trace files to isolate the messages and trace information for the adapter. It can also highlight the adapter's messages and trace information in the log viewer.

The adapter's component ID for filtering and highlighting is a string composed of the characters FTPRA plus the value of the adapter ID property. For example, if the adapter ID property is set to 001, the component ID is FTPRA001.

If you run multiple instances of the same adapter, make sure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for

multiple instances of that adapter is also unique, allowing you to correlate log and trace information to a particular instance of an adapter. To illustrate how the length of the adapter ID property affects the filtering of log and trace files, suppose you set the adapter ID property of two instances of WebSphere Adapter for FTP to 001 and 002. The component IDs for those instances, FTPRA001 and FTPRA002, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. For example, suppose you set the adapter ID properties of two instances to Instance01 and Instance02. You will not be able to separately examine the log and trace information for each adapter instance because the component ID for both instances is truncated to FTPRAInstance.

For outbound processing, the adapter ID property is located in both the resource adapter and managed connection factory property groups. If you update the adapter ID property after using the external service wizard to configure the adapter for outbound processing, be sure to set the resource adapter and managed connection factory properties consistently, to prevent inconsistent marking of the log and trace entries. For inbound processing, the adapter ID property is located only in the resource adapter properties, so this consideration does not apply.

For more information about the adapter ID property, see “Adapter ID (AdapterID)” on page 159. For more information about the Log and Trace Analyzer, see <http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/topic/org.eclipse.hyades.log.ui.doc.user/concepts/cltviews.htm>.

Handling Leap Year Exception

The default parser provided by Commons Net API v1.4.1 fails to handle files that have a creation date of February 29th. You can use the `com.ibm.j2ca.ftp.util.FTPLongListEntryParser` class to handle files that are created with the date of February 29.

When the directory contains files created on February 29, use the `com.ibm.j2ca.ftp.util.FTPLongListEntryParser` class. This class is a basic implementation that retrieves the file name and continues to process without any errors. You can implement the interface for complete parsing of the long list output.

Note: When you use this class to retrieve files, the time stamp of the file on the FTP server is lost.

ServerToServerFileTransfer

The specified file is transferred from one FTP server directory to another FTP server directory.

If the value of the Host name property is set to `localhost`, the first server is located on the same system as the adapter workstation. Adapter for FTP will generate the following error: 421 error-Can't open data connection. To work around this problem, edit the hosts file (which, for the Windows platform, is located at `<WindowsHome>/system32/drivers/etc/hosts`) and add a new entry with the external IP address, for example, `9.186.116.151 localhost`.

The adapter will also work if the host name value or the external IP address is used for example, if `FTPTEST` is used as the host name format or `9.186.116.151` is used as the IP format.

Configuring logging and tracing

Configure logging and tracing to suit your requirements. Enable logging for the adapter to control the status of event processing. Change the adapter log and trace file names to separate them from other log and trace files.

About this task

Configuring logging properties

Use the administrative console to enable logging and to set the output properties for a log, including the location, level of detail, and output format of the log.

About this task

Before the adapters can log monitored events, you must specify the service component event points that you want to monitor, what level of detail you require for each event, and format of the output used to publish the events to the logs. Use the administrative console to perform the following tasks:

- Enable or disable a particular event log
- Specify the level of detail in a log
- Specify where log files are stored and how many log files are kept
- Specify the format for log output

If you set the output for log analyzer format, you can open trace output using the Log Analyzer tool, which is an application included with your process server. This is useful if you are trying to correlate traces from two different server processes, because it allows you to use the merge capability of the Log Analyzer.

For more information about monitoring on a process server, including service components and event points, see the documentation for your process server.

You can change the log configuration statically or dynamically. Static configuration takes effect when you start or restart the application server. Dynamic, or runtime, configuration changes apply immediately.

When a log is created, the detail level for that log is set from the configuration data. If no configuration data is available for a particular log name, the level for that log is obtained from the parent of the log. If no configuration data exists for the parent log, the parent of that log is checked, and so on up the tree, until a log with a non-null level value is found. When you change the level of a log, the change is propagated to the children of the log, which recursively propagate the change to their children, as necessary.

To enable logging and set the output properties for a log, use the following procedure.

Procedure

1. In the navigation pane of the administrative console, click **Servers** → **Application Servers**.
2. Click the name of the server that you want to work with.
3. Under **Troubleshooting**, click **Logs and trace**.
4. Click **Change Log Detail Levels**.
5. Specify when you want the change to take effect:

- For a static change to the configuration, click the **Configuration** tab.
 - For a dynamic change to the configuration, click the **Runtime** tab.
6. Click the names of the packages whose logging level you want to modify. The package names for WebSphere Adapters start with **com.ibm.j2ca.***:
 - For the adapter base component, select **com.ibm.j2ca.base.***.
 - For the adapter base component and all deployed adapters, select **com.ibm.j2ca.***.
 - For the Adapter for FTP only, select the **com.ibm.j2ca.ftp.*** package.
 7. Select the logging level.

Logging Level	Description
Fatal	The task cannot continue or the component cannot function.
Severe	The task cannot continue, but the component can still function. This logging level also includes conditions that indicate an impending fatal error, that is, situations that strongly suggest that resources are on the verge of being depleted.
Warning	A potential error has occurred or a severe error is impending. This logging level also includes conditions that indicate a progressive failure, for example, the potential leaking of resources.
Audit	A significant event has occurred that affects the server state or resources.
Info	The task is running. This logging level includes general information outlining the overall progress of a task.
Config	The status of a configuration is reported or a configuration change has occurred.
Detail	The subtask is running. This logging level includes general information detailing the progress of a subtask.

8. Click **Apply**.
9. Click **OK**.
10. To have static configuration changes take effect, stop and then restart the process server.

Results

Log entries from this point forward contain the specified level of information for the selected adapter components.

Changing the log and trace file names

To keep the adapter log and trace information separate from other processes, use the administrative console to change the file names. By default, log and trace information for all processes and applications on a process server is written to the SystemOut.log and trace.log files, respectively.

Before you begin

You can change the log and trace file names at any time after the adapter module has been deployed to an application server.

About this task

You can change the log and trace file names statically or dynamically. Static changes take effect when you start or restart the application server. Dynamic or run time changes apply immediately.

Log and trace files are in the *install_root/profiles/profile_name/logs/server_name* folder.

To set or change the log and trace file names, use the following procedure.

Procedure

1. In the navigation pane of the administrative console, select **Applications > Enterprise Applications**.
2. In the Enterprise Applications list, click the name of the adapter application. This is the name of the EAR file for the adapter, but without the .ear file extension. For example, if the EAR file is named Accounting_OutboundApp.ear, then click **Accounting_OutboundApp**.
3. In the Configuration tab, in the Modules list, click **Manage Modules**.
4. In the list of modules, click IBM WebSphere Adapter for FTP.
5. In the Configuration tab, under Additional Properties, click **Resource Adapter**.
6. In the Configuration tab, under Additional Properties, click **Custom properties**.
7. In the Custom Properties table, change the file names.
 - a. Click either **logFilename** to change the name of the log file or **traceFilename** to change the name of the trace file.
 - b. In the Configuration tab, type the new name in the **Value** field. By default, the log file is called SystemOut.log and the trace file is called trace.log.
 - c. Click **Apply** or **OK**. Your changes are saved on your local machine.
 - d. To save your changes to the master configuration on the server, use one of the following procedures:
 - **Static change:** Stop and restart the server. This method allows you to make changes, but those changes do not take effect until you stop and start the server.
 - **Dynamic change:** Click the **Save** link in the Messages box above the Custom properties table. Click **Save** again when prompted. This method allows you to make changes that take effect right away.

First-failure data capture (FFDC) support

The adapter supports first-failure data capture (FFDC), which provides persistent records of failures and significant software incidents that occur during run time in WebSphere Process Server or WebSphere Enterprise Service Bus.

The FFDC feature runs in the background and collects events and errors that occur at run time. The feature provides a means for associating failures to one another, allowing software to link the effects of a failure to their causes, and thereby facilitate the quick location of the root cause of a failure. The data that is captured can be used to identify exception processing that occurred during the adapter run time.

When a problem occurs, the adapter writes exception messages and context data to a log file, which is located in the *install_root/profiles/profile/logs/ffdc* directory.

For more information about first-failure data capture (FFDC), see the WebSphere Process Server or WebSphere Enterprise Service Bus documentation.

Business faults

The adapter supports business faults, which are exceptions that are anticipated and declared in the outbound service description, or import. Business faults occur at predictable points in a business process, and are caused by a business rule violation or a constraint violation.

Although WebSphere Process Server and WebSphere Enterprise Service Bus support other types of faults, the adapter generates only business faults, which are called simply *faults* in this documentation. Not all exceptions become faults. Faults are used only when the outbound operations are configured with response type. Faults are generated for errors that are actionable, that is, errors that can have a recovery action that does not require the termination of the application. For example, the adapter generates a fault when it receives a business object for outbound processing that does not contain the required data or when the adapter encounters certain errors during outbound processing.

Note: The faults for a particular operation will be enabled only if that operation has a response configured.

Fault business objects

The external service wizard creates a business object for each fault that the adapter can generate. In addition, the wizard creates a WBIFault superset business object, which has information common to all faults, such as the message, errorCode, and primaryKeySet attributes as shown in Figure 83.

WBIFault	
message	string
errorCode	string
primaryKeySet	PrimaryKeyPairType []

Figure 83. The structure of the WBIFault business object

The WebSphere Adapter for FTP enables faults for you. Manual configuration of faults is not required. The adapter provides the following fault business objects that the wizard creates:

- DuplicateRecordFault
The adapter throws this fault when processing an outbound Create operation when an error occurs because the file specified already exists in the specified directory path. This fault is thrown during ServerToServerFileTransfer Operation also, when the file already exists in the second server directory.
- RecordNotFoundFault
The adapter throws this fault when processing the Create, Append, Delete, Overwrite, Retrieve, ExecuteFTPScript, and ServerToServerFileTransfer operations when the file directory path or script file does not exist in the specified directory path. This fault occurs when the directory path does not exist and when the sequence file does not exist during the Create operation.
- MissingDataFault
The adapter throws this fault when required values are not provided, such as when the file content is null or the file name or directory path is empty.

During a Retrieve operation, the adapter throws this fault when an error occurs because the delimiter is null or not valid. If `splitCriteria` is null or invalid when `Splittingfunctionclassname` is `SplitByDelimiter` and when `LocalDirectoryPath` is null, a `MissingData` fault is thrown with the message that the `LocalDirectoryPath` is missing. The adapter does not throw an exception when `splitCriteria` is null or not valid and `SplitBySize` is configured. During a Retrieve operation, the adapter throws this fault when an error occurs because the delimiter is null and `SplitByDelimiter` is configured.

XAResourceNotAvailableException

When the process server log contains repeated reports of the `com.ibm.ws.Transaction.XAResourceNotAvailableException` exception, remove transaction logs to correct the problem.

Symptom:

When the adapter starts, the following exception is repeatedly logged in the process server log file:

```
com.ibm.ws.Transaction.XAResourceNotAvailableException
```

Problem:

A resource was removed while the process server was committing or rolling back a transaction for that resource. When the adapter starts, it tries to recover the transaction but cannot because the resource was removed.

Solution:

To correct this problem, use the following procedure:

1. Stop the process server.
2. Delete the transaction log file that contains the transaction. Use the information in the exception trace to identify the transaction. This prevents the server from trying to recover those transactions.

Note: In a test or development environment, you can generally delete all of the transaction logs. In WebSphere Integration Developer, delete the files and subdirectories of the transaction log directory, `server_install_directory\profiles\profile_name\tranlog`.

In a production environment, delete only the transactions that represent events that you do not need to process. One way to do this is to reinstall the adapter, pointing it to the original event database used, and deleting only the transactions you do not need. Another approach is to delete the transactions from either the `log1` or `log2` file in the following directory:

```
server_install_directory\profiles\profile_name\tranlog\node_name\wps\  
server_name\transaction\tranlog
```

3. Start the process server.

org.xml.sax.SAXParseException

When the adapter is configured with the XML data handler, an `org.xml.sax.SAXParseException` exception is generated if the content is not in the specified business object format. To correct the problem, make sure the file content matches the business object structure. If the file contains multiple business objects, make sure the delimiter is specified correctly.

Symptom:

When the adapter is configured with the XML data handler, the following exception is thrown:

```
org.xml.sax.SAXParseException: Content is not allowed in trailing section
```

Problem:

The content of the file is not in the specified business object format.

Solution:

To correct this problem, use the following procedure:

1. Make sure the file content matches the business object structure.
2. If the content file contains multiple business objects, make sure the delimiter is specified correctly.

Self-help resources

Use the resources of IBM software support to get the most current support information, obtain technical documentation, download support tools and fixes, and avoid problems with WebSphere Adapters. The self-help resources also help you diagnose problems with the adapter and provide information about how to contact IBM software support.

Support Web site

The WebSphere Adapters software support Web site at <http://www.ibm.com/software/integration/wbiadapters/support/> provides links to many resources to help you learn about, use, and troubleshoot WebSphere Adapters, including the following types of

- Flashes (alerts about the product)
- Technical information including the product information center, manuals, IBM Redbooks[®], and white papers
- Educational offerings
- Technotes

Recommended fixes

A list of recommended fixes you should apply is available at the following location: <http://www.ibm.com/support/docview.wss?fdoc=aimadp&rs=695&uid=swg27010397>

Technotes

Technotes provide the most current documentation about the Adapter for FTP, including the following topics:

- Problems and their currently available solutions
- Answers to frequently asked questions
- How-to information about installing, configuring, using, and troubleshooting the adapter
- *IBM Software Support Handbook*

For a list of technotes for WebSphere Adapters, visit this address:

<http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm>

Plug-in for IBM Support Assistant

Adapter for FTP provides a plug-in for IBM Support Assistant, which is a free, local software serviceability workbench. The plug-in supports the dynamic trace feature. For information about installing or using IBM Support Assistant, visit this address:

<http://www.ibm.com/software/support/isa/>

Chapter 9. Reference information

To support you in your tasks, reference information includes details about business objects that are generated by the external service wizard and information about adapter properties, including those that support bidirectional transformation. It also includes pointers to adapter messages and related product information.

Business object information

You can determine the purpose of a business object by examining both the application-specific information within the business object definition file and the name of the business object. The application-specific information dictates what operations can be performed on the FTP server. The name typically reflects the operation to be performed and the structure of the business object.

Business object structure

The adapter supports three different types of business object structures. A generic business object, which is used to pass unstructured data. A generic business object with a business graph, which contains the action to be performed on the data and the connection-specific information. A user-defined type, which is a content-specific business object that supports very specific business object structures (such as customer and order business objects).

Business graphs are optional and can be selected in the external service wizard.

The FTPFileBG, FTPFile and UnstructuredContent generic business object definitions are automatically generated. Depending on the custom complex types selected when you create external services, the corresponding business object or objects' definitions will also be generated. For example, if you select Customer, including the optional business graph, the CustomerWrapperBG and CustomerWrapper business objects will be generated.

FTPFileBG

The FTPFileBG business object is a generic business object that contains the verb (the action to be performed on the data) and the FTPFile business object as a child. The following graphic illustrates this relationship.

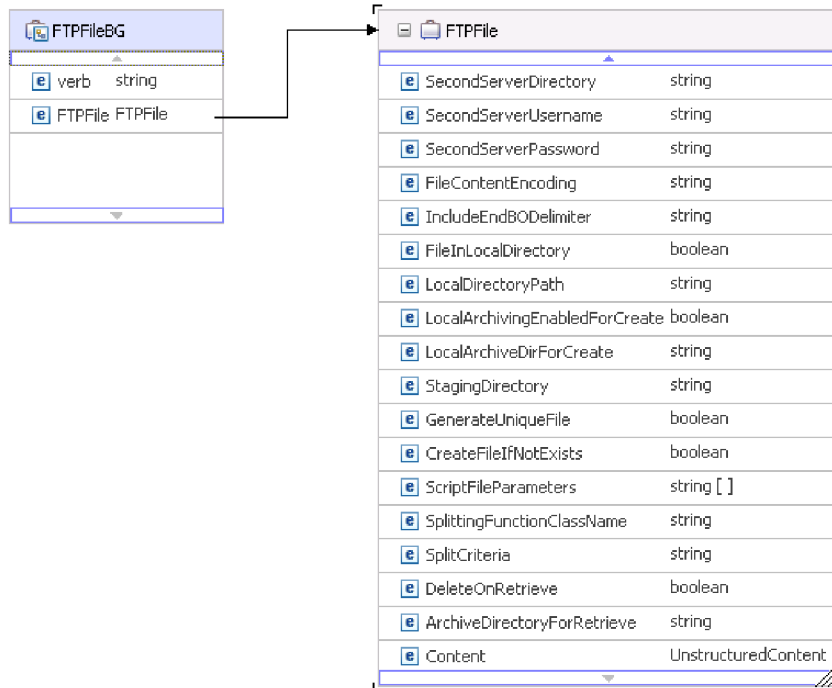


Figure 84. FTPFileBG business object

FTPFile

The FTPFile business object contains all necessary connection information, and an UnstructuredContent business object as a child. The following graphic illustrates this relationship.

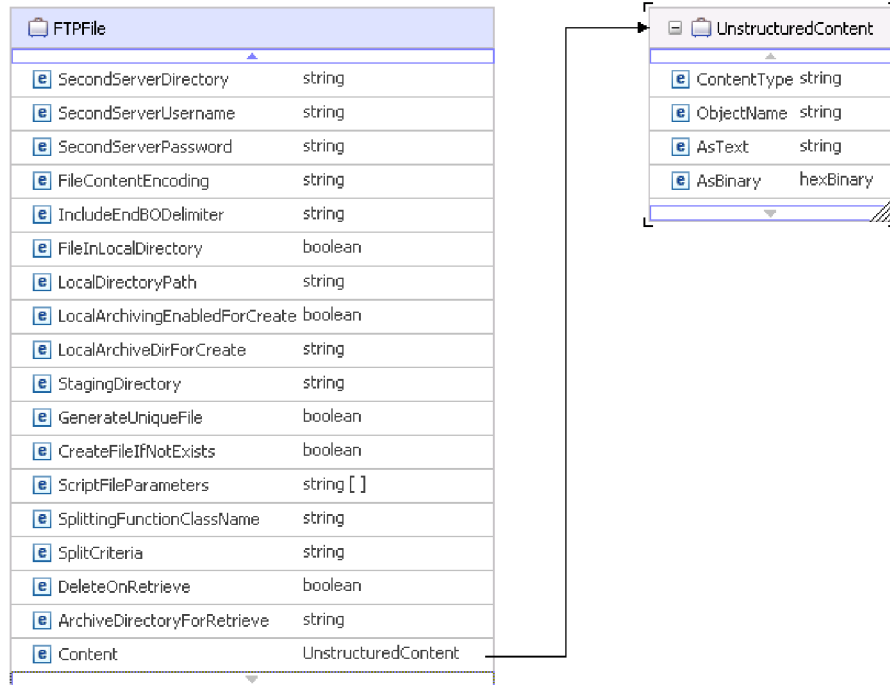


Figure 85. FTPFile business object

CustomerWrapperBG

The CustomerWrapperBG is a business object that contains the verb (the action to be performed on the data) and the CustomerWrapper business object as a child. The following graphic illustrates this relationship.

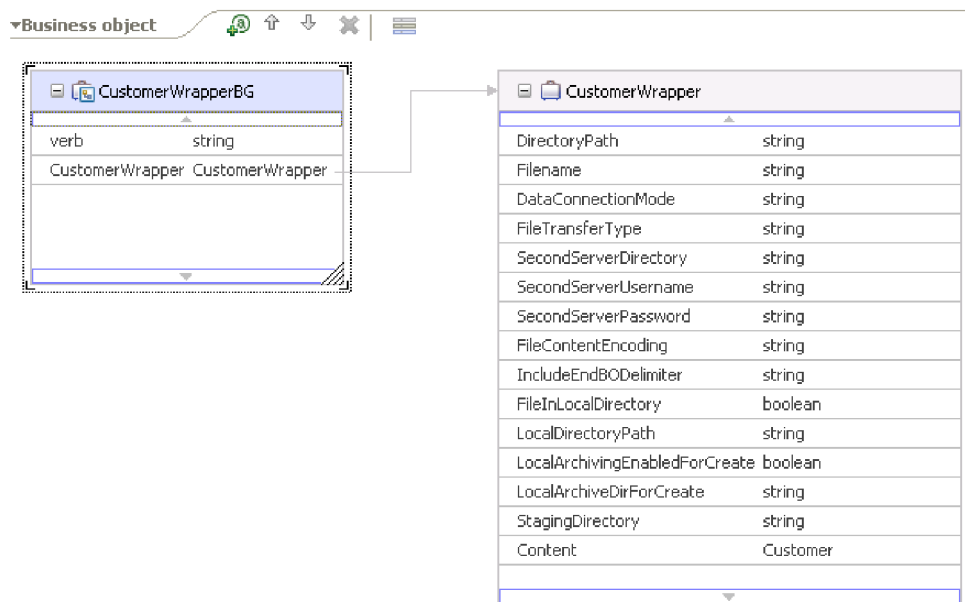


Figure 86. CustomerWrapperBG business object

CustomerWrapper

The CustomerWrapper business object is a business object that contains all necessary connection information and the content-specific Customer business object as a child. The following graphic illustrates this relationship.

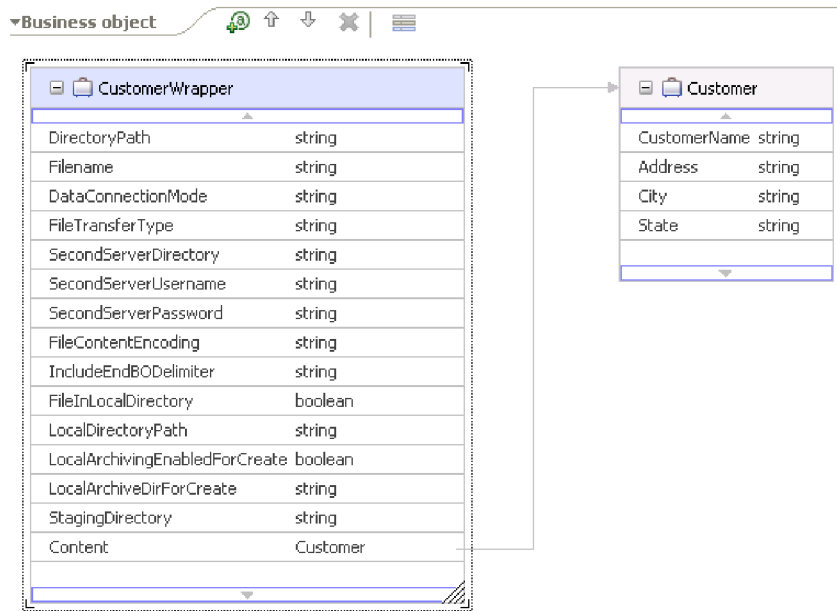


Figure 87. CustomerWrapper business object

Naming conventions

When the external service wizard generates a business object, it provides a name for the business object based on the name of the object in the FTP server that it uses to build the business object. Use the Business Object Editor to create user defined objects.

External service wizard converts the name of the object to mixed case, which means that it removes any separators, such as spaces or underscores, and then capitalizes the first letter of each word. For example, if the external service wizard uses a FTP server object called CUSTOMER_ADDRESS to generate a business object, it generates a business object called CustomerAddress.

The generated business object name can indicate the structure of the business object. However, business objects names have no semantic value to the adapter. This means that if you change the business object name, the behavior of the business object remains the same.

Important: If you choose to rename a business object, use the refactoring functionality in WebSphere Integration Developer to ensure that you update all of the business object dependencies. For instructions on using refactoring to rename business objects, refer to the following link: <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/topic/com.ibm.wbit.help.refactor.doc/topics/trenameboatt.html>.

Note: Business graph generation is optional and is supported for WebSphere Process Server only.

Business object attribute properties

Business object architecture defines various properties that apply to attributes. This section describes how the adapter interprets these properties.

The following table describes these properties and how the adapter interprets them.

Table 8. Business object attribute properties

Property	Description
Cardinality	For simple attributes, 1 is used. For container attributes, depending on the method requirements, n is used.
Foreign Key	The adapter does not have any specific elements representing Foreign Keys.
Key	The adapter does not have any specific elements representing a Key.
Name	This property represents the unique name of the attribute, if it is a simple attribute, or the name of the business object, if it is a child business object.
Required	This property specifies whether an attribute must contain a value.
Type	The attribute type can be either simple or complex. Simple types are: Boolean, String, LongText, Integer, Float, Double and Byte[]. A typical complex type is the name of another business object.

Business object operation support

An operation is the name of the action that is performed on the business object by the adapter. Every business object has an operation associated with it. The name of the operation typically indicates the type of action that is taken on the business object.

The following table defines the operations that the adapter supports.

Table 9. Supported operations of business objects

Operation	Result
Create	Creates a file with the specified file name in the specified directory with the content sent across in the request.
Append	Appends the content in the request to the end of the file.
Retrieve	Returns the content of the file specified in the request.
Delete	Deletes the file from the directory specified in the request.
Overwrite	Overwrites the file in the directory with the content specified in the request.
Exists	Returns a successful response if the file in the request exists in the specified directory or sub directories.
List	Returns all the file names in the specified directory.
ServerToServerFileTransfer	Transfers the file from one FTP server to another FTP server.
ExecuteFTPScript	Runs an FTP script file in the specified directory.

Custom business objects

If you use custom business objects, you must create predefined business objects using WebSphere Integration Developer business object wizard before running the external service wizard. The business object definitions created by the wizard are stored as xsd files on your local system. When the external service wizard creates

business objects, it looks for the predefined business objects created in the business object wizard and populates them with data specific to the module.

For more information on how to create predefined business objects, refer to the WebSphere Integration Developer documentation.

Custom file splitting

You can implement a custom class containing the splitting logic. The adapter provides a Java™ interface for the class. The details of the interface are shown below.

```
public interface SplittingFunctionalityInterface extends Iterator{
    public int getTotalBOs(String filename) throws SplittingException;
    public void setBODetails(String filename, int currentPosition, int totalBOs,
        boolean includeEndBODElimiter) throws SplittingException;
        public void setSplitCriteria(String splitCriteria);
        public void setEncoding(String encoding);
        public void setLogUtils(LogUtils logUtils);
    public boolean isSplitBySize()
}
```

- `public int getTotalBOs(String filename) throws SplittingException`
This method returns the total number of business object's present in the event file given by filename.
- `public void setSplitCriteria(String splitCriteria)`
This method takes the `splitCriteria`, which is based on the number of business object's in the event file. Each business object is returned during the `next()` call.
- `public void setLogUtils(LogUtils logUtils)`
This method is used to set the `LogUtils` object, which is the class that the user can use to write trace and log messages to the files.
- `public void setEncoding(String encoding)`
This method is used to set the encoding of the event file content. This encoding is used while reading the file content. This encoding is also used for the `SplitCriteria`.
- `public void setBODetails(String filename, int currentPosition, int totalBOs, boolean includeEndBODElimiter) throws SplittingException`
This method is used to set the current business object number so that whenever a `next()` call is made, the business object number set in the `currentPosition` is returned. It also takes an `includeEndBODElimiter` parameter, which when set to `true`, includes the `SplitCriteria` at the end of the business object content. This method must be called before every `next()` call so that the `next()` method returns the business object content for the business object set in this method.
- The iterator has 3 methods: `hasNext()`, `next` and `remove()`, which also need to be implemented. The `next()` method returns the business object content (as a `byte[]`) for the business object position set in `setBODetails()`. If the business object position is not set, it fails. The `hasNext()` method indicates whether the business object position set in the `setBODetails()` exists or not. Before a `hasNext()` call, the `setBODetails()` method must be called. The `remove()` method is called for each of the business object entries being deleted from the Event persistence table. Do not delete the event file in this method. Only clean up resources that are being used.
- `public boolean isSplitBySize()`
This method indicates whether the event file is parsed based on size or based on delimiter.

Outbound configuration properties

WebSphere Adapter for FTP has several categories of outbound connection configuration properties, which you set with the external service wizard while generating or creating objects and services. You can change the resource adapter and managed connection factory properties after you deploy the module to WebSphere Process Server or WebSphere Enterprise Service Bus using WebSphere Integration Developer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

Guide to information about properties

The properties used to configure WebSphere Adapter for FTP are described in detail in tables included in each of the configuration properties topics, such as Resource adapter properties, Managed connection factory properties, and so on. To help you use these tables, information about each row you might see is explained here.

The following table explains the meaning of each row that might be displayed in the table for a configuration property.

Row	Explanation
Required	<p>A required field (property) must have a value in order for the adapter to work. Sometimes the external service wizard provides a default value for required properties.</p> <p>Removing a default value from a required field on the external service wizard <i>will not change that default value</i>. When a required field contains no value at all, the external service wizard will process the field using its assigned default value, and that default value will also be displayed on the administrative console.</p> <p>Possible values are Yes and No.</p> <p>Sometimes a property is required only when another property has a specific value. When this is the case, the table will note this dependency. For example,</p> <ul style="list-style-type: none">• Yes, when the EventQueryType property is set to Dynamic• Yes, for Oracle databases
Possible values	Lists and describes the possible values that you can select for the property.
Default	<p>The predefined value that is set by the external service wizard. When the property is required, you must either accept the default value or specify one yourself. If a property has no default value, the table will state No default value.</p> <p>The word None is an acceptable default value, and does not mean that there is no default value.</p>
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	<p>Describes the property type. Valid property types include the following:</p> <ul style="list-style-type: none">• Boolean• String• Integer

Row	Explanation
Usage	<p>Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented:</p> <p>For Rational® Application Developer for WebSphere Software version 6.40 or earlier, the password:</p> <ul style="list-style-type: none"> • Must be uppercase • Must be 8 characters in length <p>For versions of Rational Application Developer for WebSphere Software later than 6.40, the password:</p> <ul style="list-style-type: none"> • Is not case sensitive • Can be up to 40 characters in length. <p>This section lists other properties that affect this property or the properties that are affected by this property and describes the nature of the conditional relationship.</p>
Example	<p>Provides sample property values, for example:</p> <p>"If Language is set to JA (Japanese), Codepage number is set to 8000".</p>
Globalized	<p>If a property is globalized, it has national language support, meaning that you can set the value in your national language.</p> <p>Valid values are Yes and No.</p>
Bidi supported	<p>Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing pertains to the task of processing data that contains both left-to-right (Hebrew or Arabic, for example) and right-to-left (a URL or file path, for example) semantic content within the same file.</p> <p>Valid values are Yes and No.</p>

Adapter type properties

Adapter type properties provide the external service wizard with the adapter details. These properties are configured using the external service wizard before deployment or with the WebSphere Application Server administrative console after deployment.

Note: If you set any of these adapter type properties using bidirectional script, you must set values that identify the format of the bidirectional script entered for that property.

The adapter type properties and their purpose are described in the following table. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see "Guide to information about properties" on page 155.

Table 10. Adapter type properties

Property name		Description
In the wizard	In the administrative console	
"Description property (Description)" on page 157	Description	Adapter description.

Table 10. Adapter type properties (continued)

“Display Name property (DisplayName)”	DisplayName	Adapter display name.
“ID property (ID)”	ID	ID for the adapter type.
“Vendor property (Vendor)”	Vendor	Name of the vendor providing the adapter.
“Version property (Version)” on page 158	Version	Adapter version.

Description property (Description)

Adapter description.

Table 11. Description property characteristics

Required	Yes
Default	IBM WebSphere Adapter for FTP
Property type	String

Display Name property (DisplayName)

Adapter display name.

Table 12. DisplayName property characteristics

Required	Yes
Default	IBM WebSphere Adapter for FTP
Property type	String

ID property (ID)

ID for the adapter type.

Table 13. ID property characteristics

Required	Yes
Default	FTP
Property type	String

Vendor property (Vendor)

Name of the vendor providing the adapter.

Table 14. Vendor property characteristics

Required	Yes
Default	IBM
Property type	String

Version property (Version)

Adapter version.

Table 15. Version property characteristics

Required	Yes
Default	6.2
Property type	String

Resource adapter properties

The resource adapter properties control the general operation of the adapter, such as specifying the namespace for business objects. You set the resource adapter properties using the external service wizard when you configure the adapter. After deploying the adapter, use the administrative console to change these properties.

The following properties for logging and tracing are no longer required in version 6.1.x, but are supported for compatibility with previous versions:

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

The following table lists the resource adapter properties and their purpose. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Table 16. Resource adapter properties for the Adapter for FTP

Property name		Description
In the wizard	In the administrative console	
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing
“EISEncoding (EISEncoding)” on page 159	EISEncoding	Encoding of the FTP server.
(Not available)	enableHASupport	Do not change this property.
“Disguise user data as “XXX” in log and trace files (HideConfidentialTrace) property” on page 160	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing strings of X’s instead of user data in log and trace files
(Not available)	LogFileMaxSize	Deprecated
(Not available)	LogFilename	Deprecated
(Not available)	LogNumberOfFiles	Deprecated
(Not available)	TraceFileMaxSize	Deprecated
(Not available)	TraceFileName	Deprecated
(Not available)	TraceNumberOfFiles	Deprecated

Adapter ID (AdapterID)

This property identifies a specific deployment, or instance, of the adapter.

Table 17. Adapter ID details

Required	Yes
Default	001
Property type	String
Usage	<p>This property identifies the adapter instance in log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, FTPRA, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to 001, the component ID is FTPRA001.</p> <p>If you run multiple instances of the same adapter, make sure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate log and trace information to a particular instance of an adapter.</p> <p>To illustrate how the length of the adapter ID property affects the filtering of log and trace files, suppose you set the adapter ID property of two instances of WebSphere Adapter for FTP to 001 and 002. The component IDs for those instances, FTPRA001 and FTPRA002, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. For example, suppose you set the adapter ID properties of two instances to Instance01 and Instance02. You will not be able to separately examine the log and trace information for each adapter instance because the component ID for both instances is truncated to FTPRAInstance.</p> <p>For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, can be set both at the resource adapter level and the managed connection factory level. After using the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, make sure you set them consistently, to prevent inconsistent marking of the log and trace entries.</p>
Globalized	Yes
Bidi supported	No

EISEncoding (EISEncoding)

This property specifies the encoding of the FTP server. Sets the encoding for the control connection while communicating with the FTP server. Set the property if the FTP server's directories or file names contain globalized characters.

Table 18. EISEncoding characteristics

Required	No
Default	None
Property type	String
Examples	UTF-8, ISO-8859-1

Enable high availability support (enableHASupport)

Do not change this property. It must be set to true.

Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) property

This property specifies whether to replace user data in log and trace files with a string of X's to prevent unauthorized disclosure of potentially sensitive data.

Table 19. Disguise user data as "XXX" in log and trace files details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	If you set this property to True, the adapter replaces user data with a string of X's when writing to log and trace files. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, can be set both at the resource adapter level and the managed connection factory level. After using the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, make sure you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	No
Bidi supported	No

Log file maximum size (LogFileMaxSize)

This property specifies the size of the log files in kilobytes.

Table 20. Log file maximum size details

Required	No
Default	0
Property type	Integer
Usage	When the log file reaches it maximum size, the adapter start using a new log file. If the file size is specified as 0 or no maximum size is specified, the file does not have a maximum size.
Globalized	Yes
Bidi supported	No

Log file name (LogFilename)

This property specifies the full path name of the log file.

Table 21. Log file name details

Required	No
Default	No default value
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Log number of files (LogNumberOfFiles)

This property specifies the number of log files.

Table 22. Log number of files details

Required	No
Default	1
Property type	Integer
Usage	When a log file reaches its maximum size, the adapter starts using another log file. If no value is specified, the adapter creates a single log file.
Globalized	Yes
Bidi supported	No

Trace file maximum size (TraceFileMaxSize)

This property specifies the size of the trace files in kilobytes.

Table 23. Trace file maximum size details

Required	No
Default	0
Property type	Integer
Usage	If no value is specified, then the trace file has no maximum size.
Globalized	Yes
Bidi supported	No

Trace file name (TraceFilename)

This property specifies the full path of the trace file.

Table 24. Trace file name details

Required	No
Default	No default value
Unit of measure	Kilobytes
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Trace number of files (TraceNumberOfFiles)

This property specifies the number of trace files to use. When a trace file reaches its maximum size, the adapter starts using another trace file.

Table 25. Trace number of files details

Required	No
Default	1
Property type	Integer

Table 25. Trace number of files details (continued)

Usage	If no value is specified, the adapter uses a single trace file.
Globalized	Yes
Bidi supported	No

Managed (J2C) connection factory properties

Managed connection factory properties are used by the adapter at run time to create an outbound connection instance with the FTP server.

You set the managed connection factory properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the WebSphere Process Server administrative console.

The following table lists the managed connection factory properties. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Note: The external service wizard refers to these properties as managed connection factory properties and the WebSphere Process Server administrative console refers to them as (J2C) connection factory properties.

Table 26. Managed connection factory properties

Property name		Description
In the wizard	In the administrative console	
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing
“Custom parser class name property (CustomParserClassName)” on page 164	CustomParserClassName	Specifies the fully qualified class name of the custom parser that is used to parse the ls -l output.
“Default target file name property (Filename)” on page 164	Filename	Specifies the name of the file to be used during outbound operations.
“Directory property (OutputDirectory)” on page 165	OutputDirectory	Specifies the output directory in the FTP server.
“Disguise user data as “XXX” in log and trace files (HideConfidentialTrace) property” on page 165	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing strings of X’s instead of user data in log and trace files
Enable server verification	EnableServerVerification	Enables the remote server verification for SFTP protocol
“Encoding used by FTP server property (EISEncoding)” on page 166	EISEncoding	Specifies the encoding of the FTP server.
Host key file	HostKeyFile	The absolute path of the host key file that contains the host key of the trusted servers
“Host name property (HostName)” on page 167	HostName	Specifies the host name of the FTP server.

Table 26. Managed connection factory properties (continued)

“Host name property (SecondServerHostName)” on page 167	secondServerHostName	Specifies the host name of the second FTP server.
“Host name property (SocksProxyHost)” on page 167	SocksProxyHost	Specifies the name of the workstation that is used as a proxy server.
Passphrase property (passPhrase)	passPhrase	Used for enhanced security by encrypting the private key
“Password property (Password)” on page 168	Password	Specifies the password of the user with privileges to connect to the FTP server and perform FTP operations.
“Password property (SecondServerPassword)” on page 168	SecondServerPassword	Specifies the password of the Second FTP server to which the file is transferred during a server to server file transfer outbound operation.
“Password property (SocksProxyPassword)” on page 168	SocksProxyPassword	Specifies the password used to authenticate the proxy server.
“Port number property (PortNumber)” on page 169	PortNumber	Specifies the port number of the FTP server.
“Port number property (SecondServerPortNumber)” on page 169	SecondServerPortNumber	Specifies the port number of the second FTP server.
“Port number property (SocksProxyPort)” on page 169	SocksProxyPort	Specifies the port number of the proxy server.
“Private key file property (PrivateKeyFilePath)” on page 169	PrivateKeyFilePath	Private key used to authenticate to the secure shell server.
“Protocol property (Protocol)” on page 170	Protocol	Specifies either a normal or secure FTP connection.
“Protocol property (SecondServerProtocol)” on page 170	SecondServerProtocol	Specifies either a normal or secure FTP connection for the second server.
“Second Server Directory property (SecondServerDirectory)” on page 170	SecondServerDirectory	Specifies the directory path of the second FTP server to which the ServerToServerFileTransfer outbound operation is performed.
“Sequence file property (FileSequenceLog)” on page 171	FileSequenceLog	Specifies the full path of the file where the sequence number will be stored for outbound Create processing.
“Staging directory property (StagingDirectory)” on page 171	StagingDirectory	Specifies the directory that the file is first created in to.
“User name property (SecondServerUserName)” on page 172	SecondServerUserName	Specifies the user name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.
“User Name property (SocksProxyUserName)” on page 172	SocksProxyUserName	Specifies the user name used to authenticate the proxy server.
“User name property (Username)” on page 172	Username	Specifies the name of the user.

Adapter ID (AdapterID)

This property identifies a specific deployment, or instance, of the adapter.

Table 27. Adapter ID details

Required	Yes
Default	001
Property type	String
Usage	<p>This property identifies the adapter instance in log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, FTPRA, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to 001, the component ID is FTPRA001.</p> <p>If you run multiple instances of the same adapter, make sure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate log and trace information to a particular instance of an adapter.</p> <p>To illustrate how the length of the adapter ID property affects the filtering of log and trace files, suppose you set the adapter ID property of two instances of WebSphere Adapter for FTP to 001 and 002. The component IDs for those instances, FTPRA001 and FTPRA002, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. For example, suppose you set the adapter ID properties of two instances to Instance01 and Instance02. You will not be able to separately examine the log and trace information for each adapter instance because the component ID for both instances is truncated to FTPRAInstance.</p> <p>For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, can be set both at the resource adapter level and the managed connection factory level. After using the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, make sure you set them consistently, to prevent inconsistent marking of the log and trace entries.</p>
Globalized	Yes
Bidi supported	No

Custom parser class name property (CustomParserClassName)

Fully qualified class name of the custom parser that is used to parse the `ls -l` output. Only used when the `ls -l` output deviates from standard output.

Table 28. Custom parser class name property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Default target file name property (Filename)

Name of the file to be used during outbound operations.

Table 29. Default target file name property characteristics

Required	Yes
Default	Yes
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the file name directory. Specify the name of the environment variable in braces, preceded by a \$ symbol. For example: <code>\${FILENAME}</code> . See the topic on “Defining WebSphere Application Server environment variables” on page 66 in this documentation.
Globalized	No

Directory property (OutputDirectory)

Output directory in the FTP Server that the outbound operation is performed on.

Table 30. Directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the output directory. Specify the name of the environment variable in braces, preceded by a \$ symbol. For example: <code>\${OUTPUT_DIRECTORY}</code> . See the topic on “Defining WebSphere Application Server environment variables” on page 66 in this documentation.
Globalized	Yes

Disguise user data as “XXX” in log and trace files (HideConfidentialTrace) property

This property specifies whether to replace user data in log and trace files with a string of X’s to prevent unauthorized disclosure of potentially sensitive data.

Table 31. Disguise user data as “XXX” in log and trace files details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	<p>If you set this property to True, the adapter replaces user data with a string of X’s when writing to log and trace files.</p> <p>For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, can be set both at the resource adapter level and the managed connection factory level. After using the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, make sure you set them consistently, to prevent inconsistent marking of the log and trace entries.</p>

Table 31. Disguise user data as "XXX" in log and trace files details (continued)

Globalized	No
Bidi supported	No

Encoding used by FTP server property (EISEncoding)

Encoding of the FTP server. Use this value to set the encoding for the control connection to the FTP server.

- When both EISEncoding at the adapter level and EISEncoding at the MCF level are not set (both are null), nothing is set on the control connection while communicating with the FTP server.
- When EISEncoding at the adapter level is set and EISEncoding at the MCF level is not set, the value at adapter level is set on the control connection while communicating with the FTP server. This is helpful when using multiple MCF's and the same encoding is set. In this case, set the value at the adapter level so that all the connections will have the same encoding for the control connection.
- When EISEncoding at the adapter level is not set and EISEncoding at the MCF level is set, the value at MCF level is set on the control connection while communicating with the FTP server. Since the value is at the MCF level, this is applicable for only that MCF.
- When both EISEncoding at the adapter level and EISEncoding at the MCF level are set, the value at the MCF level takes precedence.

Specify any Java-supported encoding set for this attribute.

Table 32. Encoding used by FTP server property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Enable server verification property (EnableServerVerification)

This property is used to enable the remote server verification for SFTP protocol.

Table 33. Enable server verification property details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	When this property is set to: <ul style="list-style-type: none"> • True, server authentication is enabled • False, server authentication is disabled <p>The adapter checks the HostKeyFile property in the path of the file containing the host keys of the trusted servers.</p>
Globalized	Yes
Bidi supported	No

Host key file property (HostKeyFile)

This property provides the absolute path of the host key file that contains the host key of the trusted servers.

Table 34. Host key file property characteristics

Required	This property has to be specified if the EnableServerVerification property is enabled.
Default	None
Property type	String
Usage	This is used by the adapter to verify the host key of the remote server with the host keys of the trusted servers specified in this file.
Globalized	Yes
Bidi supported	No

Host name property (HostName)

Host name of the FTP Server to which the connection is established during an outbound operation.

Table 35. Host name property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Host name property (SecondServerHostName)

Host name of the second FTP Server to which the connection is established during an outbound operation

Table 36. Host name property characteristics

Required	Yes
Default	None
Property type	String
Usage	Contains the host name or IP address of the FTP server, for example, 9.20.13.159
Globalized	Yes

Host name property (SocksProxyHost)

Host name of the workstation that is used as a proxy server through which the adapter requests are routed to the FTP server.

Table 37. Host name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Passphrase property (passPhrase) (passPhrase)

This property is used for enhanced security by encrypting the private key.

Table 38. Passphrase property (passPhrase) property characteristics

Required	No
Possible values	Any string value
Default	None
Property type	String
Usage	Used for enhanced security. It protects the private key by encrypting it in a SFTP configuration.
Globalized	Yes
Bidi supported	No

Password property (Password)

Password of the user with privileges to connect to the FTP server and perform FTP operations.

Table 39. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SecondServerPassword)

Password of the Second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 40. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SocksProxyPassword)

Password used to authenticate the proxy server.

Table 41. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Port number property (PortNumber)

Port number of the FTP server through which the connection is established during an outbound operation.

Table 42. Port number property characteristics

Required	Yes
Default	21 for FTP, 990 for FTPS and 22 for SFTP.
Property type	Integer
Globalized	No

Port number property (SecondServerPortNumber)

Port number of the second FTP server through which the connection is established during an outbound operation.

Table 43. Port number property characteristics

Required	Yes
Default	21 for FTP, 990 for FTPS.
Property type	Integer
Globalized	No

Port number property (SocksProxyPort)

Port number of the proxy server through which the adapter requests are routed to the FTP server.

Table 44. Port number property characteristics

Required	No
Default	1080
Property type	Integer
Globalized	No

Private key file property (PrivateKeyFilePath)

This property enables you to browse and select the private key, which is used to authenticate to the secure shell server.

Table 45. Private key property characteristics

Required	No
Default	None
Property type	String
Usage	Absolute path of the file which contains the private key. Used to authenticate the user to the secure shell server.
Example	c:\temp\key.ppk
Globalized	Yes
Bidi supported	No

Protocol property (Protocol)

Protocol that determines whether the connection to be established is a normal FTP connection or a secure FTP connection.

For example:

Normal connection: FTP

FTP over SSL connection: FTPS

FTP over SSH connection: SFTP

Table 46. Protocol property characteristics

Required	Yes
Default	FTP
Property type	String
Globalized	No

Protocol property (SecondServerProtocol)

Protocol that determines whether the second connection to be established is a normal FTP connection or a secure FTP connection.

For example:

Normal connection: FTP

FTP over SSL connection: FTPS

Table 47. Protocol property characteristics

Required	Yes
Default	FTP
Property type	String
Globalized	No

Second Server Directory property (SecondServerDirectory)

Directory of the second FTP server to which the ServerToServerFileTransfer outbound operation is performed. This is the remote event directory to which the file is transferred.

Table 48. Second Server Directory property characteristics

Required	No
Default	None
Property type	String
Usage	The directory located on the FTP server and used in outbound operation represents the absolute path of the FTP directory. It does not contain any host name or URL information. For example: /home/usr/output.
Globalized	Yes

Sequence file property (FileSequenceLog)

Specifies the full path of the file where the sequence number will be stored for outbound Create processing.

When the FileSequenceLog property is specified, the adapter generates a unique sequence number to insert into the file name when processing the Create operation.

The sequence of numbers will continue to increment after multiple adapter restarts.

The sequence number is inserted into the file name in the following format:

`filename.number.extension`

For example `Customer.3.txt`

When the FileSequenceLog property is not specified or contains an invalid value, no sequence number is generated.

Table 49. Sequence file property characteristics

Required	No
Default	None
Property type	String
Usage	Important: Unless they are part of a cluster, it is not recommended that two adapter instances access the same sequence file, because concurrent requests result in delay while processing batch requests.
Globalized	No

Staging directory property (StagingDirectory)

The directory that the file is first created in to during an outbound create operation. After creation, the file is moved to the directory specified in the DirectoryPath property. The staging directory is also used for Append and Overwrite operations where the specified file is copied to StagingDirectory (if present), then appended or overwritten with content and moved back to the original specified directory. If StagingDirectory is not present, the operation is run in the actual required directory. The benefit of working with a staging directory is to avoid file writing conflicts, which can occur when people are reading the file while it is still being written during append and updates.

Table 50. Staging directory property characteristics

Required	No
Default	None
Property type	String

Table 50. Staging directory property characteristics (continued)

Usage	You can use a WebSphere Application Server environment variable to represent the staging directory. Specify the name of the environment variable in braces, preceded by a \$ symbol. For example: <code>\${STAGING_DIRECTORY}</code> . See the topic on “Defining WebSphere Application Server environment variables” on page 66 in this documentation.
Globalized	Yes

User name property (SecondServerUserName)

User name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 51. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User Name property (SocksProxyUserName)

User name used to authenticate the proxy server.

Table 52. User Name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User name property (Username)

Name of the user with privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this attribute if the User name is included in the URL specified in the FtpUrl property.

Table 53. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Wrapper and interaction specification properties

Wrapper properties are attributes of the wrapper business object that enable an application programmer to control an operation for the business objects in a wrapper. Interaction specification properties control the interaction for an operation for the entire adapter.

The external service wizard sets the interaction specification properties when you configure the adapter. You can change some, but not all, of these properties. However, you can change some properties for outbound operations. Use the assembly editor to change these properties, which reside in the method binding of the import. You set the wrapper properties using the WebSphere integration developer test client or programmatically at runtime.

The following table lists the wrapper and interaction specification properties. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Table 54. Interaction specification properties

Property name		Description
In the wizard	In the administrative console	
Remote archive directory for retrieve operation	ArchiveDirectoryForRetrieve	The adapter optionally archives the file to this folder before it is deleted during a Retrieve operation.
Create new file if the file does not exist	CreateFileIfNotExists	If the file does not exist on the FTP server, the adapter creates the file when this property is set to true during Append and Overwrite operations.
FTP server connection mode	DataConnectionMode	Data connection mode used by the FTP server during file transfers.
Delete the file after retrieve operation	DeleteOnRetrieve	The adapter deletes the file from the FTP server after it is retrieved when this property is set to true.
Remote directory on FTP system	DirectoryPath	Absolute path of the directory on the FTP server where the outbound operation needs to be performed.
File content encoding	FileContentEncoding	Encoding used while writing to the file.
File in local directory	FileInLocalDirectory	If set to true during a create operation, the file content is picked from the local directory path of the adapter workstation.
Default target file name	Filename	Name of the file in the directory provided by the DirectoryPath property.
File transfer type	FileTransferType	File transfer type used during outbound operations.
Generate a unique file	GenerateUniqueFile	The adapter creates a unique file name when this property is set to true.
Host name property	SecondServerHostName	Hostname of the second FTP server.
Delimiter between business objects in the file property	IncludeEndBODelimiter	File content is appended with this value.
Local archive directory for create operation	LocalArchiveDirForCreate	When LocalArchivingEnabledForCreate is set to true during a create operation, the file is saved to the local workstation in this directory.
Archive file in the local directory for create operation	LocalArchivingEnabledForCreate	When set to true, the file is saved to the local workstation during a create operation.
Local directory	LocalDirectoryPath	The file is picked from this directory.
Port number	SecondServerPortNumber	Port number of the second FTP server.

Table 54. Interaction specification properties (continued)

Protocol	SecondServerProtocol	Determines whether the connection is normal FTP or secure FTP.
Script File Parameters	ScriptFileParameters	The parameters required by the FTP script file.
Directory	SecondServerDirectory	Directory path of the second FTP server during a ServerToServerFileTransfer operation.
Password	SecondServerPassword	Password of the second FTP server during a ServerToServerFileTransfer operation.
User name	SecondServerUsername	Username of the second FTP server during a ServerToServerFileTransfer operation.
Specify criteria to split file content	SplitCriteria	The delimiter that separates the business objects in the event file.
Split function class name	SplittingFunctionClassName	The fully qualified class name of the class file to be used to enable file splitting.
Staging directory	StagingDirectory	The file is first created into this directory.

Remote archive directory for retrieve operation property (ArchiveDirectoryForRetrieve)

During an outbound Retrieve operation, the adapter optionally archives the file to this folder before it is deleted. The archive directory must exist.

Table 55. Remote archive directory for retrieve operation property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Create new file if the file does not exist property (CreateFileIfNotExists)

During outbound Append and Overwrite operations, if the file does not exist on the FTP server, the adapter creates the file when this property is set to true. If this property is false and file does not exist, the adapter sends an error.

Table 56. Create new file if the file does not exist property characteristics

Required	No
Default	false
Property type	Boolean
Globalized	No

Directory property (SecondServerDirectory)

Directory of the second FTP server to which the server to server file transfer outbound operation is performed. This is the remote event directory to which the file is transferred.

Table 57. Directory property characteristics

Required	No
Default	None
Property type	String
Usage	For interaction specification properties, the directory located on the FTP server and used in outbound operation represents the absolute path of the FTP directory. For example: /home/usr/output. It does not contain any host name or URL information. For wrapper business object properties, the URL of the second server to which the ServerToServerFileTransfer outbound operation is performed. For example: The syntax for specifying the FTP URL is: ftp://[UserId:password@]FTPserver[:port]/DirectoryForSecondServer.
Globalized	Yes

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Takes either active or passive. This value is used only when a file transfer is taking place. This property is not used when performing a server to server file transfer outbound operation.

Table 58. FTP server connection mode property characteristics

Required	No
Default	active
Property type	String
Possible values	active or passive
Globalized	No

Delete the file after retrieve operation (DeleteOnRetrieve)

During an outbound Retrieve operation the adapter deletes the file from the FTP server after it is retrieved when this property is set to true.

Table 59. Delete the file after retrieve operation property characteristics

Required	No
Default	false
Property type	Boolean
Globalized	No

Remote directory on FTP system property (DirectoryPath)

Absolute path of the directory on the FTP server where the outbound operation needs to be performed for all operations except ExecuteFTPScript, or the directory path on the local adapter workstation for the ExecuteFTPScript operation only. The directory must already exist.

Table 60. Remote directory on FTP system property characteristics

Required	Yes
Default	None

Table 60. Remote directory on FTP system property characteristics (continued)

Property type	String
Usage	The DirectoryPath directory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter will not create this directory automatically.
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used while writing to the file. If this property is not specified, the adapter tries to read without using any specific encoding. You can specify any Java supported encoding set.

Table 61. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File in local directory property (FileInLocalDirectory)

During outbound create operations, if this property is set to true, the file content is not available in the business object. The file is retrieved from the local directory on the adapter workstation. During outbound retrieve operations, if this property is set to true, the file content is not sent to the J2EE application as part of the business object. The file is saved to the local directory of the adapter workstation.

Table 62. File in local directory property characteristics

Required	No
Default	false
Property type	Boolean
Globalized	No

Default target file name property (Filename)

Name of the file to be used during outbound operations.

Table 63. Default target file name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

File transfer type property (FileTransferType)

File transfer type used during outbound operations. Takes either ASCII or binary.

Table 64. File transfer type property characteristics

Required	No
Default	binary
Property type	String
Globalized	No

Generate a unique file (GenerateUniqueFile)

During outbound Create operation, the adapter creates a unique file name when this property is true. The adapter ignores any value that is set for the Filename property when this property is set to true.

Note: The adapter does not support both GenerateUniqueFile and StagingDirectory options at the same time.

Table 65. Generate a unique file property characteristics

Required	No
Default	false
Property type	Boolean
Globalized	No
Restrictions	The FTP server must support RFC1123 to use this feature.

Host name property (SecondServerHostName)

Hostname of the second FTP server to which the connection is established during an outbound operation.

Table 66. Host name property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Delimiter between business objects in the file property (IncludeEndBODelimiter)

File content is appended with this value. Used during the outbound create, append, and overwrite operations.

Table 67. Include business object delimiter in the file content property characteristics

Required	No
----------	----

Table 67. Include business object delimiter in the file content property characteristics (continued)

Default	<p>For the create and overwrite operations, no default value is set.</p> <p>For the append operation, the default value is <EndB0>.</p> <p>For the append operation, the following rules apply:</p> <ul style="list-style-type: none"> • If the delimiter is set to null in the business object wrapper, no delimiter is used to separate the business objects. • If the IncludeEndBODelimiter property is not set in the business object wrapper, and the value in the interaction specification is also null, the default is <EndB0>. • If a specific delimiter value is specified in the business object wrapper, the specified value will be appended. • If both the business object wrapper and the interaction specification have specified values, the business object wrapper value takes precedence.
Property type	String
Globalized	Yes

Local archive directory for create operation property (LocalArchiveDirForCreate)

During outbound create operations, when the file content is coming as part of the business object and LocalArchivingEnabledForCreate is set to true, the file is saved to the local workstation in this directory.

Table 68. Local archive directory for create property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalArchiveDirForCreate directory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter will not create this directory automatically.
Globalized	Yes

Archive file in the local directory for create operation property (LocalArchivingEnabledForCreate)

During outbound create operations, when the file content is coming as part of the business object from a J2EE application and this property is set to true, the file is saved to the local workstation in the LocalArchiveDirForCreate directory before performing the outbound operation.

Table 69. Archive file in the local directory for create operation property characteristics

Required	No
Default	false
Property type	Boolean
Globalized	No

Local directory property (LocalDirectoryPath)

During outbound create operations, when FileInLocalDirectory property is set to true, the file content is not available in the business object. Instead the file is

picked from this directory. During outbound retrieve operations, when FileInLocalDirectory property is set to true, the file content is not sent as part of business object. The file is saved to this directory.

Table 70. Local directory property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalDirectoryPath directory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter will not create this directory automatically.
Globalized	Yes

Port number property (SecondServerPortNumber)

Port number of the second FTP server through which the connection is established during an outbound operation.

Table 71. Port number property characteristics

Required	Yes
Default	21 for FTP, 990 for FTPS
Property type	Integer
Globalized	No

Protocol property (SecondServerProtocol)

Protocol that determines whether the second connection to be established is a normal FTP connection or a secure FTP connection.

Table 72. Protocol property characteristics

Required	Yes
Default	FTP
Property type	String
Globalized	No
Example	For example: Normal connection: FTP FTP over SSL connection: FTPS

Script File Parameters property (ScriptFileParameters)

During an outbound ExecuteFTPScript operation, the parameters required by the FTP script file are set in this property. During runtime the adapter replaces the parameters with these values.

Table 73. Script File Parameters property characteristics

Required	No
Default	None
Property type	String

Table 73. Script File Parameters property characteristics (continued)

Globalized	Yes
------------	-----

Password property (SecondServerPassword)

Password of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 74. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User name property (SecondServerUsername)

Username of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 75. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Specify criteria to split file content property (SplitCriteria)

Takes different values based on the value of the SplittingFunctionClassName property.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
 - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
 - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=0, chunking is disabled.

Table 76. Specify criteria to split file content property characteristics

Required	No
Default	0
Property type	String
Globalized	Yes

Split function class name property (SplittingFunctionClassName)

Takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The `com.ibm.j2ca.extension.utils.filesplit.SplitByDelimiter` class that splits the event file based on delimiter.
- The `com.ibm.j2ca.extension.utils.filesplit.SplitBySize` class that splits the event file based on the event file size.

The delimiter or file size is provided in the `SplitCriteria` property.

Table 77. Split function class name property characteristics

Required	No
Default	<code>com.ibm.j2ca.extension.utils.filesplit.SplitBySize</code>
Property type	String
Globalized	No

Staging directory property (StagingDirectory)

During outbound create operations, the file will first be created into this directory. When the file creation is complete, the file is copied to the directory specified in the `DirectoryPath` property. This staging directory is also used for Append and Overwrite operations where the specified file is copied to the `StagingDirectory`, if present. The appended or overwritten content is then moved back to the original specified directory. If `StagingDirectory` is not specified, the operation is run in the actual required directory.

Note: The adapter does not support both `StagingDirectory` and `GenerateUniqueFile` options at the same time.

Table 78. Staging directory property characteristics

Required	No
Default	None
Property type	String
Usage	The <code>StagingDirectory</code> directory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter will not create this directory automatically.
Globalized	Yes

Inbound configuration properties

WebSphere Adapter for FTP has several categories of inbound connection configuration properties, which you set with the external service wizard while generating or creating objects and services. You can change the resource adapter and activation specification properties after you deploy the module using WebSphere Integration Developer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

Guide to information about properties

The properties used to configure WebSphere Adapter for FTP are described in detail in tables included in each of the configuration properties topics, such as

Resource adapter properties, Managed connection factory properties, and so on. To help you use these tables, information about each row you might see is explained here.

The following table explains the meaning of each row that might be displayed in the table for a configuration property.

Row	Explanation
Required	<p>A required field (property) must have a value in order for the adapter to work. Sometimes the external service wizard provides a default value for required properties.</p> <p>Removing a default value from a required field on the external service wizard <i>will not change that default value</i>. When a required field contains no value at all, the external service wizard will process the field using its assigned default value, and that default value will also be displayed on the administrative console.</p> <p>Possible values are Yes and No.</p> <p>Sometimes a property is required only when another property has a specific value. When this is the case, the table will note this dependency. For example,</p> <ul style="list-style-type: none"> • Yes, when the EventQueryType property is set to Dynamic • Yes, for Oracle databases
Possible values	Lists and describes the possible values that you can select for the property.
Default	<p>The predefined value that is set by the external service wizard. When the property is required, you must either accept the default value or specify one yourself. If a property has no default value, the table will state No default value.</p> <p>The word None is an acceptable default value, and does not mean that there is no default value.</p>
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	<p>Describes the property type. Valid property types include the following:</p> <ul style="list-style-type: none"> • Boolean • String • Integer
Usage	<p>Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented:</p> <p>For Rational Application Developer for WebSphere Software version 6.40 or earlier, the password:</p> <ul style="list-style-type: none"> • Must be uppercase • Must be 8 characters in length <p>For versions of Rational Application Developer for WebSphere Software later than 6.40, the password:</p> <ul style="list-style-type: none"> • Is not case sensitive • Can be up to 40 characters in length. <p>This section lists other properties that affect this property or the properties that are affected by this property and describes the nature of the conditional relationship.</p>
Example	<p>Provides sample property values, for example:</p> <p>"If Language is set to JA (Japanese), Codepage number is set to 8000".</p>

Row	Explanation
Globalized	If a property is globalized, it has national language support, meaning that you can set the value in your national language. Valid values are Yes and No .
Bidi supported	Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing pertains to the task of processing data that contains both left-to-right (Hebrew or Arabic, for example) and right-to-left (a URL or file path, for example) semantic content within the same file. Valid values are Yes and No .

Adapter type properties

Adapter type properties provide the external service wizard with the adapter details. These properties are configured using the external service wizard before deployment or with the WebSphere Application Server administrative console after deployment.

Note: If you set any of these adapter type properties using bidirectional script, you must set values that identify the format of the bidirectional script entered for that property.

The adapter type properties and their purpose are described in the following table. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see “Guide to information about properties” on page 155.

Table 79. Adapter type properties

Property name		Description
In the wizard	In the administrative console	
“Description property (Description)”	Description	Adapter description.
“Display Name property (DisplayName)” on page 184	DisplayName	Adapter display name.
“ID property (ID)” on page 184	ID	ID for the adapter type.
“Vendor property (Vendor)” on page 184	Vendor	Name of the vendor providing the adapter.
“Version property (Version)” on page 184	Version	Adapter version.

Description property (Description)

Adapter description.

Table 80. Description property characteristics

Required	Yes
----------	-----

Table 80. Description property characteristics (continued)

Default	IBM WebSphere Adapter for FTP
Property type	String

Display Name property (DisplayName)

Adapter display name.

Table 81. DisplayName property characteristics

Required	Yes
Default	IBM WebSphere Adapter for FTP
Property type	String

ID property (ID)

ID for the adapter type.

Table 82. ID property characteristics

Required	Yes
Default	FTP
Property type	String

Vendor property (Vendor)

Name of the vendor providing the adapter.

Table 83. Vendor property characteristics

Required	Yes
Default	IBM
Property type	String

Version property (Version)

Adapter version.

Table 84. Version property characteristics

Required	Yes
Default	6.2
Property type	String

Resource adapter properties

The resource adapter properties control the general operation of the adapter, such as specifying the namespace for business objects. You set the resource adapter properties using the external service wizard when you configure the adapter. After deploying the adapter, use the administrative console to change these properties.

The following properties for logging and tracing are no longer required in version 6.1.x, but are supported for compatibility with previous versions:

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

The following table lists the resource adapter properties and their purpose. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Table 85. Resource adapter properties for the Adapter for FTP

Property name		Description
In the wizard	In the administrative console	
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing
“EISEncoding (EISEncoding)” on page 186	EISEncoding	Encoding of the FTP server.
(Not available)	enableHASupport	Do not change this property.
“Disguise user data as “XXX” in log and trace files (HideConfidentialTrace) property” on page 186	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing strings of X’s instead of user data in log and trace files
(Not available)	LogFileMaxSize	Deprecated
(Not available)	LogFilename	Deprecated
(Not available)	LogNumberOfFiles	Deprecated
(Not available)	TraceFileMaxSize	Deprecated
(Not available)	TraceFileName	Deprecated
(Not available)	TraceNumberOfFiles	Deprecated

Adapter ID (AdapterID)

This property identifies a specific deployment, or instance, of the adapter.

Table 86. Adapter ID details

Required	Yes
Default	001
Property type	String

Table 86. Adapter ID details (continued)

Usage	<p>This property identifies the adapter instance in log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, FTPRA, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to 001, the component ID is FTPRA001.</p> <p>If you run multiple instances of the same adapter, make sure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate log and trace information to a particular instance of an adapter.</p> <p>To illustrate how the length of the adapter ID property affects the filtering of log and trace files, suppose you set the adapter ID property of two instances of WebSphere Adapter for FTP to 001 and 002. The component IDs for those instances, FTPRA001 and FTPRA002, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. For example, suppose you set the adapter ID properties of two instances to Instance01 and Instance02. You will not be able to separately examine the log and trace information for each adapter instance because the component ID for both instances is truncated to FTPRAInstance.</p> <p>For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, can be set both at the resource adapter level and the managed connection factory level. After using the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, make sure you set them consistently, to prevent inconsistent marking of the log and trace entries.</p>
Globalized	Yes
Bidi supported	No

EISEncoding (EISEncoding)

This property specifies the encoding of the FTP server. Sets the encoding for the control connection while communicating with the FTP server. Set the property if the FTP server's directories or file names contain globalized characters.

Table 87. EISEncoding characteristics

Required	No
Default	None
Property type	String
Examples	UTF-8, ISO-8859-1

Enable high availability support (enableHASupport)

Do not change this property. It must be set to true.

Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) property

This property specifies whether to replace user data in log and trace files with a string of X's to prevent unauthorized disclosure of potentially sensitive data.

Table 88. Disguise user data as "XXX" in log and trace files details

Required	No
----------	----

Table 88. Disguise user data as "XXX" in log and trace files details (continued)

Possible values	True False
Default	False
Property type	Boolean
Usage	If you set this property to True, the adapter replaces user data with a string of X's when writing to log and trace files. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, can be set both at the resource adapter level and the managed connection factory level. After using the external service wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently. If you use the WebSphere Integration Developer assembly editor or the administrative console to reset these properties, make sure you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	No
Bidi supported	No

Log file maximum size (LogFileMaxSize)

This property specifies the size of the log files in kilobytes.

Table 89. Log file maximum size details

Required	No
Default	0
Property type	Integer
Usage	When the log file reaches it maximum size, the adapter start using a new log file. If the file size is specified as 0 or no maximum size is specified, the file does not have a maximum size.
Globalized	Yes
Bidi supported	No

Log file name (LogFilename)

This property specifies the full path name of the log file.

Table 90. Log file name details

Required	No
Default	No default value
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Log number of files (LogNumberOfFiles)

This property specifies the number of log files.

Table 91. Log number of files details

Required	No
----------	----

Table 91. Log number of files details (continued)

Default	1
Property type	Integer
Usage	When a log file reaches its maximum size, the adapter starts using another log file. If no value is specified, the adapter creates a single log file.
Globalized	Yes
Bidi supported	No

Trace file maximum size (TraceFileMaxSize)

This property specifies the size of the trace files in kilobytes.

Table 92. Trace file maximum size details

Required	No
Default	0
Property type	Integer
Usage	If no value is specified, then the trace file has no maximum size.
Globalized	Yes
Bidi supported	No

Trace file name (TraceFilename)

This property specifies the full path of the trace file.

Table 93. Trace file name details

Required	No
Default	No default value
Unit of measure	Kilobytes
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Trace number of files (TraceNumberOfFiles)

This property specifies the number of trace files to use. When a trace file reaches its maximum size, the adapter starts using another trace file.

Table 94. Trace number of files details

Required	No
Default	1
Property type	Integer
Usage	If no value is specified, the adapter uses a single trace file.
Globalized	Yes
Bidi supported	No

Activation specification properties

Activation specification properties are properties that hold the inbound event processing configuration information for a message endpoint.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint (a message driven bean).

You set the activation specification properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the administrative console.

The following table lists the activation specification properties. A complete description of each property is provided in the sections that follow the table. For information about how to read the property details tables in the sections that follow, see Guide to understanding property details.

Table 95. Activation specification properties

Property name		Description
In the wizard	In the administrative console	
“Ensure assured-only event delivery (AssuredOnceDelivery)” on page 193	AssuredOnceDelivery	Specifies whether the adapter provides assured once delivery of events
“Auto create event table property (EP_CreateTable)” on page 193	EP_CreateTable	Tells the adapter whether or not to create the Event Persistence table
“Create Table property (CreateTable)” on page 193	CreateTable	When set to true, the event table and related indexes are created
“Custom parser class name property (CustomParserClassName)” on page 194	CustomParserClassName	Fully qualified class name of the custom parser which is used to parse the ls -l output
“Database Password property (DatabasePassword)” on page 194	DatabasePassword	Password used by event persistence for retrieving the JDBC database connection from the data source
“Database schema name property (EP_SchemaName)” on page 194	EP_SchemaName	Schema name of the database used by event persistence
“Database Username property (DatabaseUsername)” on page 194	DatabaseUsername	User name used by event persistence for retrieving the JDBC database connection from the data source
“FTP server connection mode property (DataConnectionMode)” on page 195	DataConnectionMode	Data connection mode used by the FTP server during file transfers
(Not available)	DefaultObjectName	Supported for compatibility with earlier versions
“Type of delivery (DeliveryType)” on page 195	DeliveryType	Determines the order in which events are delivered by the adapter to the export

Table 95. Activation specification properties (continued)

“Encoding used by FTP server property (EISEncoding)” on page 195	EISEncoding	Encoding of the FTP server
(Not available)	EventContentType	Supported for compatibility with earlier versions
“Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)” on page 196	EP_DataSource_JNDIName	JNDI name of the data source used by event persistence to get the JDBC database connection
“Event recovery table name property (EP_EventTableName)” on page 196	EP_TableName	Name of the table that is used by the adapter for event persistence
“Event types to process (EventTypeFilter)” on page 196	EventTypeFilter	A delimited list of event types that indicates to the adapter which events it should deliver
“Failure file extension for local archive property (FailedArchiveExt)” on page 197	FailedArchiveExt	File extension used to archive business objects in the event file that are not successfully processed
“File content encoding property (FileContentEncoding)” on page 197	FileContentEncoding	Encoding used to read the event files
“File extension for remote archive property (ftpRenameExt)” on page 197	ftpRenameExt	File extension or suffix that the adapter uses to rename the remote FTP file
“Pass only file name and directory, not the content property (FilePassByReference)” on page 197	FilePassByReference	Specifies that the file content of the event file is not sent to the export
“File transfer type property (FileTransferType)” on page 198	FileTransferType	File transfer type used during inbound processing
“Do not process events that have a timestamp in the future (FilterFutureEvents)” on page 198	FilterFutureEvents	Specifies whether the adapter filters out future events by comparing the timestamp on each event with the system time
“Number of files to get at a time property (ftpGetQuantity)” on page 198	ftpGetQuantity	Determines the number of files retrieved from the remote FTP URL
“Number of poll periods between downloads property (ftpPollFrequency)” on page 199	ftpPollFrequency	Determines how frequently the adapter polls the FTP server
Retry limit for failed events (FailedEventRetryLimit)	FailedEventRetryLimit	The number of times the adapter attempts to redeliver an event before marking the event as failed
“Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)” on page 199	ftpScriptFileExecutedAfterInbound	Specifies the path of the script file that is run after every inbound poll cycle

Table 95. Activation specification properties (continued)

“Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)” on page 200	ftpScriptFileExecutedBeforeInbound	Specifies the path of the script file that is run prior to every inbound poll cycle
“Host name property (HostName)” on page 200	HostName	Hostname of the FTP Server to which the connection is established
“Include business object delimiter in the file content property (IncludeEndBODElimiter)” on page 200	IncludeEndBODElimiter	When set to true, the delimiter is sent with the business object content for further processing
“Local archive directory property (LocalArchiveDirectory)” on page 200	LocalArchiveDirectory	Absolute path of the local Archive directory
“Local directory property (LocalEventDirectory)” on page 201	LocalEventDirectory	Local system directory into which the adapter downloads event files from the FTP site
“Maximum connections (MaximumConnections)” on page 201	MaximumConnections	The maximum number of connections that the adapter can use for inbound event delivery
“Minimum connections (MinimumConnections)” on page 202	MinimumConnections	The minimum number of connections that the adapter can use for inbound event delivery
“File extension for local archive property (originalArchiveExt)” on page 202	OriginalArchiveExt	File extension used to archive the original event file
Passphrase property (passPhrase)	passPhrase	Used for enhanced security by encrypting the private key
“Password property (Password)” on page 202	Password	Password of the user who has privileges to connect to the FTP server and perform FTP operations
“Password used to connect to event data source property (EP_Password)” on page 202	EP_Password	Password used during event persistence
“Interval between polling periods (PollPeriod)” on page 203	PollPeriod	The length of time that the adapter waits between polling periods
“Maximum events in polling period (PollQuantity)” on page 203	PollQuantity	The number of events that the adapter delivers to the export during each poll period
“Port number property (PortNumber)” on page 204	PortNumber	Port number of the FTP server
“Private key file property (PrivateKeyFilePath)” on page 204	PrivateKeyFilePath	Private key used to authenticate to the secure shell server
“Protocol property (Protocol)” on page 204	Protocol	Determines whether the connection is normal FTP or secure FTP
“Retrieve files with this pattern property (EventFileMask)” on page 206	EventFileMask	Filter for the event files

Table 95. Activation specification properties (continued)

Retry EIS connection on startup	RetryConnectionOnStartup	Controls whether the adapter retries the connection to the FTP server if it cannot connect at startup
“Retry interval if connection fails (RetryInterval)” on page 206	RetryInterval	The length of time that the adapter waits between attempts to establish a new connection after an error during inbound operations
“Number of times to retry the system connection (RetryLimit)” on page 207	RetryLimit	The number of times the adapter tries to reestablish an inbound connection after an error
“Remote archive directory property (ftpArchiveDirectory)” on page 204	ftpArchiveDirectory	Relative path of the archive directory on the FTP server
“Remote directory property (EventDirectory)” on page 205	EventDirectory	Remote directory of the FTP server from where the event files are retrieved for inbound processing
Enable server verification	EnableServerVerification	Enables the remote server verification for SFTP protocol
Host key file	HostKeyFile	The absolute path of the host key file that contains the host key of the trusted servers
“Host name property (SocksProxyHost)” on page 208	SocksProxyHost	Host name of the machine used as a proxy server
“Password property (SocksProxyPassword)” on page 208	SocksProxyPassword	Password used to authenticate the proxy server
“Port number property (SocksProxyPort)” on page 208	SocksProxyPort	Port number of the proxy server
“User name property (SocksProxyUserName)” on page 208	SocksProxyUserName	User name used to authenticate the proxy server
“Sort event files property (SortEventFiles)” on page 209	SortEventFiles	Determines the sorting order of event files being polled
“Specify criteria to split file content property (SplitCriteria)” on page 209	SplitCriteria	Takes different values based on the value of the SplittingFunctionClassName property
“Splitting function class name property” on page 210	SplittingFunctionClassName	Takes the fully qualified class name of the class file to be used to enable file splitting
“Stop the adapter when an error is encountered while polling (StopPollingOnError)” on page 211	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling
“Success file extension for local archive property (SuccessArchiveExt)” on page 211	SuccessArchiveExt	File extension used to archive all of the successfully processed business objects
“User name property (UserName)” on page 211	UserName	Name of the user who has privileges to connect to the FTP server and perform FTP operations

Table 95. Activation specification properties (continued)

“User name used to connect to event data source property (EP_UserName)” on page 212	EP_UserName	User name used by event persistence for getting the database connection
Rule Editor	ruleTable	The collection of rules used to filter the events

Ensure assured-only event delivery (AssuredOnceDelivery)

This property specifies whether to provide ensure once-only event delivery for inbound events.

Table 96. Ensure assured-only event delivery details

Required	Yes
Possible values	True False
Default	True
Property type	Boolean
Usage	<p>When this property is set to True, the adapter provides assured once event delivery. This means that each event will be delivered once and only once. A value of False does not provide assured once event delivery, but provides better performance.</p> <p>When this property is set to True, the adapter attempts to store transaction (XID) information in the event store. If it is set to False, the adapter does not attempt to store the information.</p> <p>This property is used only if the export component is transactional. If it is not, no transaction can be used, regardless of the value of this property.</p>
Globalized	No
Bidi supported	No

Auto create event table property (EP_CreateTable)

Tells the adapter whether or not to create the Event Persistence table. If the value is true and table does not exist then the adapter creates the table. If the value is false the adapter does not create the table.

Table 97. Auto create event table property characteristics

Required	No
Default	true
Property type	Boolean
Globalized	No

Create Table property (CreateTable)

When set to true, the event table and related indexes are created. For troubleshooting table creation errors, set this property to false. The table and indexes can then be created manually.

Table 98. Create Table property characteristics

Required	No
Default	true

Table 98. Create Table property characteristics (continued)

Property type	Boolean
Globalized	No

Custom parser class name property (CustomParserClassName)

Fully qualified class name of the custom parser which is used to parse the `ls -l` output. Used only when the `ls -l` output deviates from standard output.

Table 99. Custom parser class name property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Database Password property (DatabasePassword)

Password used by event persistence for retrieving the JDBC database connection from the data source.

Table 100. Database Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database schema name property (EP_SchemaName)

Schema name of the database used by event persistence.

Table 101. Database schema name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database Username property (DatabaseUsername)

User name used by event persistence for retrieving the JDBC database connection from the data source.

Table 102. Database Username property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Accepts either active or passive settings.

Table 103. FTP server connection mode property characteristics

Required	No
Default	active
Property type	String
Globalized	No

Type of delivery (DeliveryType)

This property specifies the order in which events are delivered by the adapter to the export.

Table 104. Type of delivery details

Required	No
Possible values	ORDERED UNORDERED
Default	ORDERED
Property type	String
Usage	The following values are supported: <ul style="list-style-type: none">• ORDERED: The adapter delivers events to the export one at a time.• UNORDERED: The adapter delivers all events to the export at once.
Globalized	No
Bidi supported	No

Encoding used by FTP server property (EISEncoding)

Encoding of the FTP server. Use this value to set the encoding for the control connection to the FTP server.

- When both EISEncoding at the adapter level and EISEncoding at the activation specification level are not set (both are null), nothing is set on the control connection while communicating with the FTP server.
- When EISEncoding at the adapter level is set and EISEncoding at the activation specification level is not set, the value at adapter level is set on the control connection while communicating with the FTP server. This is helpful when using multiple activation specification's and the same encoding is set. In this case, set the value at the adapter level so that all the connections will have the same encoding for the control connection.
- When EISEncoding at the adapter level is not set and EISEncoding at the activation specification level is set, the value at activation specification level is set on the control connection while communicating with the FTP server. Since the value is at the activation specification level, this is applicable for only that activation specification.
- When both EISEncoding at the adapter level and EISEncoding at the activation specification level are set, the value at the activation specification level takes precedence.

Specify any Java-supported encoding set for this attribute.

Table 105. Encoding used by FTP server property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)

JNDI name of the data source used by event persistence to get the JDBC database connection. The data source must be created in WebSphere Process Server. The database name specified while creating the data source must already exist.

Table 106. Event recovery data source (JNDI) name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery table name property (EP_EventTableName)

Name of the table that is used by the adapter for event persistence. When using multiple activation specifications, this value must be unique for each. The same table name must not be used by other instances of same adapter or a different adapter. If the table does not exist in the database, the adapter will create the table.

Table 107. Event recovery table name property characteristics

Required	No
Default	FTPTABLE
Property type	String
Globalized	Yes

Event types to process (EventTypeFilter)

This property contains a delimited list of event types that indicates to the adapter which events it should deliver.

Table 108. Event types to process details

Required	No
Possible values	A comma-delimited (,) list of business object types
Default	null
Property type	String
Usage	Events are filtered by business object type. If the property is set, the adapter delivers only those events that are in the list. A value of null indicates that no filter will be applied and that all events will be delivered to the export.
Example	To receive only events relating to the Customer and Order business objects, specify this value: Customer,Order
Globalized	No

Table 108. Event types to process details (continued)

Bidi supported	No
----------------	----

Failure file extension for local archive property (FailedArchiveExt)

File extension used to archive business objects in the event file that are not successfully processed. This property is used only when LocalArchiveDirectory is valid and exists.

Table 109. Failure file extension for local archive property characteristics

Required	No
Default	fail
Property type	String
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used to read the event files based on the EndBODelimiter property and during string to byte[] conversions. If not specified, the adapter attempts to read without any specific encoding. You can specify any Java supported encoding set.

Table 110. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File extension for remote archive property (ftpRenameExt)

File extension or suffix that the adapter uses to rename the remote FTP file after the connector has polled for it. Renaming the file prevents the connector from polling the same file in the next poll cycle. The adapter can be configured to rename the processed event file and move it to an archive directory.

Table 111. File extension for remote archive property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Pass only file name and directory, not the content property (FilePassByReference)

Specifies that the file content of the event file is not sent to the export.

If set to true, the file is appended with a timestamp and sent to the LocalArchiveDirectory. The timestamp prevents errors and overwrites to the file when another file with the same name is received. This property can be set to true

only when the LocalArchiveDirectory property is set and the specified directory exists. This property is used only for PassThrough inbound processing. When enabled, the file is not split into chunks.

Table 112. Pass only file name and directory, not the content property characteristics

Required	No
Default	false
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during inbound processing. Accepts either ASCII or binary.

Table 113. File transfer type property characteristics

Required	No
Default	binary
Property type	String
Globalized	no

Do not process events that have a timestamp in the future (FilterFutureEvents)

This property specifies whether the adapter filters out future events by comparing the timestamp on each event with the system time.

Table 114. Do not process events that have a timestamp in the future details

Required	Yes
Possible values	True False
Default	False
Property type	Boolean
Usage	If set to True, the adapter compares the time of each event to the system time. If the event time is later than the system time, the event is not delivered. If set to False, the adapter delivers all events.
Globalized	No
Bidi supported	No

Number of files to get at a time property (ftpGetQuantity)

Determines the number of files retrieved from the remote FTP URL with each remote poll.

Table 115. Number of files to get at a time property characteristics

Required	Yes
Default	10
Property type	Integer
Globalized	No

Number of poll periods between downloads property (ftpPollFrequency)

Determines how frequently the adapter polls the FTP server, measured in the number of standard poll cycles. For example, if PollPeriod is set to 10000, and FTTPollFrequency is set to 6, the adapter polls the LocalEventDirectory every 10 seconds and polls the remote EventDirectory every 60 seconds. The adapter performs FTP polling only if you specify a value for this property. If PollPeriod is 0, you consider it as 1 for calculation. If the calculation evaluates to 0, the adapter does not perform FTP polling.

Table 116. Number of poll periods between downloads property characteristics

Required	Yes
Default	5
Property type	Integer
Globalized	No

Retry limit for failed events (FailedEventRetryLimit)

This property specifies the number of times that the adapter attempts to redeliver an event before marking the event as failed.

Table 117. Retry limit for failed events details

Required	No
Possible values	Integers
Default	5
Property type	Integer
Usage	<p>Use this property to control how many times the adapter tries to send an event before marking it as failed. It accepts the following values:</p> <p>Default If this property is not set, the adapter tries five additional times before marking the event as failed.</p> <p>0 The adapter tries to deliver the event an infinite number of times. When the property is set to 0, the event remains in the event store and the event is never marked as failed.</p> <p>> 0 For integers greater than zero, the adapter retries the specified number of times before marking the event as failed.</p> <p>< 0 For negative integers, the adapter does not retry failed events.</p>
Globalized	No
Bidi supported	No

Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)

Specifies the path of the script file that is run after every inbound poll cycle. This feature can be used to perform additional actions on the FTP server after each poll cycle. For example, it can be used to set file permissions.

Table 118. Run FTP script file after downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)

Specifies the path of the script file that is run prior to every inbound poll cycle. This feature can be used to perform additional actions on the FTP server prior to each poll cycle.ftpScriptFileExecutedAfterInbound

Table 119. Run FTP script file before downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Host name property (HostName)

Hostname of the FTP Server to which the connection is established during inbound processing.

Table 120. Create Table property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Include business object delimiter in the file content property (IncludeEndBODelimiter)

When set to true, the delimiter is sent with the business object content for further processing. This property is valid only when splitting the event files based on a delimiter.

Table 121. Include business object delimiter in the file content property characteristics

Required	No
Default	false
Property type	String
Globalized	No

Local archive directory property (LocalArchiveDirectory)

Absolute path of the local Archive directory. The directory must be valid and already exist.

Table 122. Local archive directory property characteristics

Required	No
Default	None
Property type	String
Usage	<p>You can use a WebSphere Application Server environment variable to represent the local archive directory. Specify the name of the environment variable in braces, preceded by a \$ symbol. For example: <code>\${LOCALARCHIVE_DIRECTORY}</code>. See the topic on “Defining WebSphere Application Server environment variables” on page 66 in this documentation.</p> <p>Note: The LocalArchiveDirectory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter will not create this directory automatically.</p>
Globalized	Yes

Local directory property (LocalEventDirectory)

Local system directory into which the adapter downloads event files from the FTP site. You must specify a value for this property to enable the adapter to process events.

Table 123. Local directory property characteristics

Required	Yes
Default	None
Property type	String
Usage	<p>You can use a WebSphere Application Server environment variable to represent the local event directory. Specify the name of the environment variable in braces, preceded by a \$ symbol. For example: <code>\${LOCAL_DIRECTORY}</code>. See the topic on “Defining WebSphere Application Server environment variables” on page 66 in this documentation.</p> <p>Note: The LocalEventDirectory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter will not create this directory automatically.</p>
Globalized	Yes

Maximum connections (MaximumConnections)

This property specifies the maximum number of connections that the adapter can use for inbound event delivery.

Table 124. Maximum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. The adapter considers any positive entry less than 1 to be equal to 1. Typing a negative value or 1 for this property may result in runtime errors.
Globalized	No
Bidi supported	No

Minimum connections (MinimumConnections)

This property specifies the minimum number of connections that the adapter can use for inbound event delivery.

Table 125. Minimum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. Any value less than 1 is treated as 1 by the adapter. Typing a negative value or 1 for this property may result in run time errors.
Globalized	No
Bidi supported	No

File extension for local archive property (originalArchiveExt)

File extension used to archive the original event file. This preserves the entire event file for reference in case any of its business objects fail. This property is used only when LocalArchiveDirectory is valid and exists.

Table 126. File extension for local archive property characteristics

Required	No
Default	original
Property type	String
Globalized	Yes

Password property (Password)

Password of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the password is included in the URL specified in the EventDirectory property.

Table 127. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password used to connect to event data source property (EP_Password)

The password used during event persistence to get the database connection from the data source.

Table 128. Password used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Interval between polling periods (PollPeriod)

This property specifies the length of time that the adapter waits between polling periods.

Table 129. Interval between polling periods details

Required	Yes
Possible values	Integers greater than or equal to 0.
Default	2000
Unit of measure	Milliseconds
Property type	Integer
Usage	The poll period is established at a fixed rate, which means that if running the poll cycle is delayed for any reason (for example, if a prior poll cycle takes longer than expected to complete) the next poll cycle will occur immediately to make up for the lost time caused by the delay.
Globalized	No
Bidi supported	No

Maximum events in polling period (PollQuantity)

This property specifies the number of events that the adapter delivers to the export during each poll period.

Table 130. Maximum events in polling period details

Required	Yes
Default	10
Property type	Integer
Usage	The value must be greater than 0. If this value is increased, more events are processed per polling period and the adapter may perform less efficiently. If this value is decreased, fewer events are processed per polling period and the adapter's performance may improve slightly.
Globalized	No
Bidi supported	No

Passphrase property (passPhrase) (passPhrase)

This property is used for enhanced security by encrypting the private key.

Table 131. Passphrase property (passPhrase) property characteristics

Required	No
Possible values	Any string value
Default	None
Property type	String
Usage	Used for enhanced security. It protects the private key by encrypting it in a SFTP configuration.
Globalized	Yes
Bidi supported	No

Port number property (PortNumber)

Port number of the FTP server through which the connection is established during inbound processing.

Table 132. Port number property characteristics

Required	Yes
Default	21 for FTP, 990 for FTPS and 22 for SFTP.
Property type	Integer
Globalized	No

Private key file property (PrivateKeyFilePath)

This property enables you to browse and select the private key, which is used to authenticate to the secure shell server.

Table 133. Private key property characteristics

Required	No
Default	None
Property type	String
Usage	Absolute path of the file which contains the private key. Used to authenticate the user to the secure shell server.
Example	c:\temp\key.ppk
Globalized	Yes
Bidi supported	No

Protocol property (Protocol)

Protocol that determines whether the connection to be established is a normal FTP connection or a secure FTP connection.

For example:

Normal connection: FTP

FTP over SSL connection: FTPS

FTP over SSH connection: SFTP

Table 134. Protocol property characteristics

Required	Yes
Default	FTP
Property type	String
Globalized	No

Remote archive directory property (ftpArchiveDirectory)

Relative path of the archive directory on the FTP server. The directory must already exist. There are several options for using this property to specify archiving:

- Specifying a value for this property, but no value for the FTPRenameExt property causes the adapter to append a timestamp to the event file name and move it to the FTP server archive directory specified in this property.
- Specifying a value for this property and the FTPRenameExt property causes the adapter to rename the processed event file name with a timestamp and the value specified in FTPRenameExt and moves it to the FTP server archive directory specified in this property.
- Specifying no value either for this property or the FTPRenameExt property causes the adapter to delete the processed event file without archiving it.
- Specifying no value for this property but specifying a value for the FTPRenameExt property causes the adapter to rename the processed event file, adding a timestamp and the value specified in FTPRenameExt.

Table 135. Remote archive directory property characteristics

Required	No
Default	None
Property type	String
Usage	<p>You can use a WebSphere Application Server environment variable to represent the remote archive directory. Specify the name of the environment variable in braces, preceded by a \$ symbol. For example: \${REMOTEARCHIVE_DIRECTORY}. See the topic on “Defining WebSphere Application Server environment variables” on page 66 in this documentation.</p> <p>The archive directory located on the FTP server and used in inbound configuration represents the absolute path of the archive directory. It does not contain any host name or URL information. This directory is located on the same FTP server where the Event Directory is located, for example: /home/archive.</p> <p>Note: The FTPArchiveDirectory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter will not create this directory automatically.</p>
Globalized	Yes

Remote directory property (EventDirectory)

Remote directory of the FTP server from where the event files are retrieved for inbound processing.

Table 136. Remote directory property characteristics

Required	Yes
Default	None
Property type	String
Usage	<p>You can use a WebSphere Application Server environment variable to represent the remote directory. Specify the name of the environment variable in braces, preceded by a \$ symbol. For example: \${REMOTE_DIRECTORY}. See the topic on “Defining WebSphere Application Server environment variables” on page 66 in this documentation.</p> <p>Note: The EventDirectory has to be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter will not create this directory automatically.</p>
Globalized	Yes

Retrieve files with this pattern property (EventFileMask)

Filter for the event files. The file filter is a well-qualified expression consisting of alphanumeric characters and the * and ? wild cards.

Table 137. Retrieve files with this pattern property characteristics

Required	Yes
Default	*.*
Property type	String
Globalized	Yes

Retry EIS connection on startup (RetryConnectionOnStartup)

This property controls whether the adapter attempts to connect again to the FTP server if it cannot connect at startup.

Table 138. Retry EIS connection on startup details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	<p>This property indicates whether the adapter should retry the connection to the FTP server if the connection cannot be made when the adapter is started:</p> <ul style="list-style-type: none">• Set the property to <code>False</code> when you want immediate feedback about whether the adapter can establish a connection to the FTP server, for example, when you are building and testing the application that receives events from the adapter. If the adapter cannot connect, the adapter writes log and trace information and stops. The administrative console shows the application status as <code>Stopped</code>. After you resolve the connection problem, start the adapter manually.• Set the property to <code>True</code> if you do not need immediate feedback about the connection. If the adapter cannot connect during startup, it writes log and trace information, and then attempts to reconnect, using the <code>RetryInterval</code> property to determine how frequently to retry and the value of the <code>RetryLimit</code> property to retry multiple times until that value is reached. The administrative console shows the application status as <code>Started</code>.
Globalized	No
Bidi supported	No

Retry interval if connection fails (RetryInterval)

When the adapter encounters an error related to the inbound connection, this property specifies the length of time the adapter waits before trying to establish a new connection.

Table 139. Retry interval details

Required	Yes
Default	2000
Unit of measure	Milliseconds
Property type	Integer

Table 139. Retry interval details (continued)

Usage	Only positive values are valid. When the adapter encounters an error related to the inbound connection, this property specifies the length of time the adapter waits before trying to establish a new connection.
Globalized	Yes
Bidi supported	No

Number of times to retry the system connection (RetryLimit)

This property specifies the number of times the adapter tries to reestablish an inbound connection.

Table 140. Number of times to retry the system connection details

Required	No
Possible values	0 and positive integers
Default	0
Property type	Integer
Usage	This property controls how many times the adapter retries the connection if the adapter cannot connect to the FTP server to perform inbound processing. A value of 0 indicates an infinite number of retries. To control whether the adapter retries if it cannot connect to the FTP server when it is first started, use the RetryConnectionOnStartup property.
Globalized	Yes
Bidi supported	No

Enable server verification property (EnableServerVerification)

This property is used to enable the remote server verification for SFTP protocol.

Table 141. Enable server verification property details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	When this property is set to: <ul style="list-style-type: none"> • True, server authentication is enabled • False, server authentication is disabled The adapter checks the HostKeyFile property in the path of the file containing the host keys of the trusted servers.
Globalized	Yes
Bidi supported	No

Host key file property (HostKeyFile)

This property provides the absolute path of the host key file that contains the host key of the trusted servers.

Table 142. Host key file property characteristics

Required	This property has to be specified if the EnableServerVerification property is enabled.
Default	None
Property type	String
Usage	This is used by the adapter to verify the host key of the remote server with the host keys of the trusted servers specified in this file.
Globalized	Yes
Bidi supported	No

Host name property (SocksProxyHost)

Host name of the machine used as a proxy server through which the adapter requests are routed to the FTP server.

Table 143. Host name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SocksProxyPassword)

Password used to authenticate the proxy server.

Table 144. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Port number property (SocksProxyPort)

Port number of the proxy server through which the adapter requests are routed to the FTP server.

Table 145. Port number property characteristics

Required	No
Default	1080
Property type	Integer
Globalized	No

User name property (SocksProxyUserName)

User name used to authenticate the proxy server.

Table 146. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Sort event files property (SortEventFiles)

Determines the sorting order of event files being polled. Supported values are:

- by file name – sort ascending on file name
- by time stamp – sort ascending on last modified timestamp
- no sort – not sorted

Event file ordering from which events need to be delivered is valid only if the activation specification DeliveryType property is set to ORDERED. file name sorting is provided based on the locale of the FTP server. The ICU4J package is used to track the locales and their corresponding rules.

Table 147. Sort event files property characteristics

Required	No
Default	no sort (= not sorted)
Property type	String
Globalized	No

Specify criteria to split file content property (SplitCriteria)

This property takes different values based on the value of the SplittingFunctionClassName property. For example: To specify that a file is to be split every 5KB, set the SplitCriteria property to 5000.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
 - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
 - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=0, chunking is disabled.

When FilePassByReference is enabled during inbound PassThrough, the event file is not split.

Note: For input files that contain multiple COBOL copybook records, in order to enable file splitting by size you must provide the correct length of each record. To determine the size of each record, use one of these methods:

1. Open the Business Object in a text editor.
 - a. For example:


```
<element name="CustomerNumber">
<annotation>
<appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
```

```

<td:typeDescriptorElement>
<td:initialValue kind="SPACE"/>
<td:simpleInstanceTD accessor="readWrite" attributeInBit="false"
contentSize="5" offset="0" size="5">
<td:sharedType>
<td:stringTD addrUnit="byte" alignment="byte" characterSize="1"
lengthEncoding="fixedLength" paddingCharacter=" "
prefixLength="0" width="5"/>
</td:sharedType>
</td:simpleInstanceTD>
</td:typeDescriptorElement>
</appinfo>
</annotation>
<simpleType>
  <restriction base="string">
    <maxLength value="5"/>
  </restriction>
</simpleType>
</element>

```

Each element in the business object has a corresponding <element> entry.

- b. Look for a restriction tag for each element tag (the COBOL data binding requires a fixed-width data handler).
 - c. Add up the maxLength attribute values for each of the elements. In this example, the value is 5. The sum of the maxLength values is the size of each record of type DFHCOMMAREA.
2. Open the Business Object in a text editor.
 - a. Look for the complex type tag with the business object name value in the name attribute. In the example that follows, the business object name is DFHCOMMAREA.
 - b. Locate a namespace-appended tag called aggregateInstanceTD and use the value for the attribute contentSize. In this example, the value is 117. This is the size of each record of type DFHCOMMAREA.

```

<complexType name="DFHCOMMAREA">
<annotation>
<appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
<td:typeDescriptorCT>
<td:aggregateInstanceTD accessor="readWrite" attributeInBit="false"
contentSize="117" offset="0" size="117">

```

Table 148. Specify criteria to split file content property characteristics

Required	No
Default	0
Property type	String
Globalized	Yes

Splitting function class name property

This value takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The com.ibm.j2ca.utils.filesplit.SplitByDelimiter class that splits the event file based on delimiter.
- The com.ibm.j2ca.utils.filesplit.SplitBySize class that splits the event file based on the event file size.

Optionally, you can provide a custom file splitter class and use it by inputting the class name into the SplittingFunctionClassName property.

The delimiter or file size is provided in the SplitCriteria property. If the EventContentType property is set to null, it is automatically set to a class name that performs splitting based on file size.

Table 149. Splitting function class name property characteristics

Required	No
Default	com.ibm.j2ca.utils.filesplit.SplitBySize
Property type	String
Globalized	No

Stop the adapter when an error is encountered while polling (StopPollingOnError)

This property specifies whether the adapter will stop polling for events when it encounters an error during polling.

Table 150. Stop the adapter when an error is encountered while polling details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	If this property is set to True, the adapter stops polling when it encounters an error. If this property is set to False, the adapter logs an exception when it encounters an error during polling and continues polling.
Globalized	No
Bidi supported	No

Success file extension for local archive property (SuccessArchiveExt)

File extension used to archive all of the successfully processed business objects. This property is used only when LocalArchiveDirectory is valid and exists. For example, 12345.order > 12345.order.success

Table 151. Success file extension for local archive property characteristics

Required	No
Default	success
Property type	String
Globalized	Yes

User name property (UserName)

Name of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the User name is included in the URL specified in the EventDirectory property.

Table 152. User name property characteristics

Required	No
----------	----

Table 152. User name property characteristics (continued)

Default	None
Property type	String
Globalized	Yes

User name used to connect to event data source property (EP_UserName)

User name used by event persistence for getting the database connection from the data source.

Table 153. User name used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Rule based event filtering (ruleTable)

This property is used to filter event files based on a set of rules

Table 154. Rules based event filtering

Required	Optional
Default	None
Property type	String
Usage	During an inbound processing, if the value in the rule table is specified, then the event files are fetched after filtering, based on the specified rules before polling those event files.
Globalized	Yes
Bidi supported	No

Globalization

WebSphere Adapter for FTP is a globalized application that can be used in multiple linguistic and cultural environments. Based on character set support and the locale of the host server, the adapter delivers message text in the appropriate language. The adapter supports bidirectional script data transformation between integration components.

Globalization and bidirectional transformation

The adapter is globalized to support single- and multi-byte character sets and deliver message text in the specified language. The adapter also performs bidirectional transformation, which refers to the task of processing data that contains both left-to-right (Hebrew or Arabic, for example) and right-to-left (a URL or file path, for example) semantic content within the same file.

Globalization

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- and multi-byte). Components in the WebSphere Business Integration system are written in Java. Therefore, when data is transferred between WebSphere Business Integration system components, there is no need for character conversion.

To log error and informational messages in the appropriate language and for the appropriate country or region, the adapter uses the locale of the system on which it is running.

Bidirectional transformation

Languages such as Arabic and Hebrew are written from right to left, yet they contain embedded segments of text that are written left to right, resulting in bidirectional script. When software applications handle bidirectional script, standards are used to display and process it. WebSphere Process Server and WebSphere Enterprise Service Bus use the Windows standard format, but an enterprise information system exchanging data with WebSphere Process Server or WebSphere Enterprise Service Bus can use a different format. WebSphere Adapters transform bidirectional script data passed between the two systems so that it is accurately processed and displayed on both sides of a transaction.

Bidirectional format

WebSphere Process Server and WebSphere Enterprise Service Bus use the bidirectional format of ILYNN (implicit, left-to-right, on, off, nominal). This is the format used by Windows. If an enterprise information system uses a different format, the adapter converts the format prior to introducing the data to WebSphere Process Server or WebSphere Enterprise Service Bus.

Five attributes comprise bidirectional format. When you set bidirectional properties, you assign values for each of these attributes. The attributes and settings are listed in the following table.

Table 155. Bidirectional format attributes

Letter position	Purpose	Values	Description	Default setting
1	Order schema	I	Implicit (Logical)	I
		V	Visual	
2	Direction	L	Left-to-Right	L
		R	Right-to-Left	
		C	Contextual Left-to-Right	
		D	Contextual Right-to-Left	
3	Symmetric Swapping	Y	Symmetric swapping is on	Y
		N	Symmetric swapping is off	
4	Text Shaping	S	Text is shaped	N
		N	Text is not shaped (Nominal)	
		I	Initial shaping	
		M	Middle shaping	
		F	Final shaping	
		B	Isolated shaping	

Table 155. Bidirectional format attributes (continued)

Letter position	Purpose	Values	Description	Default setting
5	Numeric Shaping	H	National (Hindi)	N
		C	Contextual shaping	
		N	Numbers are not shaped (Nominal)	

The adapter transforms data into a logical, left-to-right format before sending the data to WebSphere Process Server or WebSphere Enterprise Service Bus.

Using bidirectional properties

You can use multiple bidirectional properties to control the transformation of both content data and metadata. You can set special bidirectional properties to exclude either content data or metadata from bidirectional transformation, or to identify data that requires special treatment during a transformation.

The following table describes four types of bidirectional properties.

Table 156. Bidirectional property types

Property type	Data transformations
EIS	Controls the format for content data, or data that is sent by the enterprise information system.
Metadata	Controls the format for metadata, or data that provides information about the content data.
Skip	Identifies content or metadata to exclude from transformation.
Special Format	Identifies certain text, such as file paths or URLs, that require different treatment during the transformation process. Can be set for either content data or metadata.

You can set properties that control bidirectional transformation in three areas.

- **Resource adapter properties:** These properties store default configuration settings, including the TurnBiDiOff property, which controls whether the adapter instance performs bidirectional transformation or not. Use the administrative console of the server to configure these properties.
- **Managed (J2C) connection factory properties:** These properties are used at run time to create an outbound connection instance with an enterprise information system. Once the managed connection factory properties are created, they are stored in the deployment descriptor.
- **Activation specification properties:** These properties hold the inbound event processing configuration information for a message endpoint. Set them as you perform external service, or use the administrative console of the server.

Business object annotations

Some adapters allow you to annotate bidirectional properties within a business object. Do this to add information that specifically controls the transformation of a

business object or part of a business object. Use business object editor, a tool within WebSphere Integration Developer, to add annotations at these levels:

- Business object
- Business object application-specific attribute
- Business object attribute
- Business object attribute application-specific attribute

Property scope and lookup mechanism

After you set values for bidirectional properties for an adapter and annotate business objects where appropriate, the adapter performs bidirectional transformations. It does so by using logic that relies on a hierarchical inheritance of property settings and a lookup mechanism.

Properties defined within the resource adapter are at the top of the hierarchy, while those defined within other areas or annotated within a business object are at lower levels of the hierarchy. So for example, if you only set values for EIS-type bidirectional properties for the resource adapter, those values are inherited and used by transformations that require a defined EIS-type bidirectional property whether they arise from an inbound (activation specification) transaction or an outbound (managed connection factory) transaction.

However, if you set values for EIS-type bidirectional properties for both the resource adapter and the activation specification, a transformation arising from an inbound transaction uses the values set for the activation specification.

The processing logic uses a lookup mechanism to search for bidirectional property values to use during a transformation. The lookup mechanism begins its search at the level where the transformation arises and searches upward through the hierarchy for defined values of the appropriate property type. It uses the first valid value it finds. It searches the hierarchy from child to parent only; siblings are not considered in the search.

Bidirectional transformation in business objects

For outbound processing, you can modify the business objects to enable the bidirectional transformation of the wrapper properties in the WebSphere Adapter for FTP business object and the data in content-specific or generic business objects.

You must add an annotation to the complex type of the business object to specify the bidirectional formatting attributes in the files for the following business objects:

- For the generic business object, change the FTPFile.xsd file.
- For the user-defined business object, change the customer wrapper (for example, the CustomWrapper.xsd file and Customer.xsd).
- For the UnstructuredContent business object, change the UnstructuredContent.xsd.

The following sections include annotations that can serve as examples.

Bidirectional formatting attributes of the business object

The following annotation, which contains the bidirectional context information, applies to all the attributes in the FTP business objects. The FTPFileBaseDataBinding uses the bidirectional information in the element BiDiContext to transform all the attributes.

```

<xsd:complexType name="Customer">
<xsd:annotation>
  <xsd:appinfo
    source="http://www.ibm.com/xmlns/prod/websphere/j2ca/datatrans
formation/databindingm
apping">
  <dtm:DataBindingMapping
    xsi:type="dtm:DataBindingMapping"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:dtm="http://www.ibm.com/xmlns/prod/websphere/j2ca/da
tatransformation/databindingmapping">
    <BiDiContext>
      <orientation>rtl</orientation>
      <textShape>nominal</textShape>
      <orderingScheme>visual</orderingScheme>
      <symmetricSwapping>true</symmetricSwapping>
      <numeralShapes>nominal</numeralShapes>
    </BiDiContext>
  </dtm:DataBindingMapping>
</xsd:appinfo>
</xsd:annotation>

```

Bidirectional formatting attributes of the wrapper

You can add an annotation to the wrapper of a user-defined type business object. The annotation in the wrapper business objects such as the generic (FTPFile) and the user-defined type (CustomerWrapper) is used to do bidirectional transformation of wrapper attributes. The content-specific business objects that are used inside the wrapper business objects are not transformed using annotation in the wrapper business objects. (To transform content-specific business objects, you must edit the respective business object definition to add the annotation shown in the previous example for bidirectional formatting of attributes of the business object.

The following annotation is an example for the wrapper:

```

<complexType name="CustomerWrapper">
<annotation>
  <appinfo
    source="http://www.ibm.com/xmlns/prod/websphere/j2ca/
datatransformation/databindingmapping">
  <dtm:DataBindingMapping
    xsi:type="dtm:DataBindingMapping"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:dtm="http://www.ibm.com/xmlns/prod/websphere/j2ca/
datatransformation/databindingmapping">
    <BiDiContext>
      <orientation>rtl</orientation>
      <textShape>nominal</textShape>
      <orderingScheme>visual</orderingScheme>
      <symmetricSwapping>true</symmetricSwapping>
      <numeralShapes>nominal</numeralShapes>
    </BiDiContext>
  </dtm:DataBindingMapping>
</appinfo>
</annotation>

```

Properties enabled for bidirectional data transformation

Bidirectional data transformation properties enforce the correct format of bidirectional script data exchanged between an application or file system and integration tools and runtime environments. After these properties are set,

bidirectional script data is correctly processed and displayed in WebSphere Integration Developer and WebSphere Process Server or WebSphere Enterprise Service Bus.

Managed (J2C) connection factory properties

The following managed (J2C) connection properties control bidirectional transformation.

- Username
- Password
- Directory
- FileName
- StagingDirectory
- SecondServerUsername
- SecondServerPassword
- SecondServerDirectory
- SocksProxyUsername
- SocksProxyPassword
- FileSequenceLog

Activation specification properties

The following activation specification properties control bidirectional transformation.

- Username
- Password
- EventDirectory
- EventFileMask
- FTPArchiveDirectory
- LocalEventDirectory
- LocalArchiveDirectory
- FTPScriptFileExecutedBeforeInbound
- FTPScriptFileExecutedAfterInbound
- FTPRenameExt
- FailedArchiveExt
- OriginalArchiveExt
- SuccessArchiveExt
- SocksProxyUsername
- SocksProxyPassword

Deployment Descriptor configuration properties

The following Deployment Descriptor configuration properties control bidirectional transformation.

- EPDataSourceJNDIName
- EPEventTableName
- EPDatabaseUsername
- EPDatabasePassword

- EPDatabaseSchemaName

Wrapper business object properties

The following wrapper business object properties control bidirectional transformation.

- DirectoryPath
- Filename
- FtpServerEventDirectory
- SecondServerDirectory
- SecondServerUsername
- SecondServerPassword
- LocalDirectoryPath
- LocalArchiveDirForCreate
- StagingDirectory
- ArchiveDirectoryForRetrieve

Adapter messages

View the messages issued by WebSphere Adapter for FTP at the following location.

Link to messages: <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r2mx/topic/com.ibm.wbit.620.help.messages.doc/messages.html>

The displayed Web page shows a list of message prefixes. Click a message prefix to see all the messages with that prefix:

- Messages with the prefix CWYFT are issued by WebSphere Adapter for FTP
- Messages with the prefix CWYBS are issued by the adapter foundation classes, which are used by all the adapters

Related information

The following information centers, IBM Redbooks, and Web pages contain related information for the WebSphere Adapter for FTP.

Samples and tutorials

To help you use WebSphere Adapters, samples and tutorials are available from the Business Process Management Samples and Tutorials Web site. You can access the samples and tutorials in either of the following ways:

- From the welcome page that opens when you start WebSphere Integration Developer. To see samples and tutorials for WebSphere Adapter for FTP, click **Retrieve**. Then browse the displayed categories to make your selections.
- At this location on the Web: <http://publib.boulder.ibm.com/bpcsamp/index.html>.

Information resources

- The WebSphere Business Process Management information resources Web page includes links to articles, Redbooks, documentation, and educational offerings to help you learn about WebSphere Adapters: <http://www14.software.ibm.com/webapp/wsbroker/redirect?version=pix&product=wps-dist&topic=bpmroadmaps>

- The WebSphere Adapters library page includes links to all versions of the documentation: <http://www.ibm.com/software/integration/wbiadapters/library/infocenter/>

Information about related products

- WebSphere Business Process Management, version 6.2, information center, which includes WebSphere Process Server, WebSphere Enterprise Service Bus, and WebSphere Integration Developer information: <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r2mx/index.jsp>
- WebSphere Adapters, version 6.1.x, information center: <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r1mx/topic/com.ibm.wbit.612.help.adapter.emd.ui.doc/topics/tcreatecmps.html>
- WebSphere Business Integration Adapters information center: http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/index.jsp?topic=/com.ibm.wbi_adapters.doc/welcome_adapters.htm

developerWorks® resources

- WebSphere Adapter Toolkit
- WebSphere business integration zone

Support and assistance

- WebSphere Adapters technical support: <http://www.ibm.com/software/integration/wbiadapters/support/>
- WebSphere Adapters technotes: <http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm>. In the **Product category** list, select the name of the adapter and click **Go**.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other 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 grant 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.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

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 the 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 Corporation
Department 2Z4A/SOM1
294 Route 100
Somers, NY 10589-0100
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 measurements 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 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.

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

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples 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.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate 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.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows: (c) (your company name) (year). Portions of

this code are derived from IBM Corp. Sample Programs. (c) Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning:

Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A complete and current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

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

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

Java and all Java based trademarks and logos 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.

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

This product includes software developed by the Eclipse Project (<http://www.eclipse.org>).

Index

A

- activation specification properties
 - list of 189
 - setting in administrative console 127, 132
- Activation specification properties 189
- adapter application
 - starting 133
 - stopping 133
- adapter architecture 2
- Adapter for FTP
 - administering 123
- Adapter for FTP module
 - exporting as EAR file 120
 - installing EAR file on server 121
 - starting 133
 - stopping 133
- Adapter implementation 21
- adapter messages 218
- adapter patterns wizard 68
- adapter performance 134
- adapter technotes 219
- alias, authentication 41
- annotation 3
- Append 4
- archive, event 16
- ArchiveDirectoryForRetrieve 174
- artifacts, generating 90
- assembly editor 107
- authentication
 - description 27
 - run time 27
- Authentication 25
- authentication alias 27, 41

B

- backward compatibility
 - project interchange files 35
 - projects 35
- business faults 144
- business graph 3
- business object attribute properties 153
- business object operation support 153
- business object structure 149
- business object, custom 154
- business object, predefine 42, 44
- business objects 17, 149
 - naming conventions 152
- business objects, converting COBOL copybook files into 56
- business objects, converting into COBOL copybook files 46

C

- CEI (Common Event Infrastructure) 136
- CharacterSet 79, 92
- chunking 13
- clustered environment
 - deploying in 30
 - description 30
 - inbound processes 31

- clustered environment (*continued*)
 - outbound processes 31
- COBOL copybook files, converting from business objects 46
- COBOL copybook files, converting into business objects 56
- Common Event Infrastructure (CEI) 136
- compatibility matrix 1
- confidential data, disguising 26
- confidential tracing 26
- configuring
 - logging 141
 - Performance Monitoring Infrastructure (PMI) 134
 - tracing 141
- Create 4
- CreateFileIfNotExists 174
- custom business objects 154
- custom properties
 - activation specification 127, 132
 - managed connection factory 125, 130
 - resource adapter 123, 129
- CustomerWrapper 3
- CustomerWrapper business object 149
- CustomerWrapperBG 3, 149

D

- data transformation framework: inbound 12
- data transformation framework: outbound 3
- DataConnectionMode 175
- debugging
 - org.xml.sax.SAXParseException exception 146
 - self-help resources 146
 - XAResourceNotAvailableException exception 145
- Delete 4
- DeleteOnRetrieve 175
- delimiter 12, 13
- deployment
 - environments 115
 - options 27
 - to production environment 119
 - to test environment 115
- Description 157, 183
- developerWorks 219
- developerWorks resources, WebSphere Adapters 218
- Directory Path 175
- DisplayName 157, 184

E

- EAR file
 - exporting 120
 - installing on server 121
- education, WebSphere Adapters 218
- embedded adapter
 - activation specification properties, setting 127
 - considerations for using 29
 - description 27
 - managed connection factory properties, setting 125
 - resource adapter properties, setting 123
- EmbeddedNameFunctionSelector 11
- enableHASupport property 31

- endpoints 107
- event archive, archiving on MVS platforms 16
- event recovery 13
- event store 15
- exceptions
 - org.xml.sax.SAXParseException 146
 - XAResourceNotAvailableException 145
- ExecuteFTPScript 4
- Exists 4
- exporting module as EAR file 120
- External service connection properties 156, 183
- external service wizard
 - connection properties 93
 - start up 93
 - starting 78

F

- faults
 - description 144
- Federal information processing standard 24
- FFDC (first-failure data capture) 143
- File splitting 13
- FileContentEncoding 176
- FileInLocalDirectory 176
- Filename 176
- FilenameFunctionSelector 11
- files
 - SystemOut.log log file 142
 - trace.log trace file 142
- FileTransferType 176
- firewall 21
- first-failure data capture (FFDC) 143
- FTPFile 3
- FTPFileBG 3
- FTPFileBG business object 149
- function selector 11

G

- GenerateUniqueFile 177
- generating artifacts 90
- generating artifacts, inbound 111

H

- hardware and software requirements 1
- hardware requirements 1
- high-availability environment
 - deploying in 30
 - description 30
 - inbound processes 31
 - outbound processes 31

I

- IBM WebSphere Adapter Toolkit 219
- ID 157, 184
- implementation, Java 116
- inbound configuration properties 181
- Inbound event processing 9
- Inbound processing 9
- IncludeEndBODelimiter 177
- installing EAR file 121

- interaction specification properties
 - changing 113
- Interaction specification properties 173
- introduction 1

J

- J2EE component 85
- Java 2 security 27
- Java implementation 116

L

- List 4
- LocalArchiveDirForCreate 178
- LocalArchivingEnabledForCreate 178
- LocalDirectoryPath 178
- Log Analyzer 141
- Log and Trace Analyzer, support for 139
- log and trace files 139
- log files
 - changing file name 142
 - disabling 141
 - enabling 141
 - level of detail 141
 - location 143
- logging
 - configuring properties with administrative console 141

M

- managed (J2C) connection factory properties
 - setting in administrative console 125, 130
- Managed (J2C) connection factory properties 162
- matrix, compatibility 1
- messages, adapter 218
- migration considerations 32
- module, creating 42
- monitoring performance 134

N

- naming conventions for business objects 152

O

- org.xml.sax.SAXParseException 146
- outbound configuration properties 155
- Outbound processing 2
- overview 1
- Overwrite 4

P

- package files for adapters 142
- passive FTP mode 21
- Passthrough processing 13
- patterns 68
- Performance Monitoring Infrastructure (PMI)
 - configuring 134
 - description 134
 - viewing performance statistics 137
- performance statistics 137

- PMI (Performance Monitoring Infrastructure)
 - configuring 134
 - description 134
 - viewing performance statistics 137
- pre-defined business objects 154
- problem determination
 - org.xml.sax.SAXParseException exception 146
 - self-help resources 146
 - XAResourceNotAvailableException exception 145
- product overview 1
- project interchange (PI) file
 - updating without migrating 35
- project, creating 78
- properties
 - activation specification 127, 132
 - list of 189
 - configuration properties
 - inbound 181
 - outbound 155
 - inbound configuration 181
 - managed (J2C) connection factory 125, 130
 - outbound configuration 155
 - resource adapter 123, 129
- Public key 25

R

- RAR (resource adapter archive) file
 - description 119
 - installing on server 119
- recovery feature 13
- Redbooks, WebSphere Adapters 218
- reference bindings 85
- reference bindings, inbound 107
- related information 218
- related products, information 218
- requirements, hardware and software 1
- resource adapter archive (RAR) file
 - description 119
 - installing on server 119
- resource adapter properties
 - details 158, 185
 - setting in administrative console 123, 129
- Retrieve 4
- Retry limit property 207
- roadmap for configuring the module 39
- runtime environment
 - authentication in 27
 - deploying EAR file to 119

S

- samples 37
- ScriptFileParameters 179
- SecondServerDirectory 174
- SecondServerHostName 177
- SecondServerPassword 180
- SecondServerPortNumber 179
- SecondServerProtocol 179
- SecondServerUsername 180
- secure FTP 21
- Secure socket layer (SSL) 21
- security
 - disguising sensitive data 26
- Security 21
- security, Java 2 27

- Selecting business objects and services: Outbound 81
- selecting business objects, inbound 105
- self-help resources 146
- sensitive data, disguising 26
- ServerToServerFileTransfer 4
- Setting connection properties 79, 92
- SFTP 24
- software requirements 1
- SplitByDelimiter 13
- SplitBySize 13
- SplitCriteria 12, 13, 180
- SplittingFunctionClassName 181
- SSH over FTP 24
- SSL communication 21
- StagingDirectory 181
- stand-alone adapter
 - activation specification properties, setting 132
 - considerations for using 30
 - description 27
 - managed connection factory properties, setting 130
 - resource adapter properties, setting 129
- stand-alone reference 85
 - wiring 107
- starting adapter applications 133
- stopping adapter applications 133
- support
 - overview 139
 - self-help resources 146
 - technical 219
- Supported outbound operations 4
- SystemOut.log file 142

T

- target component 115
- Technical overview 2
- technical support 219
- technotes 1, 146, 219
- technotes, WebSphere Adapters 218
- test environment
 - adding module to 117
 - deploying to 115, 117
 - testing modules 118
- trace files
 - changing file name 142
 - disabling 141
 - enabling 141
 - level of detail 141
 - location 143
- trace.log file 142
- tracing
 - configuring properties with administrative console 141
- troubleshooting
 - org.xml.sax.SAXParseException exception 146
 - overview 139
 - self-help resources 146
 - XAResourceNotAvailableException exception 145
- trust store 21, 24
- tutorials 37

V

- Vendor 157, 184
- Version 158, 184

W

- WebSphere Adapters, version 6.0, information 219
- WebSphere Adapters, version 6.0.2.x, information 219
- WebSphere Application Server environment variables 18
- WebSphere Application Server environment variables, defining 66
- WebSphere Application Server information 219
- WebSphere Business Integration Adapters information 219
- WebSphere Business Process Management, version 6.1.x, information 219
- WebSphere Enterprise Service Bus information 219
- WebSphere Extended Deployment 30
- WebSphere Integration Developer
 - information 219
 - starting 42, 44, 78
 - test environment 115
- WebSphere Process Server information 219
- WebSphere Process Server or WebSphere Enterprise Service Bus
 - deploying to 119
- wiring 85
- wiring components 115
- wrapper business object 3

X

- XAResourceNotAvailableException 145
- xsd files 149



Printed in USA