IBM WebSphere Adapters
Version 7 Release 5

IBM WebSphere Adapter for Flat Files User Guide Version 7 Release 5

IBM WebSphere Adapters
Version 7 Release 5

IBM WebSphere Adapter for Flat Files User Guide Version 7 Release 5

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

November 2011

This edition applies to version 7, Release 5, Fix Pack 1 (7.5.0.1) of IBM WebSphere Adapter for Flat Files 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.

Contents

Chapter 1. Overview of WebSphere Adapter for Flat Files

With WebSphere® Adapter for Flat Files, you can create integrated processes that can exchange data with the local file system, without special coding.

You can use the adapter to read data from a file in the local file system, use it in an application on IBM® Business Process Manager or WebSphere Enterprise Service Bus, and send it back to the local file system. You can also use the adapter to poll a directory in the local file system for new files and send them to an application for processing.

The adapter can be used to read and write any type of file stored in the local file system. It can:

- Create new files
- Append or overwrite existing files
- Retrieve the content of a given file, retrieve a list of file names in a directory, or delete a file
- Check whether a particular file exists or not
- Poll a directory for new files and send them to an application for processing

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

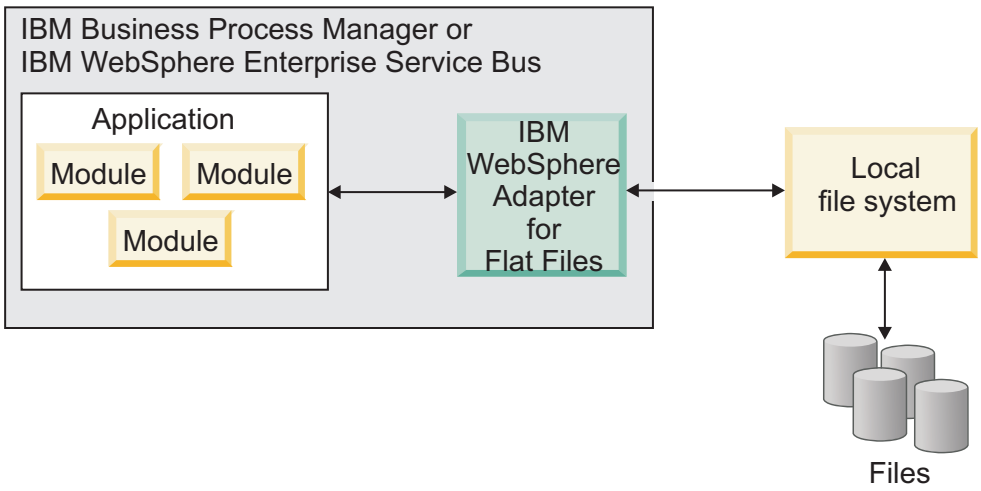


Figure 1. Adapter overview

Hardware and software requirements

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

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: http://www-947.ibm.com/support/entry/portal/Overview/Software/WebSphere/WebSphere_Adapters_Family.
- Technotes for WebSphere Adapters provide workaround 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 WebSphere Adapter for Flat Files

IBM WebSphere Adapter for Flat Files makes it possible for services running on IBM Business Process Manager or WebSphere Enterprise Service Bus to exchange data with the local file system.

Services can use the adapter to exchange data with the local file system in two ways:

- Through *outbound processing*, services running on IBM Business Process Manager or WebSphere Enterprise Service Bus use the adapter to perform operations on files in the local file system, for example, to update an order document.
- Through *inbound processing*, services running on IBM Business Process Manager or WebSphere Enterprise Service Bus use the adapter to receive events from the local file system, for example, to be notified that a customer record has been updated.

You configure the adapter to perform either outbound or inbound processing through the external service wizard, launched through IBM Integration Designer. Using the external service wizard, you can create a *module*, which consists of a project in IBM Integration Designer and a unit of deployment to IBM Business Process Manager or WebSphere Enterprise Service Bus. Each module contains components that make up a service for either an *import* or an *export*:

- An *import* is the point at which a SCA module accesses an external service (a service outside the Service Component Architecture (SCA) module) as if it were local. An import defines interactions between the SCA module and the service provider. An import consists of a binding and one or more interfaces.
- An *export*, also known as an endpoint, is an exposed interface from a SCA module that offers a business service to the outside world. An export has a binding that defines how the service can be accessed by service requesters, for example, as a Web service.

The module is packaged and deployed to IBM Business Process Manager or WebSphere Enterprise Service Bus as an enterprise archive (EAR) file.

To represent files that are exchanged between a module and the local file system, the adapter uses business objects. A business object is a logical data container that contains the data that is processed by the adapter. You can create business objects using the external service wizard or by using the business object editor in IBM Integration Designer.

The adapter uses adapter-specific data bindings and data handlers to transform data from one format to another during inbound and outbound processing. Data bindings are essentially maps that define how a business object is to be formatted.

A data binding reads the fields in a business object and populates the corresponding fields in the file. The data binding that is used depends on the internal format of the file. Each data type has an equivalent data binding. You use the external service wizard to configure the data binding.

Data handlers perform the conversions between a business object and a native format. When you select a data type that contains business objects, you must specify the data handler that performs the conversion. Data handlers are provided by IBM Business Process Manager or WebSphere Enterprise Service Bus.

Outbound processing

During outbound processing, the adapter receives a request from the module, in the form of a business object, to perform an operation on a file in the local file system. The adapter performs the requested operation and returns a business object, if applicable, that represents the result of the operation to the component.

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

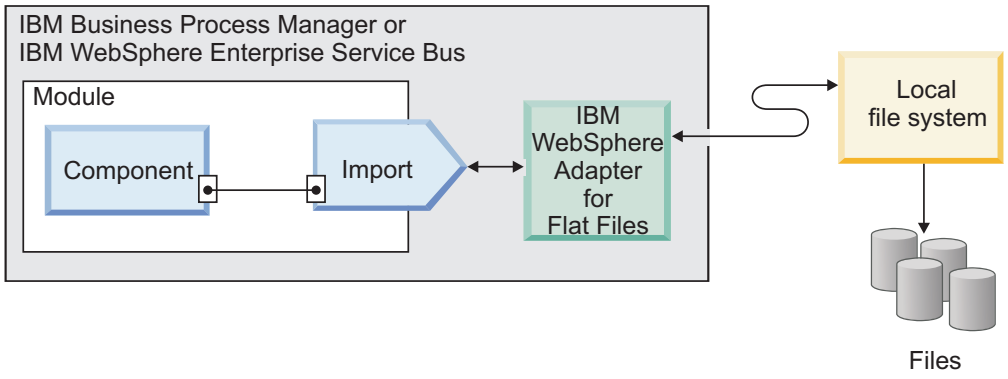


Figure 2. Outbound processing

Outbound operations

An operation is the action that an adapter can perform on the local file system during outbound processing. The name of the operation indicates the type of action that the adapter takes.

The adapter supports the following operations during outbound processing.

Append operation:

The Append operation appends content to a specified file.

If you select the **Enable response type for the operation** check box in the external service wizard, the file name is returned to the component in a business object.

You can specify a delimiter between business objects in a file by using the IncludeEndBODelimiter property. When this property is not specified, its default value `<EndBO>` gets appended at the end of each business object in a file.

Note: If required, you can define any escape sequence character and Unicode escape characters as the value for the endBODelimiter property in the "Interaction specification properties" on page 180.

If the CreateFileIfNotExists property is set to `true` and the file does not exist, the adapter creates a file. If the GenerateUniqueFile property is set to `true`, the adapter generates a unique file and ignores the value in the Filename property.

Note: The GenerateUniqueFile property has been deprecated. Although you can currently set this property, but the adapter always treats the value for this property as `false`.

If a staging directory is specified in the StagingDirectory property, the file to be appended is copied from the output directory to the staging directory, and the content is added for that file in the staging directory. The file is then moved back to the output directory. If a staging directory is not specified, the content is added on the file in the output directory.

If the file that is to be appended does not exist and the CreateFileIfNotExists property is set to `false`, the adapter generates a RecordNotFoundException error.

If the Filename property has no value, the adapter generates a MissingDataException error.

Note: For a wrapper business object, if you do not specify the value for the CreateFileIfNotExists property on the wrapper, then the adapter uses the value specified in its interaction specification property.

Create operation:

The Create operation creates a file with the specified name. You can modify the file name by specifying different properties. For example, you can attach a sequence number to the file.

If you select the **Enable response type for the operation** check box in the external service wizard, the file name is returned to the component in a business object. If a file with the specified name exists, the adapter generates a DuplicateRecordException error, and no file is created.

If a staging directory is specified in the StagingDirectory property, the file that is to be created is copied from the output directory to the staging directory, and the content is written for that file in the staging directory. The file is then moved back to the output directory. If a staging directory is not specified, the content is written on the file in the output directory.

Note: You can configure a staging directory only if the file content is to be written before the Create operation returns the resultant values. You cannot use a staging directory if the Create operation returns an output stream and the component writes to this stream.

Generate unique file name during Create operation

You can configure the adapter to generate a unique file name in the wrapper business object during run time. If you specify the GenerateUniqueFile property as `True` in the wrapper business object, the adapter generates a unique file name and ignores the value specified in the Filename property. The name of the unique file that is generated by the adapter is in the form of a random number. You can specify a prefix and suffix (file extension) for the file name. For example, a file name with an **abcd** prefix and a suffix of **.xyz**, can be: **abcd23423.xyz**, where **23423** is the randomly generated number by the adapter.

Note: You can also specify a staging directory, when configuring the adapter to generate a unique file name during the Create operation.

The adapter selects the values for the unique file name by using the following order of precedence:

- Wrapper business object properties
- Interaction specification properties
- Managed connection factory properties
- If no values are found from the preceding three properties, the adapter uses the default values.

Note: You must specify a minimum of three letters for the prefix. If you do not specify the prefix and suffix for the unique file name, the adapter uses `ffa` as the prefix and `.tmp` as the file extension during the unique file name generation.
If you do not specify the GenerateUniqueFile property at the wrapper level, the adapter sets the value as `False` and does not generate a unique file name.

Sequence file

The adapter appends a sequence number to the output file name to create a sequence file along with the output file. For example, if the output file name in the request is `Customer.txt`, a file with the name `Customer.`*n*`.txt` is created, where *n* is the sequence number for the request. If another request with an output file name of `Order.txt` is received, the sequence number increments by 1 and `Order.n+1.txt` is generated.

If the FileSequenceLog managed connection property is specified, the adapter appends a sequence number to the output file name specified in the request and the next request uses the sequence number in the sequence file.

No new sequence number is created for each individual file name. If the output file name does not have an extension, the sequence is appended at the end of the file name. For example, if the output file name in the request is Customer, a file with the name `Customer`*n* is created.

To avoid setting the output directory and file name in the business object for each request, you can generate file sequencing for a particular type of request by setting the output directory and file name at the managed connection level. When the adapter receives a request to create a file, it checks the file sequence log to see whether a file with that name exists. If one does, the adapter uses the file sequence number to create a file name.

Note: The directory path and file name specified in the business object take precedence over the managed connection property values.

In a clustered environment, an environment in which you have one instance of the adapter running on several systems, the sequence file specified by the FileSequenceLog property must be on a mapped drive that is accessible by all the nodes in the cluster. The adapter must have write permission for the sequence log file, or an IOException error is returned.

Note: In the Windows operating systems, such as, Windows 7, Windows Vista, and Windows Server 2008, there are issues faced in the mapped drive connection. Due to this issue, in a clustered environment, where the nodes are running on different machines, the files in the mapped event directory might not be processed

correctly during outbound operations. For more information about working with mapped drives, refer to articles on mapped drive connection to network sharing, for your operating system.

If the FileSequenceLog property is specified and the GenerateUniqueFile property is enabled, the GenerateUniqueFile property takes precedence over the FileSequenceLog property. The sequence number will continue to increment after an adapter restart. If the sequence file is deleted manually, the sequencing starts again from 1.

Delete operation:

The Delete operation deletes a specified file.

Delete

Optionally, you can select to return the output of the delete operation to a component in a business object. If you select the **Enable response type for the operation** check box in the external service wizard, the adapter returns true if the file is deleted successfully. If the file permission for delete is not present, the adapter returns false.

If the file does not exist, the adapter generates a `RecordNotFoundException` error.

Note: In z/OS® environment, the WebSphere Adapter for Flat Files also deletes a file configured with the `No Delete` permission, if the file is marked for deletion during the delete operation.

Exists operation:

The Exists operation checks to see whether a specified file exists.

Exists

If the file that is indicated exists in the specified directory or any of the sub folders, a successful response is returned to the component in the form of a business object. The business object has one attribute, which is set to true if the file exists or false if the file does not exist. If the file does not exist, or if the directory does not exist, the adapter returns `false`.

List operation:

The List operation lists the file names in the specified directory.

List

If the directory does not exist, the adapter generates a `RecordNotFoundException` error.

Overwrite operation:

The Overwrite operation overwrites the specified file with the content specified in the request.

If you select the **Enable response type for the operation** check box in the external service wizard, the file name is returned to the component in a business object. If a

staging directory is specified in the StagingDirectory property, the file that is to be overwritten is copied from the output directory to the staging directory, and the content is overwritten for that file in the staging directory. The file is then moved back to the output directory. If a staging directory is not specified, the content is overwritten on the file in the output directory.

Note: You can configure a staging directory only if the file content is to be written before the Overwrite operation returns the resultant values. You cannot use a staging directory if the Overwrite operation returns an output stream and the component writes to this stream.

When the input request is received as a FlatFileOutputStreamRecord record, the adapter returns an output stream.

If the CreateIfFileNotExists property is set to `true`, the adapter creates a file. The GenerateUniqueFile property has been deprecated. Although you can currently set this property, but the adapter always treats the value for this property as `false`.

If the file to be updated does not exist and the CreateFileIfNotExists property is set to `false`, the adapter generates a RecordNotFoundException error.

Note: For a wrapper business object, if the value is not specified for the CreateFileIfNotExists property on the wrapper, the adapter uses the value specified in its interaction specification property.

Retrieve operation:

The Retrieve operation retrieves the content of the specified file and returns it in the form of a business object. During outbound processing, you can also delete and archive the file returned during the retrieve operation.

During a retrieve operation, the adapter retrieves the content of the file specified in the Retrieve request and returns it in the form of a generic or a content-specific business object. The adapter uses the file splitting feature to divide a large file into smaller chunks, which are then retrieved separately. The file content is split according to the SplittingFunctionClassName and SplitCriteria properties defined in the interaction specification. These properties contain the outbound connection properties the adapter uses to interface with the file system. If you configure a data handler, the adapter returns a content-specific business object; otherwise, it returns a generic business object.

To delete the original file after it has been retrieved, set the DeleteOnRetrieve property in the interaction specification. To archive the file before it is deleted, set the ArchiveDirectoryForDeleteOnRetrieve property.

During the retrieve operation, if the file that is specified in the Retrieve request does not exist, the adapter generates a RecordNotFoundException error.

Note: For a wrapper business object, if the value is not specified for the DeleteOnRetrieve property on the wrapper, the adapter uses the value specified in its interaction specification property.

Outbound data transformation

During outbound processing, the adapter performs data transformation based on the adapter-specific data binding and data handler you select when you configure the adapter for outbound processing in the external service wizard.

Outbound processing with data transformation

During outbound processing, the adapter transforms business objects to the data format expected by the application. The process is controlled by an adapter-specific data binding and data handler that you select when you configure the module for outbound processing.

Figure 3 illustrates the way data is transformed during outbound processing.

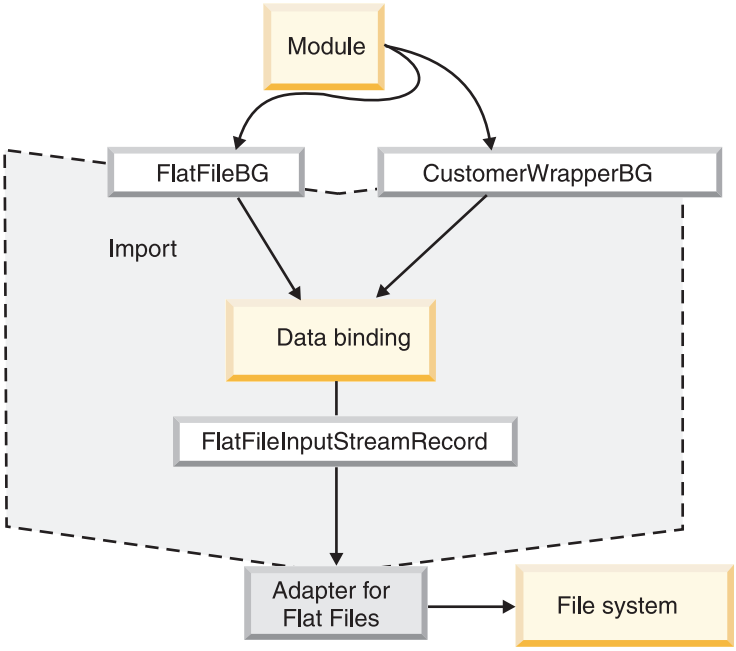


Figure 3. Data transformation during outbound processing

The following steps describe outbound processing with data transformation.

1. For all operations except Retrieve, the adapter performs data transformation based on the input data type and the configured data handler. If the input type is not a generic type (FlatFile or FlatFileBG), the adapter transforms the data. For the Retrieve operation, the adapter transforms the data only if the data handler property of the data binding is configured.
2. The configured data binding is invoked to process the business object.
3. The data binding checks the value specified for the data handler property in the data binding properties, and invokes a content-specific data handler based on the value set for the data handler property.
4. The adapter performs the requested operation on the file and can return a response business object:
   - For the Create, Append, and Overwrite operations, if output is configured, the response business object contains the file name.
   - For the List operation, the response business object contains a list of files in the specified directory.
   - For the Exists operation, the response business object contains a value of either `true` or `false`.

- For the Retrieve operation, the content of the retrieved file is returned in the form of a generic or content-specific response business object.
- For the Delete operation, if output is configured, the response business object contains a value of either `true` or `false`

Outbound processing without data transformation

For all operations except Retrieve, if the input data type is a generic type (FlatFile or FlatFileBG), the adapter performs outbound processing without data transformation. For Retrieve operations, if no value is set for the data handler property of the data binding, no data transformation takes place. During this type of processing, a special data structure, UnstructuredContent, is used to hold the content.

Figure 4 illustrates outbound processing without data transformation.

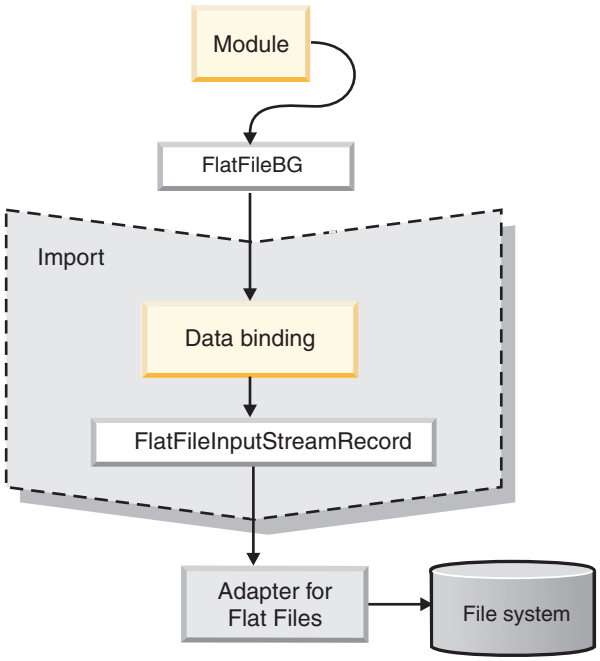


Figure 4. Outbound processing without data transformation

The following steps describe outbound processing without data transformation.

1. For all operations except Retrieve, the adapter checks the input type of the request data object. If the input type is a generic type (FlatFile or FlatFileBG), the adapter does not perform any data transformation on the incoming object. For the Retrieve operation, the adapter checks for the data handler property. If no value is specified, it does transform the data.
2. The configured data binding is invoked to process the business object.
3. For the Retrieve operation, the adapter checks the data handler property. If no value is set for the data handler, the adapter does not transform the data.
4. The adapter performs the requested operation on the file and can return a response business object as follows:
   - For the Create, Append, and Overwrite operations, if output is configured, the response business object contains the file name.

- For the List operation, the response business object contains a list of files in the specified directory.
- For the Exists operation, the response business object contains a value of either `true` or `false`.
- For the Retrieve operation, the content of the retrieved file is returned in the form of a generic or content-specific response business object.
- For the Delete operation, if output is configured, the response business object contains a value of either `true` or `false`.

File splitting

To support files that contain multiple records, the adapter provides an optional file splitting feature. When you use this feature during the Retrieve operation, the adapter divides large files into smaller chunks, which are then retrieved separately.

Depending upon the type of content contained in the file, the file can be split by delimiter or by size.

- When the content of the business object has a definite structure, for example, if it contains elements such as name, address and city, the file is split by delimiter.
- When the business object contains unstructured data, such as plain text or binary files, the file is split by size.

By default, the adapter splits files by size.

The value specified in the SplitCriteria property determines the method that is used. The default value for SplitCriteria property is zero, which means that no splitting is performed. You can also leave the values of the SplitCriteria and SplittingFunctionClassName properties empty if no splitting is required.

You can optionally provide a custom file splitter class. Set the SplittingFunctionClassName property to the name of the class.

File splitting by delimiter

When one or more characters such as a comma (`,`), semicolon (`;`), quotation mark (`",` `'`), brace ({}), or slash (`/` `\`) (delimiters) are used to separate the business objects in a file, the adapter can split the file into smaller chunks based on the delimiter. You define the delimiter that separates the business objects in the file in the SplitCriteria property.

You can enable file splitting by delimiter by specifying the value of the SplittingFunctionClassName property as com.ibm.j2ca.utils.filesplit.SplitByDelimiter.

The following rules apply to the use of delimiters:

- All new lines in the delimiter are represented by platform-specific newline characters. The platform-specific newline characters are shown in Table 1.

Table 1.

Platform	Newline character
Macintosh	\r
Microsoft Windows	\r\n
UNIX	\n

- If there is more than one delimiter, each delimiter must be separated by a semicolon (;). The delimiters are matched in the order in which they are given. If the semicolon is part of the delimiter, it must be escaped as `\;`. For example, if the delimiter is `##\;##`, it is processed as `##;##`.
- 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 `##;;$$`, the adapter considers `##$$` as the delimiter and skips `content skipped by the adapter`:

  ```
  Name=Smith
  Company=IBM
  ##content skipped by the adapter$$
  ```

- The delimiter can have any value, and there are no restrictions on it. The delimiter is a combination of a valid string, the newline character (for example, `\n`), and a semicolon separator if there is more than one delimiter. A delimiter does not have to comprise the newline character and a semicolon. The newline character is used only when a newline is to be considered when splitting the contents of the file. Examples of valid delimiters include:
  - `####;\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.

Example of a common scenario and the recommended delimiter format:

Table 2.

Data binding	BO content	Recommended delimiter format

XML	`<?xml version="1.0" encoding="UTF-8"?>`
	`<customer:Customer xsi:type="customer:Customer"`
	`xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"`
	`xmlns:customer="http://www.ibm.com/xmlns/prod/websphere/`
	`j2ca/flatfile/customer">`
	`<CustomerName>Deepa</CustomerName>`
	`<Address>IBM</Address>`
	`<City>Bangalore</City>`
	`<State>KA</State>`
	`</customer:Customer>`
		`\n`

File splitting by size

The value specified in the SplittingFunctionClassName property determines whether a file is split by size. If the SplittingFunctionClassName property is set to `com.ibm.j2ca.utils.filesplit.SplitBySize`, the SplitCriteria property must contain a valid number that represents the maximum file size, in bytes. If the file is

larger than the value specified in the SplitCriteria property, the file is split into chunks and each chunk is posted to the import separately. If the file is smaller than the SplitCriteria value, the entire file is posted to the import.

When event files are split into chunks, each chunk becomes a business object. This means that the value specified for the PollQuantity property and the number of business objects delivered to the import can be different. Although the adapter polls according to the PollQuantity value, it actually processes the number of business objects in the file one at a time. For example, if an event file is chunked into three parts, one file is polled and the three business objects are delivered to the import (because each chunk creates a single business object).

At the import, the adapter does not reassemble the chunked data into a single file, but it provides information about the chunks to enable IBM Business Process Manager or WebSphere Enterprise Service Bus to reassemble them into a single file. The chunk information is included in the ChunkFileName property of the FlatFileInputStreamRecord record, and includes the chunk size in bytes and the event ID. The event ID of a chunk uses the following form: `eventFileLocation_/_timestampStr_/_MofN`, where `M` is the current chunk number and `N` is the total number of chunks. An event ID would look like the following example:

`C:\flatfile\eventdir\eventfile.in_/_2005_01_10_10_17_49_864_/_3of5`, where timestampStr has the following format:
`year_month_day_hour_minutes_seconds_milliseconds`.

Generating unique file names

To generate unique file names during Create operations, add a persistent sequence number to the default file name or use random numbers to generate file names.

There are two ways to generate unique file names during Create operations:

1. Add a persistent sequence number to the default file name. This method is recommended, especially in a clustered environment.
2. Use random numbers to generate unique file names without any persistence.

Generating unique file names using a persistent sequence number

To generate unique file names using a persistent sequence number, specify:

- The sequence file, which is the complete path of the file where the sequence numbers are stored
- 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.

Note: By default, the sequence number starts from one and continues until the number **9223372036854775807**. When this number is reached, the sequence number is reset and the numbering starts automatically from one.

The properties that control the generation of unique file names are present 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 specify properties for individual business objects, use the properties in the managed connection factory to control the generation of file names.

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` in the managed connection factory, the output file names created are `Customer.1.txt`, `Customer.2.txt`, and so on.

For each request, the adapter increments the number in the sequence file, and the input type takes the sequence number that is currently stored in the sequence file. Sequence numbers are not maintained separately for different input data types.

For compatibility with sequence files generated with previous versions of the adapter, where sequence numbers were maintained separately for different input data types, the adapter checks for all entries in the file that have the older format (`<dirPath>/Customer.txt = 2`, where `Customer.txt` is the default file name and `2` is the sequence number to be used when the adapter receives another Create request on the same file). The adapter searches for all such sequence numbers for each input type and uses the highest sequence number as the sequence number for the next input type. The adapter then overwrites the entire file with the new (increments) sequence number.

Important: Unless they are part of a cluster, two adapter instances must not access the same sequence file, because it might result in delayed processing of batch requests.

Generating unique file names using random numbers

To generate unique file names using random numbers, set the GenerateUniqueFile property in the managed connection factory, interaction specification, or in the wrapper business object to `true`. When generating unique file names, you can specify a prefix and suffix (file extension) for the file name. This setting enables the adapter to generate file names that are unique but with a predefined prefix and suffix to its file name. For instance, if you specify the prefix as `ffaa` and the file extension as `.abc`, the adapter generates unique file names with the following format: `ffaa[`*`RandomNumber`*`].abc`, where *RandomNumber* is the random number that the adapter has generated. For example, `ffaa23423.abc`.

Inbound processing

WebSphere Adapter for Flat Files supports inbound event processing. It polls the local file system at specified intervals for events, such as the creation of a file. When it detects an event, it converts the event data into a business object and sends it to the module for processing.

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

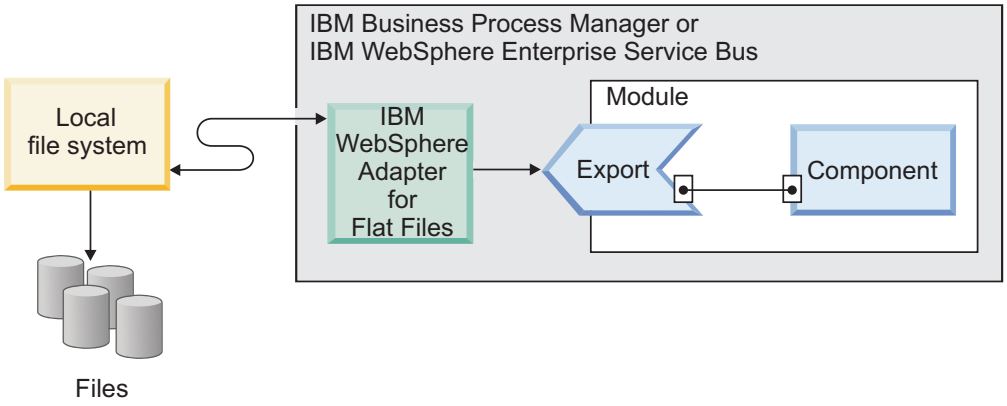


Figure 5. Inbound processing

When a change occurs in the local file system, an event file, which is a new or changed file, is created in a specific directory. You configure this directory as the event directory for the adapter. Although an event file can represent one or more events in the file system, it forms a single unit of transfer to the adapter.

The adapter polls the event directory on the file system at regular intervals, based on the value set in the PollPeriod property. When a file arrives in the event directory, the adapter sends the content of the file to the export. The file content can be sent in its entirety or split into several business objects, or chunks. The adapter sends the business objects to the export by using a function selector, which selects an operation to invoke on the component and provides the correct data binding.

The inbound processing flow is described in the following steps.

1. Events, in the form of files, are generated in the file system.
2. The adapter polls the event directory.
3. The adapter assigns each event an event ID and stores the event ID in the event store. The event store is a persistent cache where event records are saved until a polling adapter can process them. You must create this database before you configure the adapter. The default name of the database is FFDB.
4. The adapter reads each event file as bytes. If file splitting is enabled, the adapter parses the event file based on the values set in the SplittingFunctionClassName and SplitCriteria properties:
   - If splitting is based on a delimiter, the class that performs this functionality and the split criteria are provided.
   - If splitting is based on file size, the class name that performs this functionality is provided.
5. If the configured data type is object-specific, for example, CustomerWrapper, the data handler is configured on the DataBinding, and the adapter transforms the data. If the configured data type is FlatFile or FlatFileBG, the adapter passes the content of the file as a byte array within a FlatFile business object, and no transformation is performed.

   Note: If file splitting is enabled, the business object contains the file size and the event ID.
6. The adapter sends the business object to the export through a function selector, which selects an operation to invoke on the component and provides the

correct data binding. After the business object has reached the export, the event is deleted from the event store. If archiving is enabled, the event is moved to an archiving table before it is deleted.

Polling files in subdirectories

By default, when the adapter polls files in the event directory, it polls files from the root directory only and ignores files in the subdirectories. It you set the PollSubDirectories property in the activation specification to `True`, the adapter first polls the files in the root directory and then polls the files in the subdirectories. After the adapter has retrieved all the files, it sorts them according to the value set for the SortEventFiles property. The adapter then processes the files according to the value set for the PollQuantity property and sends the business objects to the downstream components.

Event archiving

To track successfully polled events, you can configure an archive directory on the file system by using the ArchiveDirectory activation specification property in the external service wizard. The files are copied to the archive directory with either a `success` or `fail` extensions, as specified in the activation specification.

Event file locking

File locking behavior is operating system dependent. On Windows, if the adapter polls any file from the event directory that another application is using and the file is 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 using the following metadata:

- FileName
- File Size
- Directory
- Last Modified

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

The rule is applied to the files that are filtered by the event file mask criteria. By default, event file mask will have "*.*" as default value.

Rule-based filtering does not support the logical "OR" operator values between multiple rules.

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

Table 3. Metadata filtering properties

Property	Valid operators	Value	Prerequisites
FileName	Matches_File_Pattern	For example: *.txt	Nil
	Matches_RegExp	Java Regular Expression	
FileSize	Greater than, Less than, Greater than or equal to, Less than or equal to, Equal to, Not equal to.	Numeric value in Bytes. For example: 10000	Nil
Directory	Matches_RegExp	Java regular expression	pollSubDirs = true
LastModified	Greater than, Less than, Greater than or equal to, Less than or equal to, Equal to, Not equal to. **Note:** Select the 'Equal to' operator when you choose the days of a week.	Day of the week or Time. For example : MONDAY or 20:41:10	Nil
END-OF-RULE	END-OF-RULE	END-OF-RULE	Nil

Event persistence

The adapter supports event persistence for inbound processing in case of abrupt termination. Event persistence (or assured-once delivery) is a way to ensure that events are delivered only once to the export in the case of a failure. During event processing, the adapter maintains the event state in an event store located on the data source. You must set up the data source using IBM Business Process Manager or WebSphere Enterprise Service Bus before you create the event store. To use the recovery feature provided by IBM Business Process Manager or WebSphere Enterprise Service Bus, set the AssuredOnceDelivery property in the activation specification to `True`. This recovery feature is enabled by default.

The adapter also provides for event persistence by using an in-memory representation of the event store. When you use this feature, you do not need to create a JNDI data source or an external event store, and event processing is faster. However, with this feature there is no support for event recovery. If there is a server failure, the in-memory event stores are lost. To prevent the loss of events in the case of server failure, the recommended approach is to use the database event store.

To use the in-memory event persistence capability of the adapter, you must set the AssuredOnceDelivery property to `false`, or the adapter will log a warning message.

Related reference

Chapter 3, “Samples and tutorials,” on page 57
To help you use WebSphere Adapters, samples and tutorials are available from the Business Process Management Samples and Tutorials website.

“Activation specification properties” on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

Event store

The event store is a persistent cache where event records are saved until the polling adapter can process them. The adapter uses event stores to track inbound events in the system. Each time a file is created, updated, or deleted, the adapter updates the status of the event in an event store. The status of each event is continuously updated by the adapter for recovery purposes until the events are delivered to the export.

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

When the adapter polls the local file system, it creates an entry in the event store 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.

If an event is successfully posted, event store entries are deleted. For failed events, the entries remain in the event store. Optionally, the adapter can archive successfully polled event files in an archive directory. The adapter supports processing of event files without any restriction on the file size.

Note: Failed events can result from incorrect data in the event file. For example, a field named `fname` might appear as `fnam`. The only way to correct the situation is to send the event file again with the correct data.

The adapter provides assured-once event delivery. This means that each event is delivered once and only once. If you set the AssuredOnceDelivery activation specification property to `True`, the adapter stores an XID (transaction ID) value for each event in the event store. When an event is obtained for processing:

1. The XID value for the event is updated in the event store.
2. The event is delivered to its corresponding export.
3. The event is deleted from the event store.

The following figure illustrates the event management flow for the adapter.

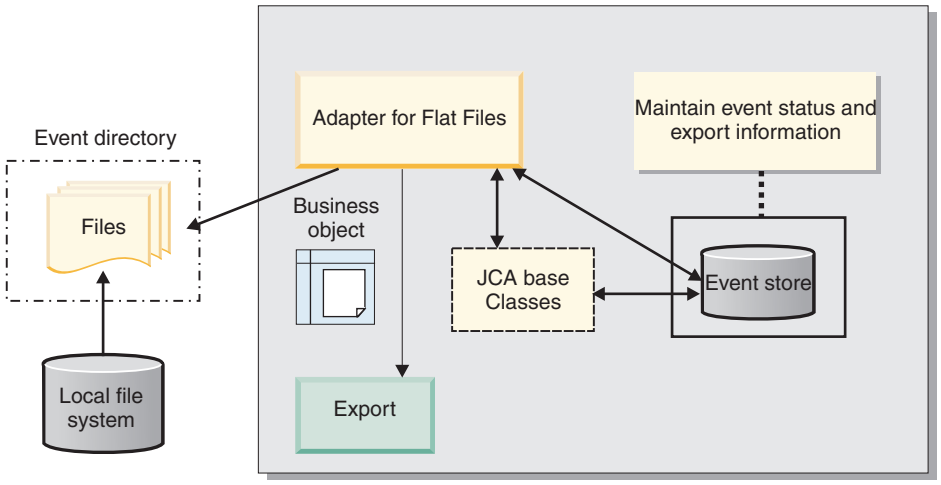


Figure 6. Event management flow

Event store structure:

The event store is used by the adapter to track events.

The following event table notes the values that are stored for each event.

Table 4. Event store structure

Column name	Type (length)	Description
EVNTID	Varchar (255)	Used to track events during inbound processing. Each event requires an event ID for tracking purposes. This event ID must be a unique identifier in the table.

Table 4. Event store structure (continued)

Column name	Type (length)	Description
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: **NEW (0)** The event is ready to be processed. **FETCHED (3)** The adapter picked up the event for processing. **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 to track failed events so that they are not processed again during recoveries. Failed events are marked "ARCHIVED."
BOSRTPOS	Long	Used to store the starting file pointer position of the business object associated with this event.
BOENDPOS	Long	Used to store the ending file pointer position of the business object associated with this event.
TIMESTMP	Timestamp	Used to represent the time when the event was picked up for processing.

Event archival values:

You can configure the adapter to archive processed event files in a directory, which you can then access to obtain a list of processed events. The file extension reflects whether an archived event was successful or not.

All archived events in the specified archive directory are stored with success, failure, and original file extensions. Success is used when the event processing is

successful. If the event processing fails, the file is archived with failure and original extensions. If the event file has multiple business objects and some of them are successful, there is also a file with a success extension.

The archive extensions are configurable based on the following activation specification properties: FailedArchiveExt, OriginalArchiveExt, and SuccessArchiveExt.

The following table lists the archive extensions used by the adapter.

Table 5. Event archive values

Extension	Definition	Format
SUCCESS	The event file was delivered to the export.	<filename>_<timestamp>.SUCCESS
FAIL	The event file was not delivered to the export.	<filename>_<timestamp>.FAIL
ORIGINAL	The original event file that was picked up for processing.	<filename>_<timestamp>.ORIGINAL

File store

The file store is a persistent cache where event files are saved until the polling adapter can process them. The adapter uses file store to track inbound files in the system. Each time a file is created, updated, or deleted the adapter updates the status of the file in the file store.

The file store mechanism provides the following advantages for the adapter:

- When multiple adapters are running in a clustered environment, the file store enables sharing of file processing work. Also, it avoids duplicate file processing by multiple adapter instances.
- The adapter can process large files (files of any size).

When the adapter polls the local file system, it creates an entry in the file store for each event file that matches the search criteria specified in the activation specification properties. The adapter records the status of each new entry as `NEW` in the event table.

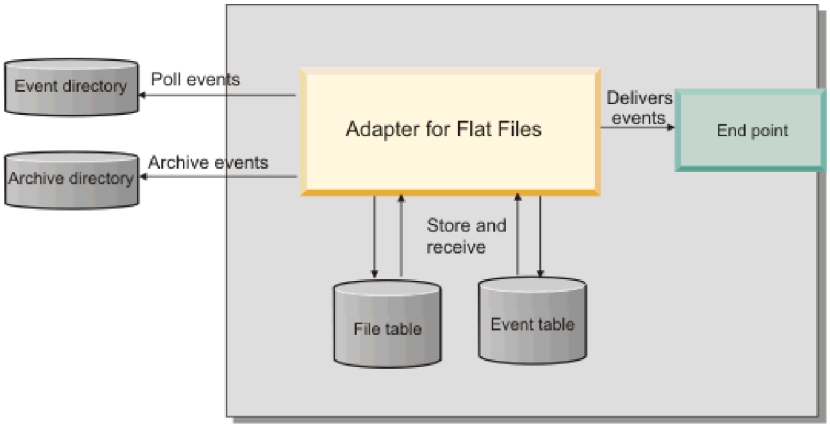


Figure 7. File management flow

File table structure:

The adapter uses the file table to read, track, and record the details of the event files that are being polled from the event directory.

File table

The file table contains the entries for the files to be polled by the adapter. The entries in the table support the adapter to read only the file content required by the polling quantity. In addition, the last position of the file pointer after the partial read is recorded in this table.

The following table describes each file table column.

Table 6. File table structure

Column name	Type	Description
FILENAME	Varchar (255)	Name of the event file to be processed.
FILESTAT	Integer	Status of the file entry. The adapter uses the status to determine whether the file is a new event to be processed or if the event is being currently processed. **UNPROCESSED (0)** The new file is ready to be processed. WebSphere Adapter for Flat Files polls the event directory for files and creates an entry in the file table. **IN-PROCESS (1)** A file is in-process if the adapter is reading the file content. When the file status is 1, no other adapter is allowed to process the file. The timestamp is updated when the file is picked up for processing. **EVENTS UPDATED (2)** The adapter reads only the file content required by the polling quantity and generates the new events for the current set of business objects. **PROCESSED (3)** The file processing is complete and the event entries are generated in the event table for the business objects. **FAILED (4)** The adapter was unable to read the file because of an unexpected error. The file might be corrupt or invalid. **ARCHIVING (5)** The archiving process for this file is in progress. **EOF REACHED (6)** Shows the end of file status, when the adapter is enabled to poll for the modified file content.
LBOCOUNT	Long	Specifies the number of business objects that were processed until the file was previously read.
LREADPOS	Long	Specifies the end position of the file pointer after when the file was previously read.
TIMESTAMP	Timestamp	Indicates the time when the file was picked up for processing.
LMDFTIME	Timestamp	Indicates the last modified time of the file.

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 three function selectors, `FilenameFunctionSelector`, `EmbeddedNameFunctionSelector`, and `RootNameFunctionSelector`.

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 7. Examples of matching rules for FilenameFunctionSelector

FileName	ObjectName	Rule
`Customer0001.txt`	`Customer`	`CUST.*TXT`
`2231ORZ93.z21`	`Order`	`[0-9]*OR[A-Z][0-9]{2}.*`
`2231ORZ93.z21`	`Order`	`*OR.*`

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 can occur any number of times.

To generate the native function name, the function selector prefixes `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 must be the payload object name, for example, Customer or Order, and not the wrapper or business graph name. For pass-through scenarios, use FlatFile 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 emitFlatFile.

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 content data, and 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.

RootNameFunctionSelector

RootNameFunctionSelector is used only for global elements in business objects, where the global element name is the root element name in the event XML file. It returns the function name based on the global element name. For example, if the global element name is CustomerType1, the function returned by the root name function selector is 'emit CustomerType1'.

RootNameFunctionSelector should be used only for global elements with XML Datahandler or UTF8XMLDatahandler.

Note: To use global Elements with Delimited Datahandler or FixedWidth Datahandler, you should use FilenameFunctionSelector instead of RootNameFunctionSelector.

RootNameFunctionSelector does not need any more configuration, as it does not depends on the data handler to get the correct function name.

Related reference

"Connection properties for the wizard" on page 188
Connection properties are used to build a service description and save the built-in artifacts. These properties are configured in the external service wizard.

File retrieval

During inbound processing, you can manage the retrieval of the files by using the Poll event files for modified content or the Time interval for polling unchanged files property. You can also use the Include only the newly appended content property to retrieve only the appended file contents.

The Poll event files for modified content property and the Time interval for polling unchanged files property are mutually exclusive properties.

File retrieval based on last recorded time stamp

The Poll event files for modified content property enables the adapter to poll for event files repeatedly (when the file content has changed) in the event directory during the subsequent poll cycles after the previous event poll. The adapter then retrieves the event files and delivers them to the endpoint in the next poll cycle.

When you configure this property, the adapter retrieves the new files added to the event directory since the last poll cycle along with the existing modified files.

This property enables the adapter to monitor file changes based on the last modified time stamp of each file. When the adapter is started for the first time, all event files are polled and processed from the event directory. The adapter does not delete any polled event file from the event directory after event processing.

During the next poll cycles, only those event files are picked for polling whose lastModifiedTimeStamp values have changed. If the lastModifiedTimeStamp for a file is same, it means that the file has not changed and therefore it is not picked up for polling. This property allows for polling the complete content of the event file

every time the file content has been changed. For more information, see the "Poll event files for modified content" on page 202 property details.

You can also configure the adapter to deliver the newly appended file contents at the end of the file by using the `Include only the newly appended content` property. This property is enabled when you select the `Poll event files for modified content` property in the external service wizard.

If there is a change in the last modified time stamp value, during the next poll cycle, the adapter checks the event file for any change in the file content. The changes to the file contents that are considered by the adapter for polling again are in the form of appended business objects. If the appended business objects exist, the adapter retrieves only the appended file contents by comparing the file with the file contents of the previous poll. The adapter compares by using the total number of business objects in the previously polled contents and the contents in the current poll. It does not process any business object if the counts of the business objects are same or less than the last poll.

The following scenarios illustrate how the adapter decides if a business object is to be delivered to the endpoint. In this example, three business objects are taken as a sample count. The scenarios depict how the adapter processes the business objects based upon their new processing order in the event file.

- If another business object is added after the three business objects, the adapter delivers the fourth business object to the endpoint.
- If the second business object is deleted, and two more business objects are added at the end, the adapter delivers only the last business object to the endpoint. In the changed position, the third business object is not delivered although it is a new business object.
- If the second business object is deleted, and no new business objects are added, the adapter does not poll the event file for delivery to the endpoint. If two more business objects are added at the end, the adapter delivers both the business objects to the endpoint.
- If one business object is added in between the second and the third business object, then the existing third business object is delivered again to the endpoint.
- If two business objects are deleted and two new business objects are added, then the adapter does not deliver any business object to the endpoint.
- If the second business object is deleted and two new business objects are added in its place, the existing third business object becomes last in the row. This configuration increases the count of the business objects and the adapter delivers the earlier existing third business object to the endpoint.

Note: When the server is restarted after a shutdown, the adapter polls all the contents of the files modified during this time to the endpoint that also include the appended contents.

For more information, see the "Include only the newly appended content" on page 199 property details.

Note: If you select the `Poll event files for modified content` property, then you cannot configure the `Time interval for polling unchanged files` and `File pass by reference` properties.

File retrieval based on time interval

The `Time interval for polling unchanged files` property monitors the changes to files in the event directory for the specified time interval. When you configure this property, the adapter polls the files for event processing that have not undergone any change during the time interval. The adapter also polls files that are currently being edited but have not been saved during the specified time interval. The unsaved contents are not processed during the event processing. This configuration prevents occurrence of any erroneous results.

When the adapter polls the directory, it uses this property to check if a file has been modified by any event during the specified time interval. The adapter uses the lastModifiedStamp values of the files to determine if a file has changed during the time interval.

The adapter retrieves the unchanged files in their present state and the changed files from their last saved state. For more information, see the Time interval for polling unchanged files property details.

Note: If you select the `Time interval for polling unchanged files` property, then you cannot configure the `Poll event files for modified content` property.

Related tasks

"Setting deployment and runtime properties" on page 97
After you have decided whether your module is to be used for outbound or inbound communication with the enterprise information system (local file system), you must configure the activation specification properties, which hold the inbound event processing configuration information for the export.

Related reference

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

File splitting

The adapter supports an optional file splitting feature to reduce memory loading during the event processing. When this feature is used, the adapter divides large event files into smaller chunks, which are then posted separately to the endpoint.

The adapter splits large event files into several business objects, also called chunks, based on the value you specify in the SplitCriteria property, which can be either a delimiter or a chunk size. Each business object is delivered to the endpoint separately. You can split files using a delimiter when the content of the business object has a definite structure; for example, if you have a customer business object with elements such as name, address, and city. You can also split files by size when the business object contains unstructured data, such as plain text or binary files.

When event files are split into such chunks, each chunk creates a business object. This means that the value specified for the PollQuantity property and the number of business objects delivered to the endpoint can be different. When file splitting based on a delimiter is enabled, the PollQuantity activation specification property specifies the number of such event files that are present in the event store, and the class used to split the event file is set in the SplittingFunctionClassName activation specification property.

The adapter does not reassemble the chunked data.

The value specified in the SplitCriteria property determines the method that is used. The default value for SplitCriteria property is zero, which means that no splitting is performed. You can also leave the values of the SplitCriteria and SplittingFunctionClassName properties empty, if no splitting is required.

You can optionally provide a custom file splitter class. Set the SplittingFunctionClassName property to the name of the class.

File splitting by delimiter

When one or more characters such as a comma (`,`), semicolon (`;`), quote (`",'`), brace (`{}`) or slash (`/ \`) delimiters are used to separate the business objects in a file, the adapter can split the file into smaller chunks based on the delimiter. Each chunk is a logical unit that is used to construct a business object when forwarded to IBM Business Process Manager or WebSphere Enterprise Service Bus. You define the delimiter that separates the business objects in the file in the SplitCriteria property.

To demonstrate how the PollQuantity value works with delimiter file splitting, consider two event files. The first event file contains a business object and the second event file contains two business objects. If the PollQuantity value is 2, the first business object from the first event file and the next business record from the second event file are sent in the first poll cycle. The second business object from the second file is sent in the second poll cycle.

The following rules apply to the use of delimiters:

- All new lines in the delimiter are represented by platform-specific newline characters. The platform-specific newline characters are shown in Table 1.

Table 8. Platform-specific newline characters

Platform	Newline character
Macintosh	\r
Microsoft Windows	\r\n
UNIX	\n

- If there is more than one delimiter, each delimiter must be separated by a semicolon (`;`). The delimiters are matched in the order in which they are given. If the semicolon is part of the delimiter, it must be escaped as `\;`. For example, if the delimiter is `##\;##`, it is processed as `##;##`.
- 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 `##;;$$`, the adapter considers `##$$` to be the delimiter and skips the words "content skipped by the adapter", as shown here:

```
Name=Smith
Company=IBM
##content skipped by the adapter$$
```

- The delimiter can have any value, and there are no restrictions on it. The delimiter is a combination of a valid string, the newline character (for example, \n), and a semicolon separator if there is more than one delimiter. A delimiter does not have to comprise the newline character and a semicolon. The newline character is used only when a newline is to be considered when splitting the contents of the file. Examples of valid delimiters include:
  - `####;\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.

An example of a scenario with the commonly used delimiter format is shown in Table 2.

Table 9. Using the delimiter format

Data binding	BO content	Recommended delimiter format

XML	`<?xml version="1.0" encoding="UTF-8"?>`
	`<customer:Customer xsi:type="customer:Customer"`
	`xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"`
	`xmlns:customer="http://www.ibm.com/xmlns/prod/websphere/`
	`j2ca/flatfile/customer">`
	`<CustomerName>Deepa</CustomerName>`
	`<Address>IBM</Address>`
	`<City>Bangalore</City>`
	`<State>KA</State>`
	`</customer:Customer>`
	`##`
		`##;\n`

File splitting by size

The value specified in the SplittingFunctionClassName property determines whether a file is split by size. If the SplittingFunctionClassName property is set to `com.ibm.j2ca.utils.filesplit.SplitBySize`, the SplitCriteria property must contain a valid number that represents the maximum file size, in bytes. If the event file is larger than the value specified in the SplitCriteria property, the file is split into chunks and each chunk is posted to the endpoint separately. If the event file is smaller than the SplitCriteria value, the entire event file is posted to the endpoint.

When event files are split into chunks, each chunk becomes a business object. This means that the value specified for the PollQuantity property and the number of business objects delivered to the endpoint can be different. Although the adapter polls according to the PollQuantity value, it actually processes the business objects in the file one at a time. For example, if an event file is chunked into three parts, one file is polled and the three business objects are delivered to the endpoint (because each chunk creates a single business object).

If you use the FileChangeNotification property, then the size of the event file must be a multiple of the split chunks. For example, for an event file that contains 90 bytes, the split size can either be 15, 6, 3, or 2.

When the event file is not a multiple of the split chunks and the last business object is smaller than the split size, the adapter delivers the last business object to

the endpoint correctly during the first event poll. When new contents are appended to the event file and the FileChangeNotification property is specified as `True`, then the updated business object that was smaller than the split size, does not send any new content to the endpoint. The sample scenarios for this configuration, when a content is split by 2 bytes, are described in the following example.

When the content "ABCDE" is split by 2 bytes, so that the last business object contains only "E", then the adapter delivers the contents "AB", "CD", and "E" to the endpoint during the first event poll. In the next event poll, if the content is changed to:

- "ABCDEF", the content is split to "AB", "CD", and "EF", and the adapter delivers the contents "AB", "CD", and "E" to the endpoint.
- "ABCDEFG", the content is split to "AB", "CD", "EF", and "G", and the adapter delivers the contents "AB", "CD", "E", and "G" to the endpoint.

Note: When an event file has failed business objects and file splitting by size is enabled, then the event file is archived with the `.fail` extension in the specified archive directory.

At the endpoint, the adapter does not reassemble the chunked data into a single file, but it provides information about the chunks to enable IBM Business Process Manager or WebSphere Enterprise Service Bus to reassemble them into a single file. The chunk information is included in the ChunkFileName property of the FlatFileInputStreamRecord record, and includes the chunk size in bytes and the event ID. The event ID of a chunk uses the following form: `eventFileLocation_/_timestampStr_/_MofN`, where `M` is the current chunk number and `N` is the total number of chunks.

`C:\flatfile\eventdir\eventfile.in_/_2005_01_10_10_17_49_864_/_3of5`, where timestampStr has the following format:
`year_month_day_hour_minutes_seconds_milliseconds`.

Chunk information in WebSphere Adapter for Flat Files, version 7.5

With WebSphere Adapter for Flat Files, version 7.5, the event ID does not contain the total business object count. Therefore, by default the total business object count is no longer part of the chunk information being sent to the endpoint. The format of the event ID is changed to:
`EventID=AbsolutePathOfEventFileNameInLocalEventDirectory_/_YYYY_MM_DD_HH_mm_ss_SSS_/_currentBONumber`, where the `YYYY_MM_DD_HH_mm_ss_SSS` string represents `year_month_day_hour_minutes_seconds_milliseconds`.

Optionally, you can include the total business object count in the chunk information by using the includeBOCountInChunkInfo property. When you enable the includeBOCountInChunkInfo property, the total business object count is included in the chunk information being sent to the endpoint.

Following is the format of the chunk information, when you enable the includeBOCountInChunkInfo property:

```
AbsolutePathOfEventFileNameInLocalEventDirectory_/_YYYY_MM_DD_HH_mm_ss_SSS_/_currentBONumberofTotalBO
```

For example, the chunk information can be: `C:\flatfile\eventdir\c5.txt_/_2010_11_17_14_35_34_509_/_4of5`

Following is the format of the chunk information, when you disable the includeBOCountInChunkInfo property:

```
AbsolutePathOfEventFileNameInLocalEventDirectory_/_YYYY_MM_DD_HH_mm_ss_SSS_/_currentBONumber
```

For example, the chunk information can be: `C:\flatfile\eventdir\c5.txt_/_2010_11_17_14_35_34_509_/_4`

For more information, see "Include total business object count in the ChunkInfo (includeBOCountInChunkInfo)" on page 200.

Related reference

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

Inbound data transformation

During inbound processing, the adapter performs data transformation based on the adapter-specific data binding and data handler that you select when you configure the module in the external service wizard.

Inbound processing with data transformation

The process of data transformation during inbound processing is controlled by the adapter-specific data binding and data handler that you select when you configure the module. The following steps describe inbound processing with data transformation.

1. Each individual event is retrieved from the event file based on the value set in the SplitCriteria property. The content is set on the record and sent to the data binding.
2. The adapter checks the expected data type of the inbound operation. If it is not a generic type (FlatFile or FlatFileBG), the adapter checks for the data handler property in the data binding.
3. If the data handler is set, the adapter transforms the data. The data binding invokes the data handler and returns a content-specific business object.
4. The adapter passes this content-specific business object to the endpoint by calling the method returned by the function selector.

Inbound processing without data transformation

If no data transformation is required on the content, for example, when text\xml content must be retained as text\xml content, the event data is not converted into business objects but is passed through as unstructured content.

The following steps describe inbound processing without data transformation.

1. Each individual event is retrieved from the event file based on the value set in the SplitCriteria property. The content is set on the record and sent to the data binding.
2. The data binding checks for the expected type of the event. If it is a generic type (FlatFile or FlatFileBG), the adapter does not transform the data.
3. The data binding sets the content on the UnstructuredContent record and sends it back to the adapter.
4. The adapter passes this business object to the endpoint by calling the method returned by the function selector.

Business objects

A business object is a logical data container that represents the data that is processed by the adapter. The data can represent either a business entity, such as an invoice or an employee record, or unstructured text, such as the body of an e-mail or a word-processing document. The adapter uses business objects to send data to or obtain data from the local file system.

How the adapter uses business objects

During outbound processing, the adapter:

1. Receives a business object from the module that represents a request to perform an operation on a file in the local file system.
2. If necessary, converts the business object into a format that can be understood by the local file system.
3. Performs the requested operation.
4. Returns a business object, if applicable, that represents the result of the operation to the module.

During inbound processing, the adapter:

1. Retrieves a file from the event directory on the local file system.
2. Constructs a business object out of the data, transforming the data, if necessary, into the required format.
3. Sends the business object to the export

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 IBM Integration Designer. If you use the external service wizard, the wizard examines files in the file system and generates business objects to represent them. It also generates other artifacts needed by the adapter.

If you use the business object editor, you create business objects manually. After you create your business objects, you can use the business object editor to define the hierarchy of the business objects.

When you run the external service wizard, the WebSphere Adapter for Flat Files generates two types of business objects: content-specific and generic. The adapter generates these generic business object XSD files:

- FlatFile.xsd
- FlatFileBG.xsd
- UnstructuredContent.xsd
- FileContent.xsd

An example of a content-specific business object is Customer. If you select Customer, these content-specific XSD files are generated, in addition to the generic XSD files:

- Customer.xsd
- CustomerWrapper.xsd
- CustomerWrapperBG.xsd

Note: In this example, the business graph CustomerWrapperBG.xsd is generated. The generation of business graphs is optional.

During adapter configuration, you can optionally choose 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 7.0 and later releases, business graphs are optional; they are required only when you are adding business objects to a module created with a version of IBM Integration Designer earlier than version 7.0. If business graphs exist, they are processed, but the verb is ignored.

Related reference

"Business object information" on page 159
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 local file system. The name typically reflects the operation to be performed and the structure of the business object.

Global elements

Global elements are the globally defined schema elements, which can be reused by referencing them in other parts of the schema or from other schema documents.

WebSphere Adapter for Flat Files supports global elements in structured business objects. The adapter supports global elements of anonymous type and global elements of named type, with namespace as well as without namespace in schema business objects.

For more information see, "Global elements in a structured business object" on page 162.

WebSphere Application Server environment variables

WebSphere Application Server environment variables can be used in the external service wizard to specify directory values.

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 IBM Business Process Manager or WebSphere Enterprise Service Bus administrative console.

With IBM Business Process Manager or WebSphere Enterprise Service Bus Version 7.0, 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 change the environment variable in the IBM Business Process Manager or WebSphere Enterprise Service Bus 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.

When you create 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:\flatfile\event.
- The scope for the environment variable, which 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. The Node is the default scope level.
  - Cell scope limits visibility to all servers on the named cell.

To create WebSphere Application Server environment variables, use the IBM Business Process Manager or WebSphere Enterprise Service Bus administrative console.

Related tasks

"Defining WebSphere Application Server environment variables" on page 62
Use the administrative console of IBM Business Process Manager or WebSphere Enterprise Service Bus to define WebSphere Application Server environment variables.

Related reference

"Managed connection factory properties" on page 172
Managed connection factory properties specify information the adapter needs at run time for outbound communication with the local file system.

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

The external service wizard

Use the external service wizard to configure your adapter before it is deployed to IBM Business Process Manager or WebSphere Enterprise Service Bus. The wizard examines files on the local file system, builds services (based on search criteria you provide), and generates business objects and interfaces.

The external service wizard provides a blueprint for business objects. It allows you to select the artifacts of interest and generates deployable service objects and descriptions. By selecting meta-object nodes from the metadata tree structure, you can generate business objects for EIS or database entities. The metadata is transformed into service data objects consisting of business graphs and business objects.

The following figure illustrates the external service wizard flow. When finished, an EAR file containing all the information for your adapter project is created. This EAR file can then be deployed to the application server.

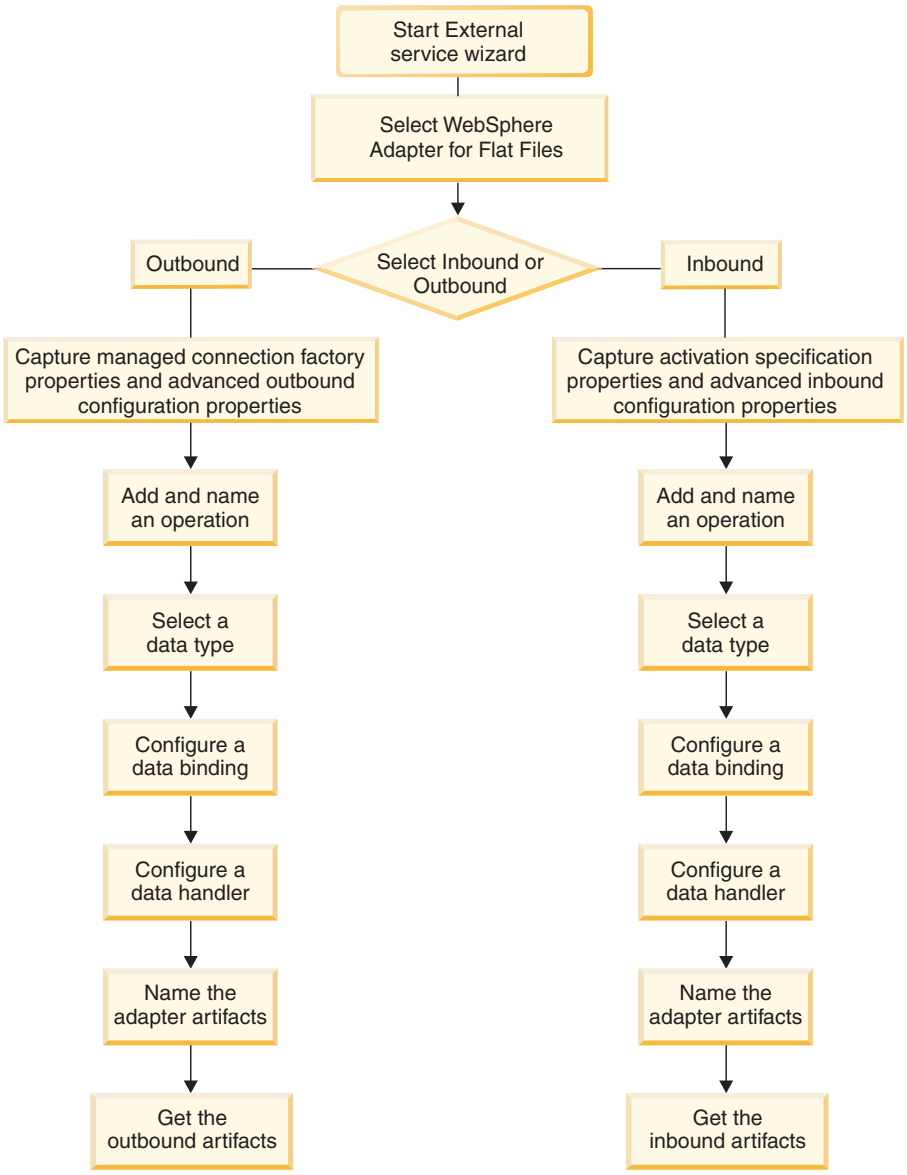


Figure 8. Basic external service wizard flow

Chapter 2. Planning for adapter implementation

To implement WebSphere Adapter for Flat Files, you must plan for inbound and outbound processing and consider security and performance requirements. Also, if you are migrating from an earlier version of WebSphere Adapter for Flat Files, perform any migration tasks.

Before you begin

Before you begin to set up and use the adapter, you must possess a thorough understanding of business integration concepts, the capabilities, and requirements of the integration development tools and runtime environment you are going to use, and the environment where you are going to build and use the solution.

To configure and use WebSphere Adapter for Flat Files, you must 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 use to build the solution. You must know how to use these tools to create modules, test components, and complete other integration tasks.
- The capabilities and requirements of the runtime environment you use for the integration solution. You must 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.

Security

WebSphere Adapter for Flat Files relies on the permissions of the user that starts the IBM Business Process Manager.

The user of the adapter must have sufficient privileges to access the directories and files that the adapter tries to access, read, or modify.

Support for protecting sensitive user data in log and trace files

You can configure the adapter to prevent sensitive or confidential data, in the log and trace files, from being viewed by users without authorization.

Log and trace files for the adapter can contain data from your local file system, which might contain sensitive or confidential information. Sometimes these files might be seen by individuals without authorization to view sensitive data. For example, a support specialist must use the log and trace files to troubleshoot a problem.

To protect the data in such situations, the adapter lets you specify whether you want to prevent confidential user data from displaying in the adapter log and trace files. You can select this option in the external service wizard or change the HideConfidentialTrace property. When this property is enabled, the adapter replaces the sensitive data with XXX's.

See "Managed connection factory properties" on page 172 for information about this optional property.

The following types of information are considered potentially sensitive data and are disguised:

- The contents of a business object
- The contents of the object key of the event record
- User name and password
- 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 are not hidden:

- The contents of the event record that are not part of the event record object key, for example, the XID, event ID, business object name, and event status
- Business object schemas
- Transaction IDs
- Call sequences

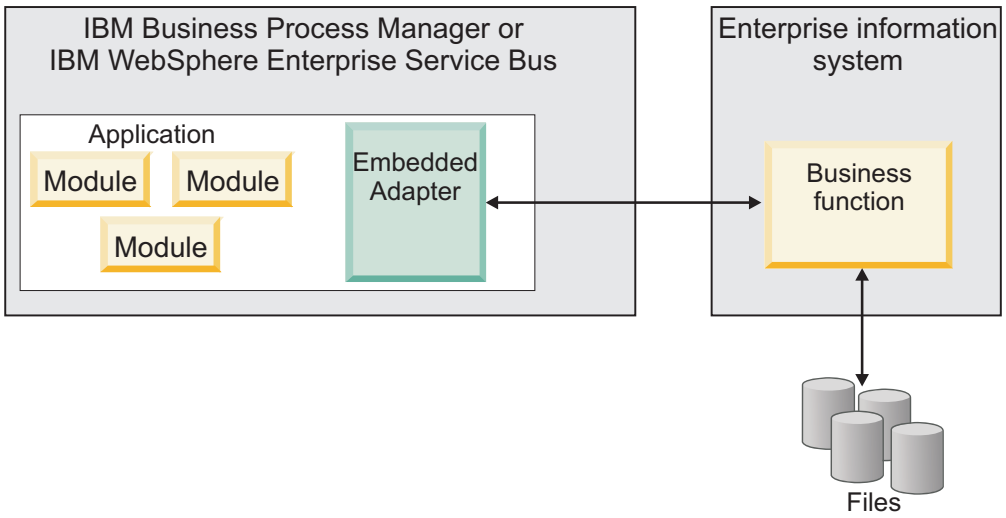
Deployment options

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 affect the type of deployment option you choose.

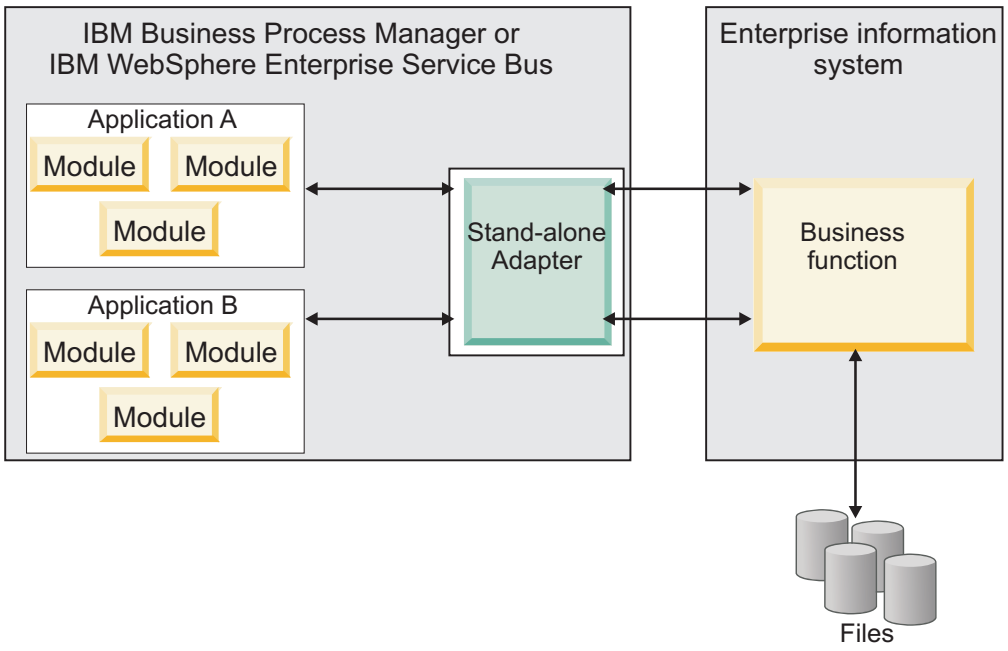
The following are the deployment options:

- **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 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 IBM Integration Designer, you can choose how to package the adapter [either bundled with the (EAR) file or as a stand-alone (RAR) file]. Your choice affects how the adapter is used in the run time 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 to run 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

Consider the following items if you plan to embed 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 run time environments. *Class loader isolation* means that 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

Consider the following items if you plan to use 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.

Note:

When you install multiple adapters with different versions of CWYBS_AdapterFoundation.jar, and if a lower version of the CWYBS_AdapterFoundation.jar is loaded during runtime, the adapter will return the ResourceAdapterInternalException error message, due to a version conflict. For example, when you install Oracle E-Business Suite adapter version 7.0.0.3 and WebSphere Adapter for Flat Files version 7.5, the following error message is displayed: IBM WebSphere Adapter for Flat Files has loaded file:/C:/IBM/WebSphere/ProcServer7/profiles/ProcSrv01/installedConnectors/CWYOE_OracleEBS.rar/CWYBS_AdapterFoundation.jar with version 7.0.0.3.

However, the base level of this jar required is version 7.5. When you install multiple adapters with different CWYBS_AdapterFoundation.jar versions, the adapter returns the ResourceAdapterInternalException message, due to a version conflict. To overcome this conflict, you must migrate all adapters to the same version level. For further assistance, contact WebSphere Adapters Support for help.

Considerations while deploying WebSphere Adapter 7.5 with another version

There are occasions when you have to work with embedded adapters that do not need a client-server communication, standalone adapters that need a server connection, or a hybrid mix of adapter connections.

The following scenarios cover the different behaviors of AFC version conflict detection.

Deploying a standalone Adapter

1. Install WebSphere Adapter for Flat Files version 7.0.1.0 through the IBM Business Process Manager administrative console.
2. Install WebSphere Adapter for SAP Software version 7.5.0.0 through the administrative console.
3. Create ActivationSpec for an ALE passthrough inbound operation.
4. Create an application in IBM Integration Designer for a standalone ALE passthrough inbound operation.
5. Install and start the application through the administrative console.
6. Verify the error.

Note: An error message will be generated in the log/trace area of IBM Business Process Manager, to indicate an AFC version conflict.

Deploying an embedded Adapter

1. Import a build of WebSphere Adapter for FTP version 7.0.1.0, using a RAR file.
2. Create a FTP Inbound EMD operation.
3. Import a build of WebSphere Adapter for Oracle E-Business Suite version 7.5.0.0, using a RAR file.
4. Create an Oracle inbound EMD operation, in the same module where you have created the FTP Inbound EMD operation.
5. Deploy the module to IBM Business Process Manager.
6. Check the trace.

At step 5, the deployment will fail. At step 6, you will get an internal error message due to the AFC version conflict.

Note: To avoid a name conflict between the business object generated by the two adapters, you may need to generate the artifacts into different folders.

Deploying a combination of standalone and embedded Adapters

1. Install WebSphere Adapter for JDBC version 7.0.1.0 through the IBM Business Process Manager administrative console.
2. Create an ActivationSpec for a JDBC inbound operation.
3. Create an application in IBM Integration Designer for a JDBC inbound operation, for the standalone Adapter deployment.

4. Deploy the JDBC inbound application and trigger your inbound events.
5. Create an application in IBM Integration Designer for a WebSphere Adapter for SAP Software version 7.5.0.0 inbound embedded Adapter deployment.
6. Deploy an SAP inbound application, and trigger your inbound events.

Note: You can resolve the AFC version conflict by using different class loaders for the standalone and embedded deployments. With this approach, the migration process will handle different CWYBS_AdapterFoundation.jar files and will not conflict with each other. You can start both JDBC and SAP inbound applications successfully, and process Inbound events without exception.

For further assistance, visit http://www-947.ibm.com/support/entry/portal/Overview/Software/WebSphere/WebSphere_Adapters_Family.

WebSphere Adapters in clustered environments

You can improve adapter performance and availability by deploying a module on a clustered server environment. Clusters are groups of servers that are managed together to balance workloads and to provide high availability and scalability.

The module you deployed is replicated across all servers in a cluster, regardless of whether you deploy the module using a stand-alone or an embedded adapter. The following IBM products support WebSphere Adapters in a clustered environment:

- IBM Business Process Manager or WebSphere Enterprise Service Bus
- WebSphere Application Server Network Deployment
- WebSphere Extended Deployment

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 an adapter instance. For information about creating clustered environments, see the following link: http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/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 instance instead of a static Workload Manager. The dynamic Workload Manager instance 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 systems with different capacities and configurations to handle load variations evenly. For information about the benefits of WebSphere Extended Deployment, see http://publib.boulder.ibm.com/infocenter/wxdinfo/v6r1m1/index.jsp.

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

Restriction: During inbound and outbound communication WebSphere Adapter for Flat Files is not able to switch polling between a IBM Business Process Manager 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 local file system. WebSphere Adapter for Flat Files 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 must be on a shared file system and not local to any of the cluster machines.

When you deploy a module to a cluster, the Java 2 Platform, Enterprise Edition (J2EE) Connector Architecture (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.

Note: In the active-passive configuration mode of the adapters, the endpoint application of the passive adapter instance also listens to the events/messages even if the `enableHASupport` property is set to `True`. This is because the `alwaysactivateAllMDBs` property in the JMS activation specification is set to `True`. To stop the endpoint application of the passive adapter instance from listening to the events, you must set the `alwaysactivateAllMDBs` property value to `False`. For more information, see "Disabling end point applications of the passive adapter" on page 155.

If the value for the enableHASupport property is set to `False`, all adapter instances poll for events in the inbound cluster and the adapter works in an Active-Active configuration. Multiple instances of WebSphere Adapter for Flat Files can be made active in a HA cluster in the active configuration mode. When more than one adapter instance actively polls in a cluster setup, it serves as a load balancer. If one of the adapter instances in the cluster fails, the other active instances in the cluster handle the events.

Note: In clustered environments, when the adapter works in a HA Active-Active configuration, it provides both high availability and load balancing support. This functionality is useful in production environments where high performance is needed.

In the HA Active-Active configuration, WebSphere Adapter for Flat Files ensures that an event is not processed by more than one adapter instance. This results in each adapter instance polling for a unique event, and delivering the event without any duplication to the endpoint.

Note:

- You must configure all the event persistence properties, if the adapter uses the HA Active-Active configuration.

- The local event directory must be present in a mapped drive that can be accessed by all the adapter instances in the clustered environment.
- Sorting of event files being polled is not supported.
- Supports only unordered delivery type of events to the export.
- In the Windows operating systems, such as, Windows 7, Windows Vista, and Windows Server 2008, there are issues faced in the mapped drive connection. Due to this issue, in a clustered environment, where the nodes are running on different machines, the files in the mapped event directory might not be processed completely or correctly. This may occur during both inbound and outbound operations. For more information about working with mapped drives, refer to articles on mapped drive connection to network sharing, for your operating system.

Database support in clustered environments

The adapter currently supports only the following databases:

- IBM DB2®
- Oracle
- Microsoft SQL Server
- Apache Derby

Note: If a different database is used, you must manually create the event persistence table and the file table. For more information about the event table and the file table, see "Event store structure" on page 18 and "File store" on page 20.

In addition, the databases must support the following features to enable the adapter to run in the Active-Active configuration:

- Batch Processing to allow efficient bulk database updates and automated transaction processing
- Transaction to ensure data integrity
- `FOR UPDATE` clause in the `SELECT` statement with queries that select a range of data that uses `LIMIT`, `TOP`, or the database equivalent.

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 Flat Files 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 about workload management, see the following link: http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html.

Migrating to version 7.5 of WebSphere Adapter for Flat Files

By migrating to version 7.5 of WebSphere Adapter for Flat Files, 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 use features and capabilities present in version 7.5.

Migration considerations

WebSphere Adapter for Flat Files version 7.5 may have some features and updates that might affect your existing adapter applications. Before migrating applications that use WebSphere Adapter for Flat Files, you must consider some factors that might affect your existing applications.

Compatibility with earlier versions

WebSphere Adapter for Flat Files version 7.5 is fully compatible with the custom business objects (XSD files) and data bindings that are created using the adapter version 6.1x, version 6.2x, and version 7.0 and enables the existing business objects and data bindings to work well in the latest version of the adapter.

Because version 7.5 of WebSphere Adapter for Flat Files is fully compatible with version 6.1x, version 6.2x, and version 7.0, any of your applications that used previous versions of WebSphere Adapter for Flat Files runs unchanged when you upgrade to version 7.5. However, if you want your applications to use features and functionality present in version 7.5 of the adapter, perform the migration of the artifacts as well as the upgrade of the adapter.

The migration wizard replaces (upgrades) version 6.1.x, version 6.2.x, or version 7.0 of the adapter with version 7.5 and enables version 7.5 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 7.5 of the adapters. If any of your applications embed an adapter that is version 7.0 or earlier and you are upgrading to version 7.5, and you want your applications to take advantage of the features and functions in version 7.5, you might need to change those applications.

If the artifacts within a module have inconsistent versions, the entire module is marked as unavailable for migration and cannot be selected. Version inconsistencies are recorded in the workspace log, as they indicate that a project might be corrupted.

The adapter migration wizard in IBM Integration Designer version 7.5 only supports the migration of adapters from version 6.1x, version 6.2x, and version 7.0 to version 7.5. It does not support the adapter migration from lower versions to any of the versions prior to version 7.5.

Deciding whether to upgrade or to upgrade and migrate

The default processing of the migration wizard is to perform an upgrade of the adapter and to migrate the application artifacts so that the applications can use features and functions in version 7.5 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 6.1.x, version 6.2.x and version 7.0 to version 7.5, 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 selecting any adapter artifacts installs and upgrades your adapter. As the artifacts are not migrated, your applications cannot take advantage of the features and capabilities that exist in version 7.5 of the adapter.

Migrating multiple adapters referred within a project

When a module contains one or more connector projects, each of which references to different adapters (for example, a module project that contains connector projects referring to JDBC and SAP adapters), the migration wizard identifies the artifacts belonging to each adapter and migrates these artifacts without disrupting the artifacts of other adapters.

When you select the module project and launch the migration wizard:

- The **Source connector** field lists the connector projects with the selected module project.
- The **Dependent artifact projects** area lists only the selected module project.

If you select the connector project and launch the migration wizard:

- The **Source connector** field lists only the selected connector project.
- The **Dependent artifact projects** area lists all projects which reference the selected connector project, including the module project.

Run the migration wizard in a test environment

Because adapter migration might require you to change those applications that use version 7.5 of WebSphere Adapter for Flat Files, you must 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

Become familiar with the deprecated features in version 7.0 and make any required changes to your applications.

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 Flat Files have been deprecated in version 6.2.x and might require changes to your applications:

- Activation specification:
  - ArchivingProcessed
  - EventContentType
  - DefaultObjectName
- InteractionSpecification:
  - DefaultObjectName
- Wrapper properties:

– RetrieveContentType
– DefaultObjectName

Performing the migration

You can migrate a project or EAR file to version 7.5 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 IBM Integration Designer, complete the following steps.

Note: After migration is complete, the following changes occur:

- the module will no longer be compatible with previous versions of IBM Business Process Manager or WebSphere Enterprise Service Bus, IBM Business Process Manager or WebSphere Enterprise Service Bus, or IBM Integration Designer.
- an XML data handler is added to all the operations. Because this data handler is not needed for the pass-through operation, you must configure one data binding without the data handler against the pass-through operation.
- a file table is automatically created for storing the event persistence information. The table is created for the supported databases.

The following steps describe how to run the adapter migration wizard from the connector project menu while in the Java EE perspective in IBM Integration Designer.

Procedure

1. Import the PI (project interchange) file for an existing project into the workspace.

   Note: Ensure that you do not modify the contents of the RAR or copy the adapter JAR file outside the connector project.
2. When projects are created in an earlier version of IBM Integration Designer, the Workspace Migration wizard starts automatically and selects the projects to migrate. Follow the wizard and complete the workspace migration. For more information, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.wid.imuc.doc/topics/tmigsrcart.html.
3. Change to the Java EE perspective.
4. Right-click the module and select **Migrate connector project**. For example, the adapter RAR module.

   You can also launch the adapter migration wizard in the following ways:

   - Right-click the project in the Java EE perspective and select **Migrate adapter artifacts**.
   - From the Problems view, right-click a migration-specific message and select **Quick Fix** to correct the problem.
5. In the Select Projects window, perform the following steps:

a. The **Source connector** field displays the name of the connector project that you are migrating. If you are migrating a module project, this field lists all the connector projects in the module project. Select the source project from the list. For more information, see "Migrating multiple adapters referred within a project" on page 44.

b. The **Target connector** field displays the name of the connector to which you are migrating. If you are working with more than one adapter version, this list displays the names of all the compatible connectors. Select the connector you want to migrate.

c. The **Target version** field displays the version corresponding to the target connector that you selected in the previous step.

d. The **Dependent artifacts project** area lists the adapter artifacts that are migrated. If you are migrating a module project, this area lists only the selected module project. If you are migrating a connector project within the module project, this area lists all projects which reference the selected connector project, including the module project. By default, all the dependent artifact projects are selected. If you do not select a dependent artifact project, that project is not migrated. You can migrate any project that you have not selected 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. For more information, see "Upgrading but not migrating a project" on page 48.

e. Click **Next**. A warning window is displayed with the message, "Properties that are not supported in this version of the target adapter will be removed during the migration".

f. Click **OK**.

6. In the Review Changes window, review the migration changes that occur in each of the artifacts that you are migrating. To view the details, expand each node by clicking the + sign.
7. To complete the migration:
   - Click **Finish**.
   - If the files that need to be updated during migration are in read-only mode, you will be unable to click on the **Finish** button. To view these files, click **Next**. The Update Read-only files window displays the read-only files. To update these files and continue with the migration, click **Finish**. To exit the wizard without migrating the adapter, click **Cancel**.

   Before running the migration process, the wizard performs a backup of all projects affected by the migration. The projects are backed up to a temporary folder within the workspace. If the migration fails for any reason, or if 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. If you are migrating an EAR file, optionally create a new EAR file with the migrated adapter and artifacts, and deploy it to IBM Business Process Manager 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 7.5. You do not need to run the external service wizard after exiting the adapter migration wizard.

What to do next

After completing the migration, you must manually update the structure of the event table. To update the structure of the event table, use the sample database scripts available at "<IID_HOME>\Resource Adapters\FlatFile_7.5.0.1\Scripts".

Migrating databases

With WebSphere Adapter for Flat Files, version 7.5, the schema of the event persistence table is modified. Hence, after completing the adapter migration, you must update the structure of the event table to work with the adapter version 7.5. Use the sample database scripts available at "<IID_HOME>\Resource Adapters\FlatFile_7.5.0.1\Scripts" to update the structure of the event table.

Before you begin

Before updating the structure of the event table, ensure that only the failed events are available in the event table. Ensure that you process or delete the unprocessed events before performing this task.

Note: The database migration is required for both single and HA Active-Active instance of adapter configuration. After migrating the adapter, if you use an existing event table with the adapter version 7.5, then a runtime exception is thrown.

About this task

Perform the following steps to run the scripts and update the structure of the event table.

Procedure

1. Go to the "<IID_HOME>/Resource Adapters/FlatFile_7.5.0.1/Scripts" folder.
2. Double-click one of the following scripts corresponding to your database:
   - scripts_db2_upgrade.sql – for DB2 and Derby database
   - scripts_mssql_upgrade.sql – for Microsoft SQL Server database
   - scripts_oracle_upgrade.sql – for Oracle database
3. The selected script performs the following actions:
   a. A temporary event table with the same structure of the existing event table is created.

      Note: Ensure that the name of the temporary event table (mentioned in the script) is not already in use for any existing table. If the name is already in use, then change the name of the temporary event table in the database script accordingly.

   b. The event data from the existing event table is copied to the temporary event table.
   c. If the default name of the event table is not used in your application or project, ensure that you specify the name of the existing event table in the database script.
   d. An event table with the new structure is created.
   e. After the data from the temporary table is copied to the new event table, the temporary table is deleted from the database.

Results

The updated event table can now be used in your project.

Upgrading but not migrating a project

You can upgrade the adapter from an earlier version, to version 7.5 while choosing not to migrate the adapter project artifacts.

About this task

Running the migration wizard without selecting any adapter artifacts installs and upgrades your adapter. As the artifacts are not migrated, your applications cannot take advantage of the features and capabilities that exist in version 7.5 of the adapter.

Procedure

1. Import the PI (project interchange) file into the workspace.
2. When projects are created in an earlier version of IBM Integration Designer, the Workspace Migration wizard starts automatically and selects the projects to migrate. Follow the wizard and complete the workspace migration. For more information, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.wid.imuc.doc/topics/tmigsrcart.html.
3. In the Java EE perspective, right-click the project name and click **Migrate connector project**. The **Adapter Migration** wizard is displayed.
4. In the Select Projects window, clear the dependent artifact projects, and click **Next**. A warning window is displayed with the message, "The properties that are not supported in the version of the target adapter will be removed during the migration."
5. Click **OK**.
6. In the Review Changes window, review the migration changes that occur during updating the project. To view the details, expand each node by clicking the + sign.
7. To complete the migration:
   - Click **Finish**.
   - If the files that need to be updated during migration are in read-only mode, you will be unable to click on the **Finish** button. To view these files, click **Next**. The Update Read-only files window displays the read-only files. To update these files and continue with the migration, click **Finish**. To exit the wizard without migrating the adapter, click **Cancel**.

Results

The project can now be used with WebSphere Adapter for Flat Files, version 7.5.

Migrating WebSphere Business Integration applications

You need to migrate the WebSphere Business Integration applications so that they become compatible with Version 7.5 of your adapter.

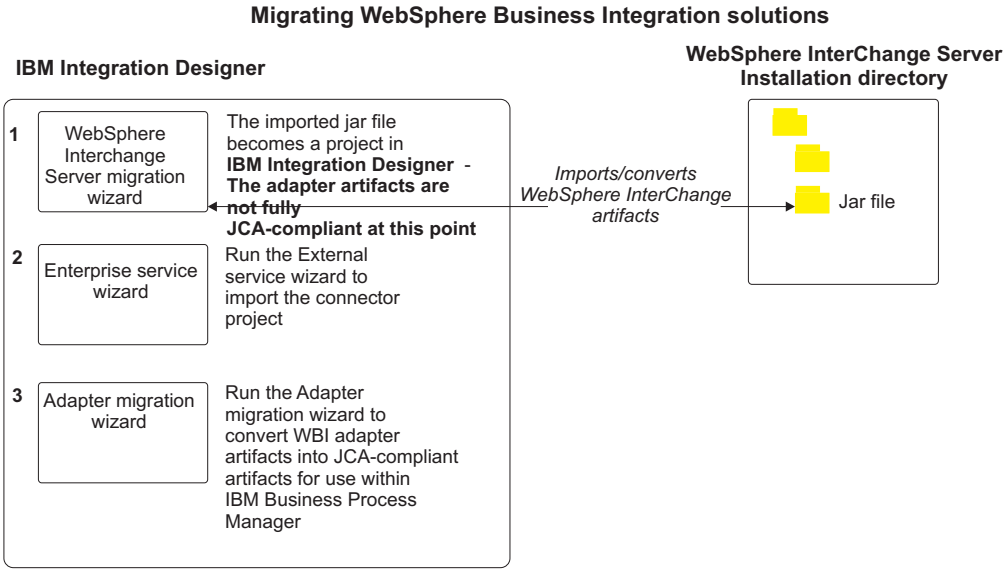
About this task

Migrating WebSphere Business Integration applications for use with Version 7.5 of your WebSphere adapter is a multistep process. First, the artifacts from WebSphere

InterChange Server are migrated and converted. A project is then created for the artifacts in IBM Integration Designer. In the remaining steps, the adapter-specific artifacts are migrated and converted into the JCA-compliant format supported by Version 7.5 of the adapter.

Example

The following diagram shows the wizards that you use to migrate WebSphere Business Integration solutions from WebSphere InterChange Server, so that these applications can be used with Version 7.5 of your adapter.



Migrating applications from WebSphere InterChange Server

To use Version 7.5 of WebSphere Adapter for Flat Files with applications from WebSphere InterChange Server, you need to migrate the application artifacts and convert them so that they can be deployed and run on IBM Business Process Manager or WebSphere Enterprise Service Bus. Understanding this task at a high level helps you perform the steps that are needed to accomplish the task.

The following figure illustrates the flow of the migration task. The steps that follow the figure describe this task at a high level only. See the topics following this roadmap for the details on how to perform each of these steps.

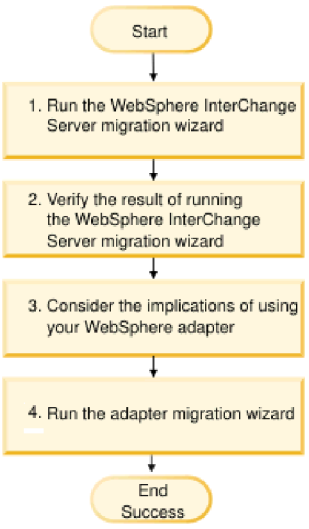


Figure 9. Roadmap for migrating applications from WebSphere InterChange Server

Migrating applications from WebSphere InterChange Server

This task consists of the following steps:

1. Run the WebSphere InterChange Server migration wizard.

   The WebSphere InterChange Server migration wizard moves the application artifacts into IBM Integration Designer. The migrated adapter artifacts are not fully JCA-compliant at the completion of this task.
2. Verify that the WebSphere InterChange Server migration is successful.

   Review all messages from the Migration results window and take action if required.
3. Consider the implications of using Version 7.5 of WebSphere Adapter for Flat Files.

   In addition to considerations for migrating WebSphere InterChange Server applications, you need to consider how Version 7.5 of WebSphere Adapter for Flat Files works with the migrated applications. Some of the adapter operations supported by WebSphere InterChange Server applications might be supported and implemented differently with Version 7.5 of the adapter.
4. Run the adapter migration wizard.

   Run the adapter migration wizard to update adapter-specific artifacts such as the schemas and service definition files (.import,.export, and .wsdl files) for use with Version 7.5 of the adapter.

Migration considerations for WebSphere Business Integration adapters

By migrating to WebSphere Adapter for Flat Files Version 7.5, you have an adapter that is compliant with the Java 2 Platform, Enterprise Edition (J2EE) Connector Architecture (JCA) and designed specifically for service-oriented architecture.

Application artifacts

Before running the adapter migration wizard, use the WebSphere InterChange Server migration wizard to generate the application artifacts for the WebSphere Business Integration adapter, including the business objects, maps, and collaborations. Then you can run the adapter migration wizard to update the

adapter-specific artifacts such as the schemas and service definition files (.import,.export, and .wsdl) so that they are suitably converted into a format that is compliant with JCA.

Run the migration wizard in a test environment first

Because migrating from a WebSphere Business Integration adapter to WebSphere Adapter for Flat Files might require changes to the applications that use Version 7.5 of WebSphere Adapter for Flat Files, always perform the migration in a development environment first and test your applications before deploying the application to a production environment.

Migrating application artifacts from WebSphere InterChange Server

To migrate the application artifacts into IBM Integration Designer, run the WebSphere InterChange Server migration wizard. The wizard imports and converts most of the artifacts into a format that is compatible with IBM Business Process Manager or WebSphere Enterprise Service Bus.

Before you begin

Launch the WebSphere InterChange Server migration wizard from within IBM Integration Designer to migrate the application artifacts from WebSphere InterChange Server format into artifacts that are compatible with IBM Business Process Manager or WebSphere Enterprise Service Bus.

For information about how to prepare to migrate artifacts from WebSphere InterChange Server and for detailed instructions on performing the migration and verifying that the migration was successful, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.wid.imuc.doc/topics/twics.html.

About this task

Running WebSphere InterChange Server migration wizard might not fully convert adapter-specific artifacts (such as service descriptors, service definitions, and business objects) into IBM Business Process Manager or WebSphere Enterprise Service Bus compatible artifacts. To complete the migration of adapter-specific artifacts, run the adapter migration wizard after you have successfully run the WebSphere InterChange Server migration wizard.

Note: While you run the WebSphere InterChange Server migration wizard, ensure that you set each connector in the repository to the same adapter version.

Results

The project and application artifacts are migrated and converted into IBM Business Process Manager compatible artifacts.

What to do next

Run the adapter migration wizard to migrate the adapter-specific artifacts.

Migrating adapter-specific artifacts

After a project is created for the artifacts in IBM Integration Designer, you can migrate the project using the adapter migration wizard. The adapter migration wizard updates adapter-specific artifacts such as the schemas and service definition files (.import, .export, and .wsdl) for use with version 7.5 of the adapter. When you finish running the adapter migration wizard, the migration is complete and you can work in the project or deploy the module.

Before you begin

Before running the adapter migration wizard, you should do the following steps:

- Review the information in "Migration considerations" on page 43.
- Run the WebSphere InterChange Server migration wizard to migrate the project and convert data objects for use with IBM Business Process Manager or WebSphere Enterprise Service Bus.

About this task

After migration is complete, the module will work only with Version 7.5 of your adapter.

To perform the migration in IBM Integration Designer, complete the following steps.

Procedure

1. Import the PI (project interchange) file for an existing project into the workspace.
2. When projects are created in an earlier version of IBM Integration Designer, the Workspace Migration wizard starts automatically and selects the projects to migrate. Follow the wizard and complete the workspace migration. For more information, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.wid.imuc.doc/topics/tmigsrcart.html.
3. Change to the Java EE perspective.
4. Right-click the connector project and select **Migrate connector project**.

   You can also launch the adapter migration wizard by using the right-click option and selecting the module project in the Java EE perspective and selecting **Migrate adapter artifacts**.

   Note:

   If the adapter type (for example, CICS/IMS adapter) is not supported by the migration wizard, the **Migrate connector project** and **Migrate adapter artifacts** menus are not available for selection. If the adapter project is of the latest version and the module projects referencing this adapter project are also of the latest version, these menus are disabled.
   When you launch the migration wizard from the connector project while in the Java EE perspective, by default all the dependent artifact projects are selected. If you do not select a dependent artifact project, that project is not migrated.
5. In the Select Projects window, perform the following steps:
   a. The **Source connector** field displays the name of the connector project that you are migrating. Select the source project from the list.

b. The **Target connector** field displays the name of the connector to which you are migrating. If you are working with more than one adapter version, this list displays the names of all the compatible connectors. Select the connector to which you want to migrate.

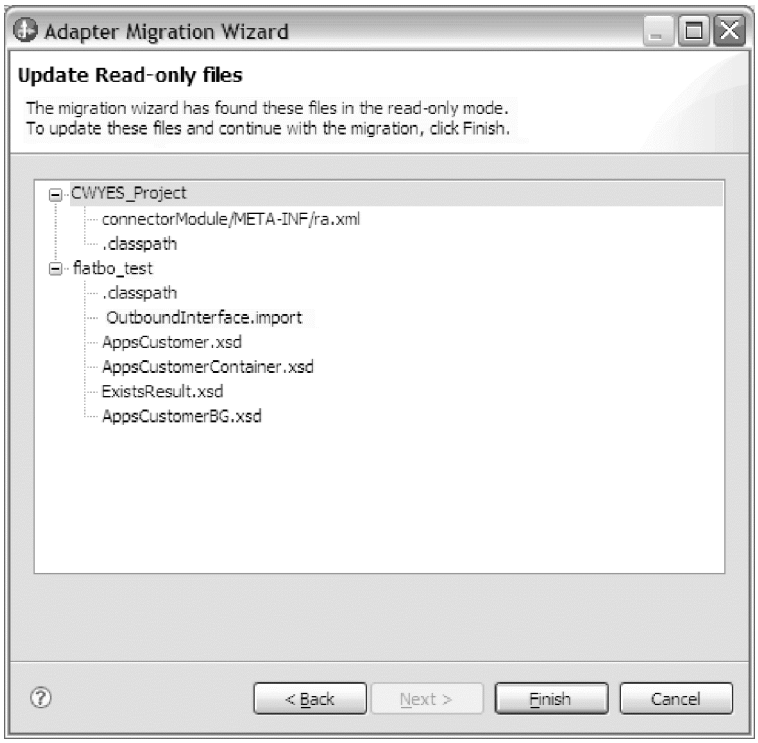
c. The **Target version** field displays the version corresponding to the target connector you selected in the previous step.

d. The **Dependent artifacts project** area lists the adapter artifacts that are migrated.

e. Review the tasks and warnings presented on the welcome page, and click **Next**. A warning window is displayed with the message, "The properties that are not supported in the version of the target adapter are removed during the migration."

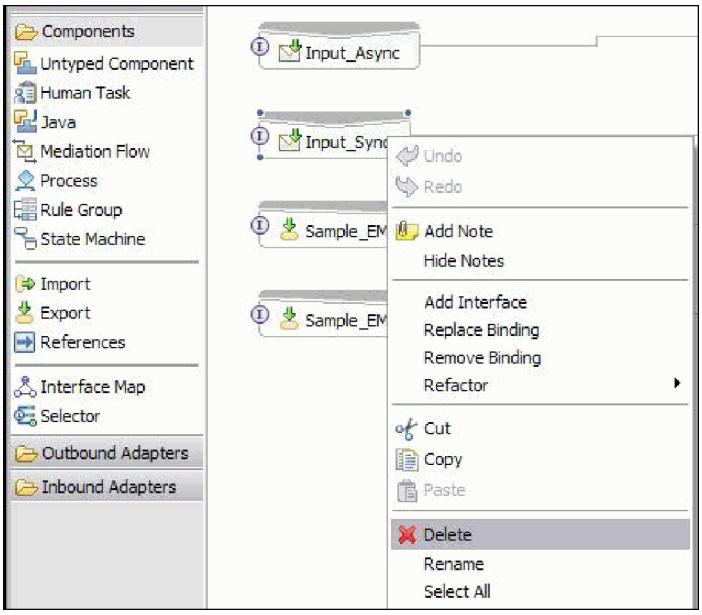
f. Click **OK**.

6. In the Review Changes window, review the migration changes that occur in each of the artifacts that you are migrating. To view the details, expand each node by clicking the + sign.
7. To complete the migration:
   - Click **Finish**.
   - If the files that need to be updated during migration are in read-only mode, you will be unable to click on the **Finish** button. To view these files, click **Next**. The Update Read-only files window displays the read-only files. To update these files and continue with the migration, click **Finish**. To exit the wizard without migrating the adapter, click **Cancel**.



Before performing the migration process, the wizard backs up all projects affected by the migration. The projects are backed up to a temporary folder within the workspace. If the migration fails for any reason, or if 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.

8. Select **Project > Clean**, to refresh and rebuild the workspace for the changes to take effect.
9. On successful migration, all backed up projects are deleted. Remove the Sync inbound flow manually as this flow is not used by the adapter. From the migrated project, select the Input_Sync inbound flow, right-click and select **Delete**.



10. If you are migrating an EAR file, create a new EAR file with the migrated adapter and artifacts, and deploy it to IBM Business Process Manager or WebSphere Enterprise Service Bus. For information about exporting and deploying an EAR file, see "Deploying the module for production" on page 121.

Results

The project is migrated to Version 7.5. You do not need to run the external service wizard after exiting the adapter migration wizard.

Changes to the import, export, and WSDL files after migration

When the WebSphere InterChange Server migration wizard moves the application artifacts into IBM Integration Designer, changes made are reflected in the service definition files: the import, export and WSDL files.

The migrated adapter artifacts are not fully JCA-compliant at the completion of this task. You can complete the migration of the adapter-specific artifacts (such as service descriptors, service definitions, and business objects) to a JCA compatible format by running the adapter migration wizard.

Changes to the import file

During migration, the affected module artifacts are migrated to an import file. The existing JMS Binding property is changed to the EIS Binding property in the import file. The other property details added in the import file include information about the data binding configuration, changes to the connection information in the Managed Connection Factory properties, and several new method bindings.

Changes to the export file

During migration, the affected module artifacts are migrated to an export file. The existing JMS Binding property is changed to the EIS Binding property in the export file. The other property details added in the export file include information about the data binding configuration, changes to the connection information in the Activation Specification properties, and several new method bindings.

Changes to the WSDL file after migration

During migration, the affected module artifacts are migrated to corresponding WSDL files that include Flat File specific service description WSDL artifacts. The service description files become JCA compatible. The WSDL files will have an input and output type for each operation. Both the inbound and outbound operations work on their specific input types to produce corresponding output types after the operations are performed. The outbound operations generated during migration are shown in the following table:

Operations	Input	Output
Create	CustomerWrapperBG	CreateResponse
Append	CustomerWrapperBG	AppendResponse
Overwrite	CustomerWrapperBG	OverwriteReponse

Note:

- When you migrate multiple top level inbound business objects in the project, only the first top-level business object inbound feature works correctly. For the other top level inbound business object to work correctly, you must manually modify the "emit + [verb name] + after image + [business object name]" method in the Input_Processing.java and Input_Async_Processing.java class to call the correct destination services.
- The WebSphere Business Integration Adapter for JText properties that are either not valid or not supported by WebSphere Adapter for Flat Files are removed from the migrated artifacts.

Chapter 3. Samples and tutorials

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

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

- From the welcome page of IBM Integration Designer, click **Go to Samples and Tutorials**. In the Samples and Tutorials pane, under More samples, click **Retrieve**. Browse the displayed categories to make your selection.
- From the Business Process Management Samples and Tutorials website: 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 IBM Business Process Manager or WebSphere Enterprise Service Bus, use IBM Integration Designer 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.

Road map for configuring the module

Before you can use WebSphere Adapter for Flat Files 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 Flat Files by using IBM Integration Designer. 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 road map.

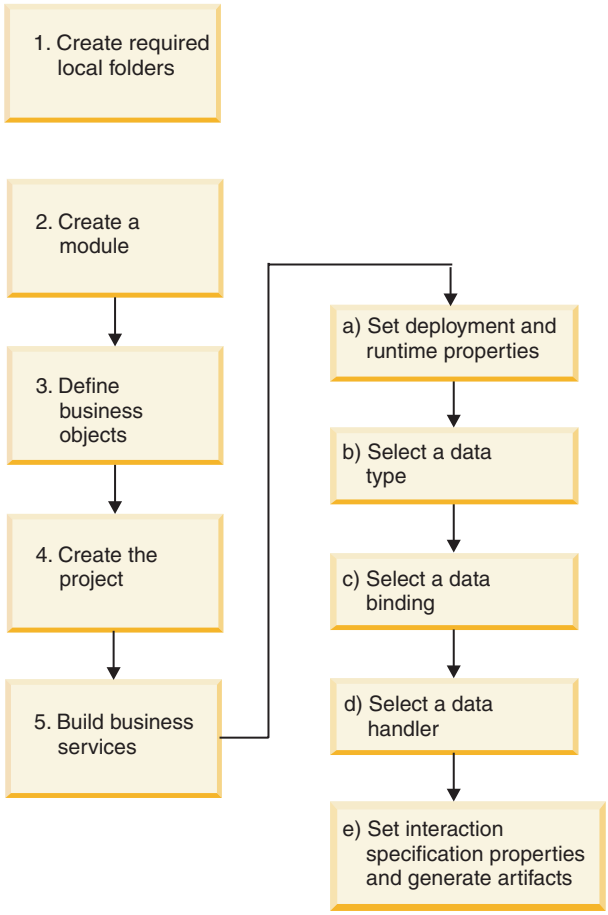


Figure 10. Road map for configuring the module

Configuring the module

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

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

1. Create a module in IBM Integration Designer. You create business objects in the module.
2. Define the business objects that to be used by the project.
3. Create a project, which is used to organize the files associated with the adapter using the external service wizard in IBM Integration Designer.
4. Build business services by running the external service wizard from IBM Integration Designer, 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 information:
      - The operation kind. For example, Create, Append, Exists.
      - Specify if the operation is pass-through 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 to 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 the required local folders

Before you create inbound or outbound modules, you must create folders on the local file system for events and output. You can optionally create folders for staging and archiving.

Before you create inbound or outbound modules, you must specify the event directory and the output directory on the Service Configuration Properties screen of the external service wizard. You can also create a staging directory and an archive directory, but these directories are not required.

- The event directory stores events for inbound processing. The adapter polls this folder at regular intervals and sends any found events, in the form of business objects, to the server.
- The output directory is used by the adapter to write the final output files for Create, Append, and Overwrite operations during outbound processing.
- The staging directory is a temporary directory where the adapter writes the initial output files during Create and Overwrite operations, to avoid write conflicts. The output files are then renamed and copied to the output directory.

- The archive directory is a directory where the adapter stores processed event files.

Instead of specifying the names of these directories when you run the external service wizard, you can use WebSphere Application Server environment variables.

Related tasks

"Defining WebSphere Application Server environment variables" on page 62
Use the administrative console of IBM Business Process Manager or WebSphere Enterprise Service Bus to define WebSphere Application Server environment variables.

Related reference

"Managed connection factory properties" on page 172
Managed connection factory properties specify information the adapter needs at run time for outbound communication with the local file system.

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

Creating the module

You create the module in IBM Integration Designer. The module allows you to define business objects that are used by the project.

About this task

Start the external service wizard and follow this procedure to create a module.

Procedure

1. If IBM Integration Designer is not currently running, start it now.
   a. Click **Start** > **Programs** > **IBM** > **IBM Integration Designer** > **IBM Integration Designer 7.5**.
   b. If you are prompted to specify a workspace, either accept the default value or select another workspace.

      The workspace is a directory where IBM Integration Designer stores your project.
   c. Optional: When the IBM Integration Designer window is displayed, click **Go to the Business Integration perspective**.
2. Right-click inside the Business Integration section of the IBM Integration Designer window. Click **New** > **Module**.

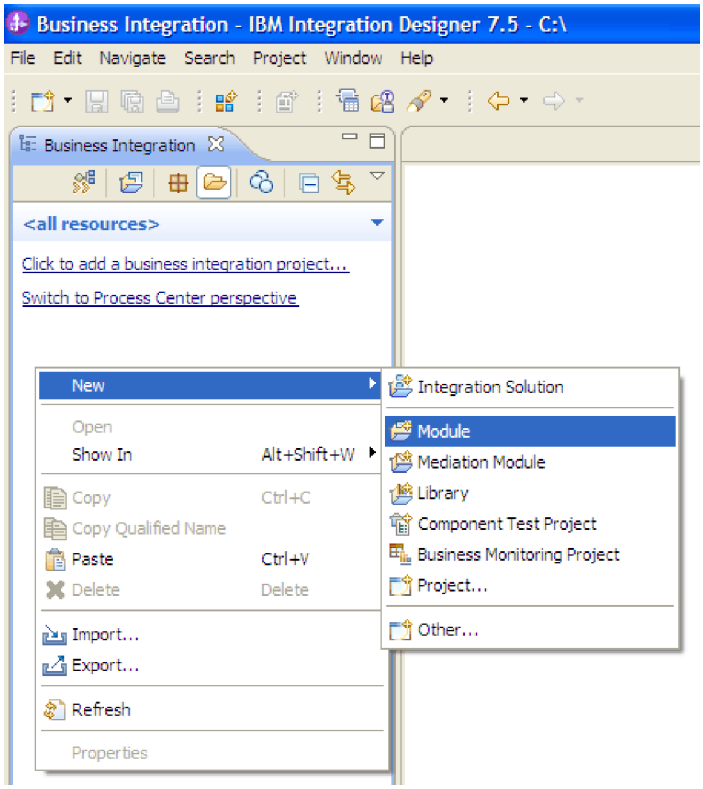


Figure 11. Business Integration section of the window

3. Type a new **Module Name** in the New Module window. Leave the other options (**Use default location** and **Open module assembly diagram**) checked.
4. 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 WebSphere Application Server environment variables

Use the administrative console of IBM Business Process Manager or WebSphere Enterprise Service Bus 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.
2. 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, `Cell=widCell` is used.

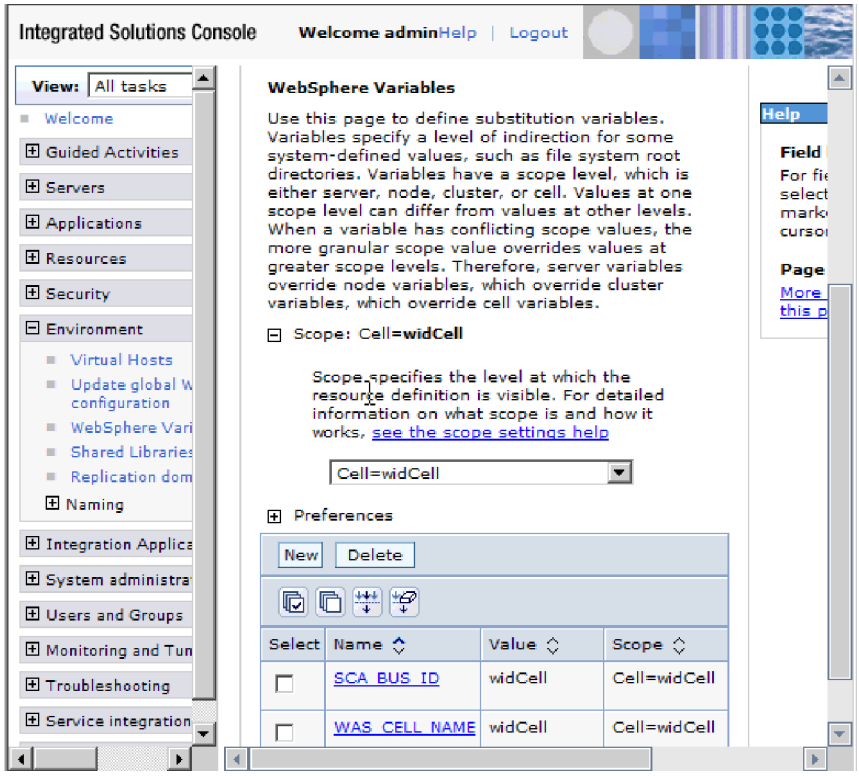


Figure 12. Setting the scope for the environment variable

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 `C:/flatfile/event`. You can use the **Description** field, which is optional, to describe the purpose of the variable.

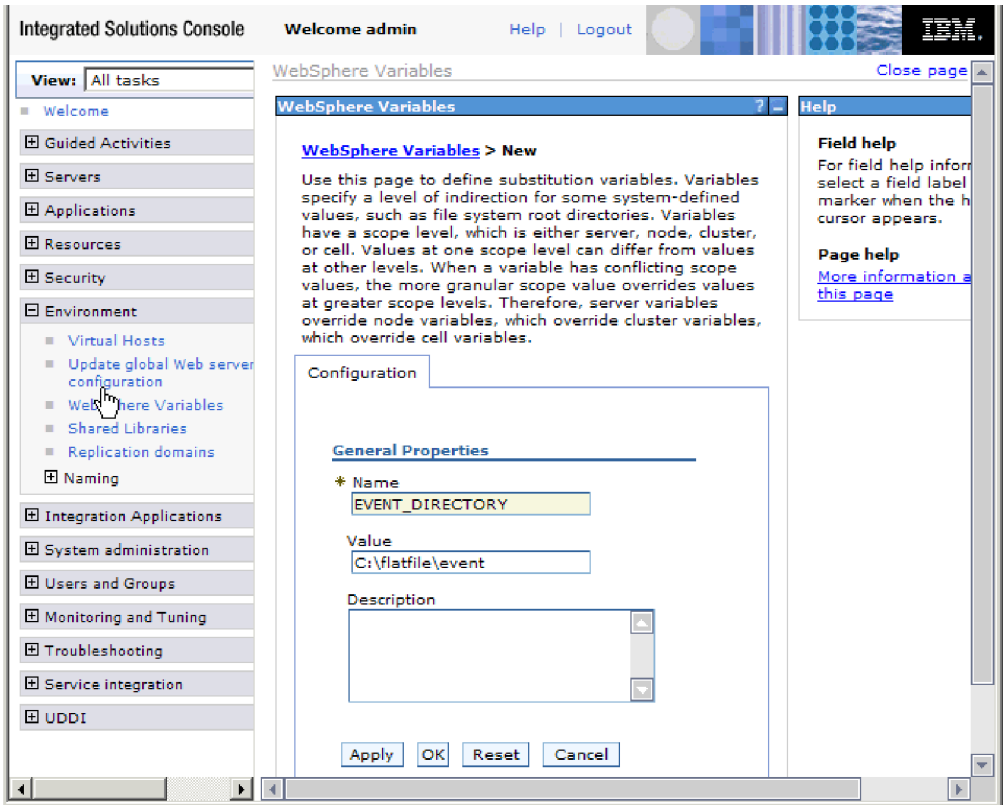


Figure 13. Providing a name and a value for the environment variable

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

Results

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

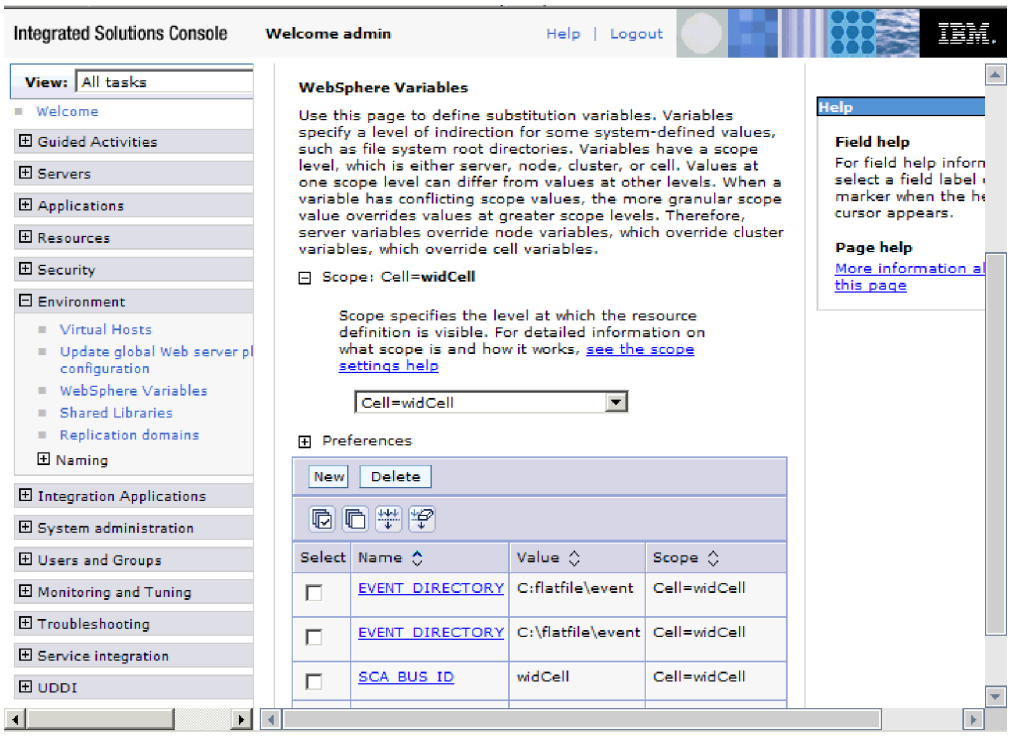


Figure 14. The new environment variable EVENT_DIRECTORY displayed in the WebSphere Variables window

What to do next

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

Related concepts

"WebSphere Application Server environment variables" on page 31
WebSphere Application Server environment variables can be used in the external service wizard to specify directory values.

"Creating the required local folders" on page 60
Before you create inbound or outbound modules, you must create folders on the local file system for events and output. You can optionally create folders for staging and archiving.

Related reference

"Managed connection factory properties" on page 172
Managed connection factory properties specify information the adapter needs at run time for outbound communication with the local file system.

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

Defining business objects

Predefine the business objects in IBM Integration Designer to be used by the project that you create in the next topic.

About this task

To predefine new business objects using the business object editor, complete the following steps.

Procedure

1. Expand the new module located inside of the Business Integration section of the IBM Integration Designer window.
2. Right-click the **Data Types** folder and select **New > Business Object**.
3. Type a new **Name** in the Business Object window. For example, `Customer` to create a customer business object.
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.

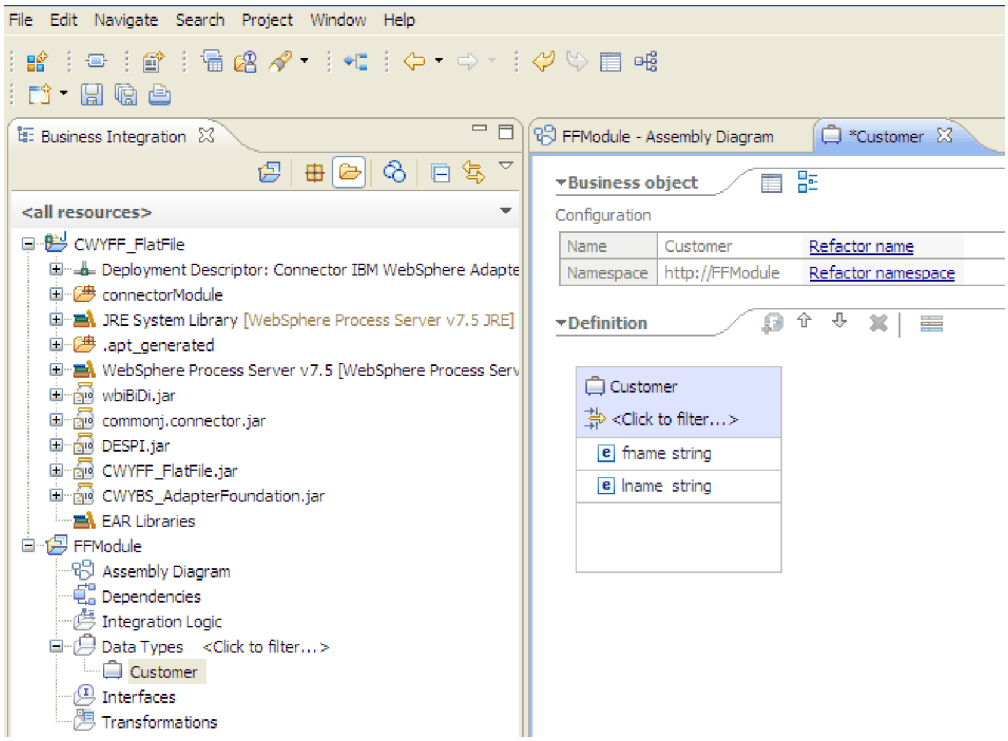


Figure 15. Add Business object fields icon

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.

Related concepts

"Business objects" on page 30
A business object is a logical data container that represents the data that is processed by the adapter. The data can represent either a business entity, such as an invoice or an employee record, or unstructured text, such as the body of an e-mail or a word-processing document. The adapter uses business objects to send data to or obtain data from the local file system.

Related reference

"Business object information" on page 159
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 local file system. The name typically reflects the operation to be performed and the structure of the business object.

Converting business objects to COBOL copybook files during outbound processing

Use the external service wizard in IBM Integration Designer 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 IBM Integration Designer.
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 service 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 service wizard to generate a wrapper business object definition from the generated business object.

Procedure

1. Generate the business object definition for the COBOL program source file.
   a. In the Business Integration section of the window, right-click the module and select **New** > **New Business Object From External Data**.
   b. In the Select an Input Source window, select **Languages -> COBOL**. Click **Next**.

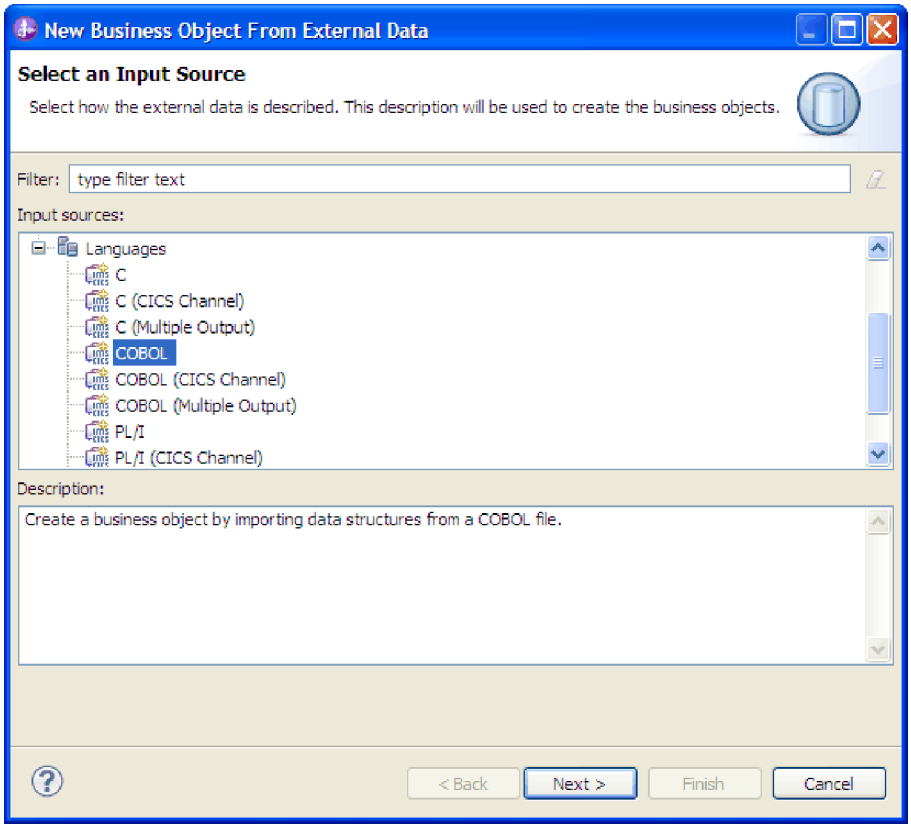


Figure 16. The Select an Input Source window

c. In the Provide Details for the Mapping window, make sure the **Selected mapping** value is **COBOL to Business Object**. Click **Browse** and select the .ccp file. For example, `taderc99.ccp` can be the name of the .ccp file. Click **Next**.

d. In the Select Data Structures window, click **Find**. The new business object, called DFHCOMMAREA is displayed.

e. Select the business object DFHCOMMAREA and click **Next**.

f. In the Generate Business Objects window, specify the **Module**, **Folder**, **Name**. Select **Generate style** from the list and 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 option to generate wrapper business object definitions is displayed only when the adapter RAR file has been imported into the workspace.

a. In the Business Integration section of the window, right-click the module and select **New** > **New Business Object From External Data**.

b. In the Select an Input Source window, expand **Adapters** and select **Flat File**. Click **Next**.

c. In the Specify the Location window, select the module in the **Module** field and click **Next**.

d. In the Specify the Properties window, click **Browse** and select the business object created in Step 1. For example, DFHCOMMAREA for the data type.

e. To generate a business graph, select the **Generate business graph for each business object** check box. To generate a retrieve wrapper, select the **Generate retrieve container to retrieve multiple business objects** check box.

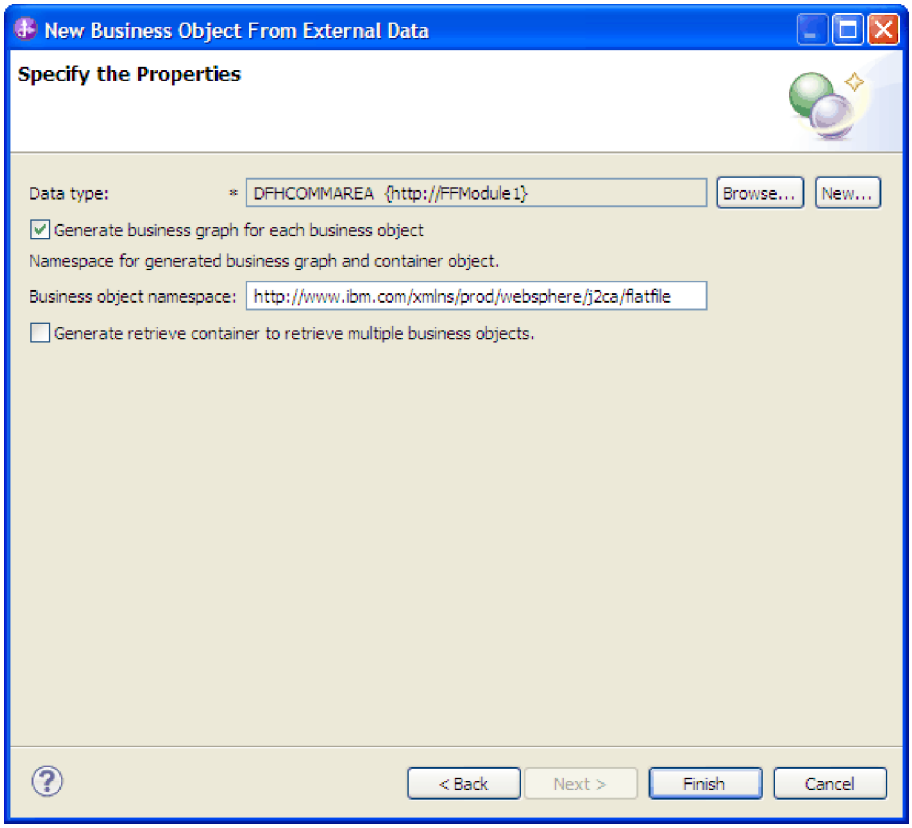


Figure 17. The Specify the Properties window

f. Click **Finish**.

A wrapper business object and a business graph, called DFHCOMMAREAWrapper and DFHCOMMAREAWrapperBG as shown in the figure, are listed for the current module in the Business Integration window. If wrappers for retrieve are selected, then business object called DFHCOMMAREARetrieveWrapper and a business graph called DFHCOMMAREARetrieveWrapperBG are also listed for the current module in the Business Integration window.

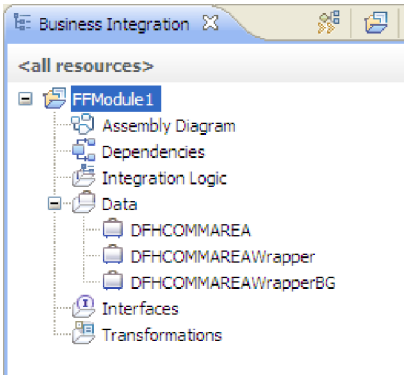


Figure 18. The wrapper business object and the business graph listed in the Business Integration window

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 the module and select **New** > **External Service**.
   b. Select **Adapters** and click **Next**.
   c. In the Select an Adapter window, select **IBM WebSphere Adapter for Flat Files (IBM : 7.5.0.1)** adapter and click the **CWYFF_FlatFile** connector project. Click **Next**.
   d. In the Select the Processing Direction window, select **Outbound**.
   e. Click **Next**.
   f. In the Specify the Security and Configuration Properties window, from the **Data format options** list, select **Use COBOL, C or PL/I data format** option. Click **Next**.

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

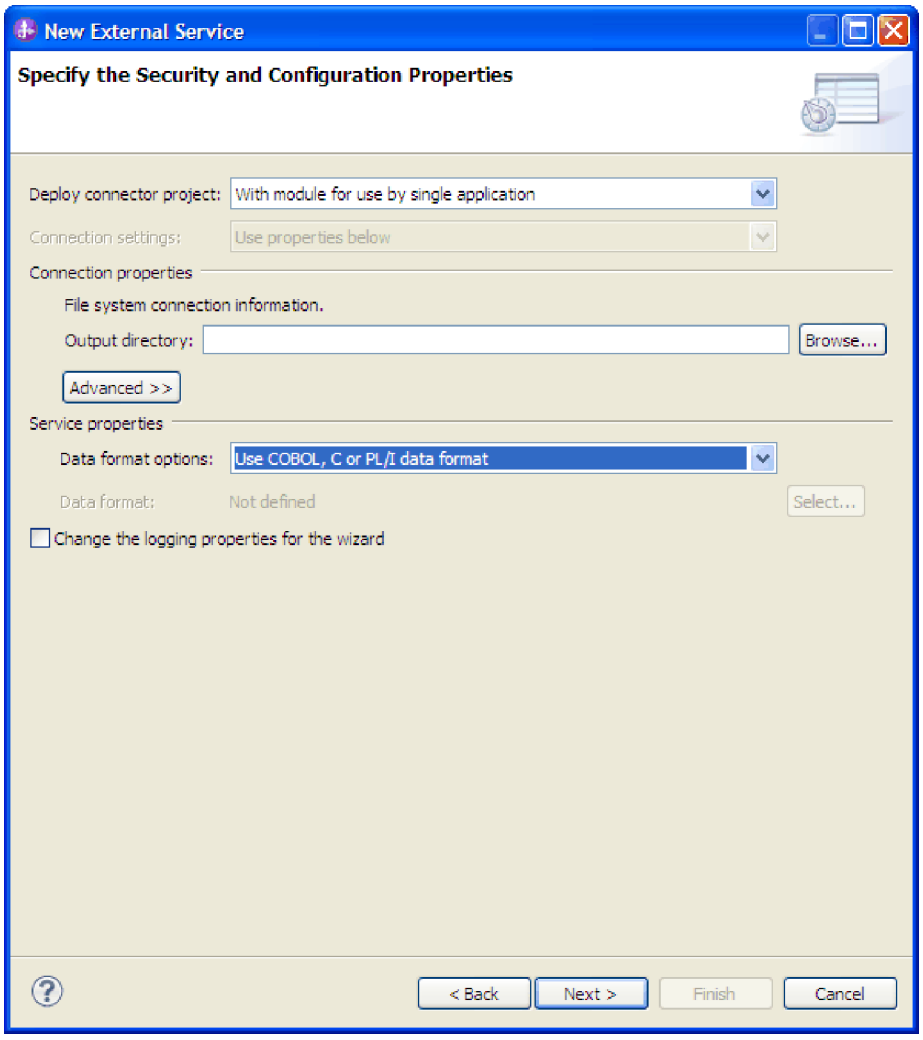


Figure 19. The Specify the Security and Configuration Properties window

g. In the Add, Edit, or Remove Operations window, click **Add**.

h. In the Specify the I/O Properties window, select **Create** in the Operation kind field. For `Retrieve` operation, select **Retrieve**.

i. Select **User-defined type** for the **Data type for the operation** field. Click **Next**.

j. Browse for the input type (DFHCOMMAREA, DFHCOMMAREAWrapper, or DFHCOMMAREAWrapperBG) and click **OK**. For **Retrieve** operation, browse for the appropriate input type (DFHCOMMAREA, DFHCOMMAREARetrieveWrapper, or DFHCOMMAREARetrieveWrapperBG).

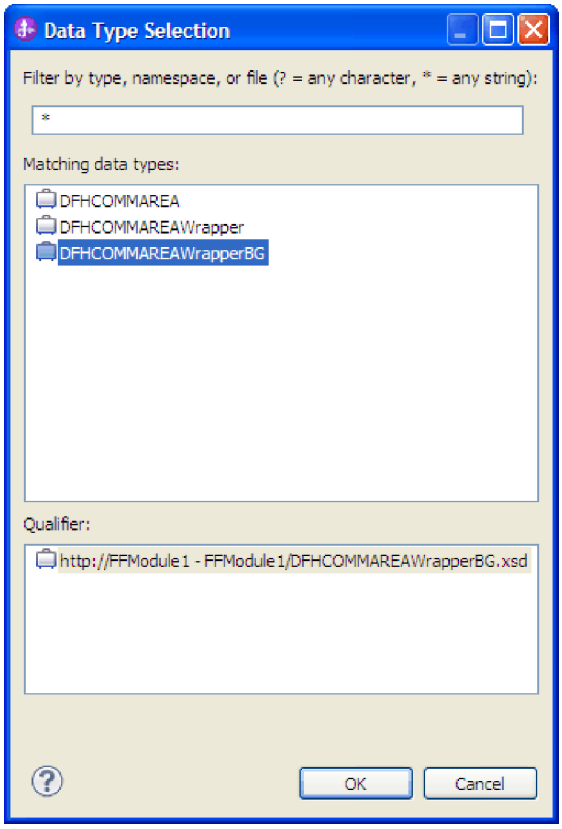


Figure 20. The Data Type Selection window

k. Click **Next** and complete the external service wizard.

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

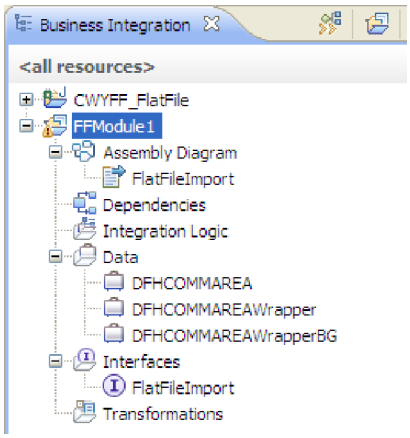


Figure 21. Data bindings used by COBOL copybook, WSDL files, import files, and other artifacts

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 COBOL copybook data binding. This module can be deployed on IBM Business Process Manager or WebSphere Enterprise Service Bus and tested for a 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 service wizard in IBM Integration Designer 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 IBM Integration Designer.
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. It 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 service 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 service wizard to generate a wrapper business object definition from the generated business object.

Procedure

1. Generate the business object definition for the COBOL program source file.
   a. In the Business Integration section of the window, right-click the module and select **New** > **New Business Object From External Data**.
   b. In the Select an Input Source window, select **Languages -> COBOL**. Click **Next**.

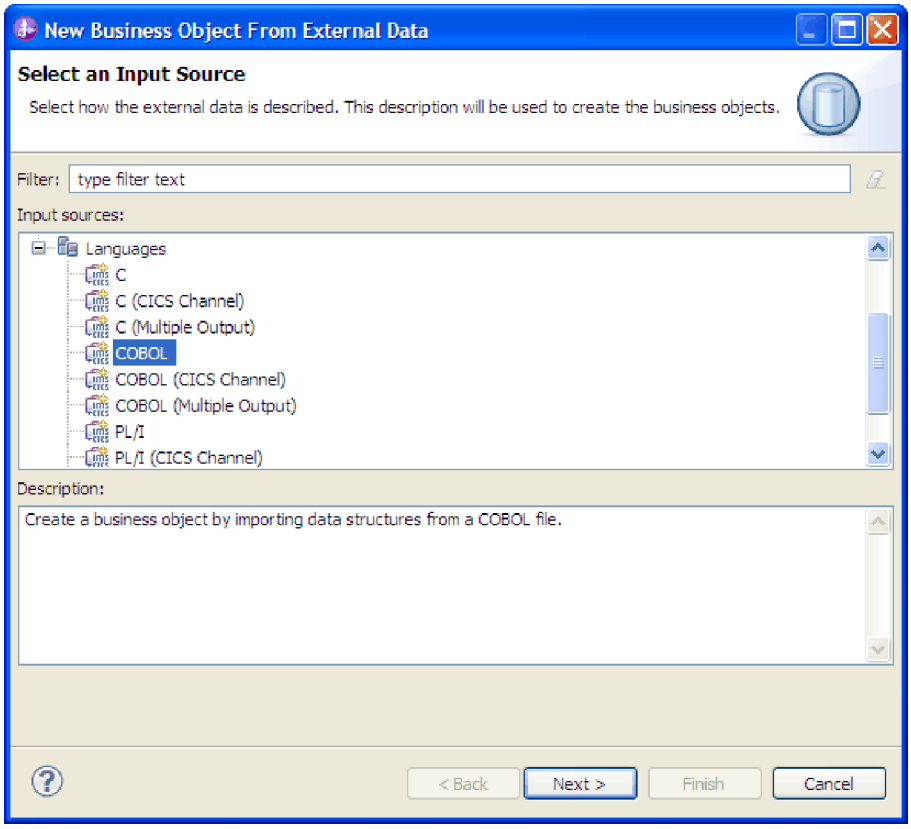


Figure 22. The Select an Input Source window

c. In the Provide Details for the Mapping window, make sure the **Selected mapping** value is **COBOL to Business Object**. Click **Browse** and select the .ccp file. For example, `taderc99.ccp` can be the name of the .ccp file. Click **Next**.

d. In the Select Data Structures window, click **Find**. The new business object, called DFHCOMMAREA is displayed.

e. Select DFHCOMMAREA and click **Next**.

f. In the Generate Business Objects window, specify the **Module**, **Folder**, **Name**. Select **Generate style** from the list and click **Finish**.

A business object, called DFHCOMMAREA in the figure, 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 option to generate wrapper business object definitions is displayed only when the adapter RAR file has been imported into the workspace.

a. In the Business Integration section of the window, right-click the module and select **New** > **New Business Object From External Data**.

b. In the Select an Input Source window, expand **Adapters** and select **Flat File**. Click **Next**.

c. In the Specify the Location window, select the module in the **Module** field and click **Next**.

d. In the Specify the Properties window, click **Browse** and select the business object created in Step 1, for example, DFHCOMMAREA, for the data type.

e. To generate a business graph, select the **Generate business graph for each business object** check box.

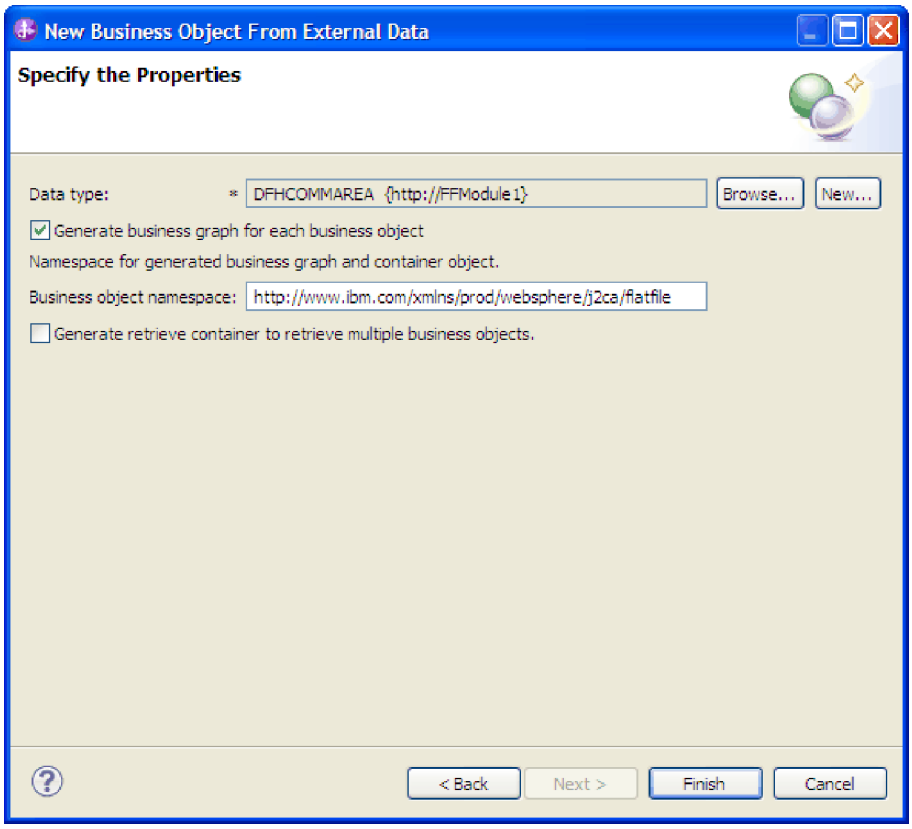


Figure 23. The Specify the Properties window

f. Click **Finish**.

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

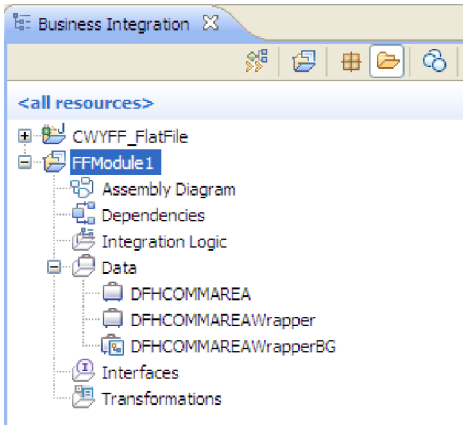


Figure 24. The wrapper business object and the business graph listed in the Business Integration window

3. Generate the required artifacts for the COBOL copybook inbound module.

a. In the Business Integration section of the window, right-click the module and select **New** > **External Service**.
b. Select **Adapters** and click **Next**.
c. In the Select an Adapter window, select **IBM WebSphere Adapter for Flat Files (IBM : 7.5.0.1)** adapter and click the **CWYFF_FlatFile** connector project. Click **Next**.
d. In the Select the Processing Direction window, select **Inbound** and click **Next**.
e. In the **Event directory** field, click **Browse** and select the event directory.
f. From the **Function selector options** list, select the default value.
g. From the **Data format options** list, select the **Use COBOL, C or PL/I data format** option. Click **Next**.

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

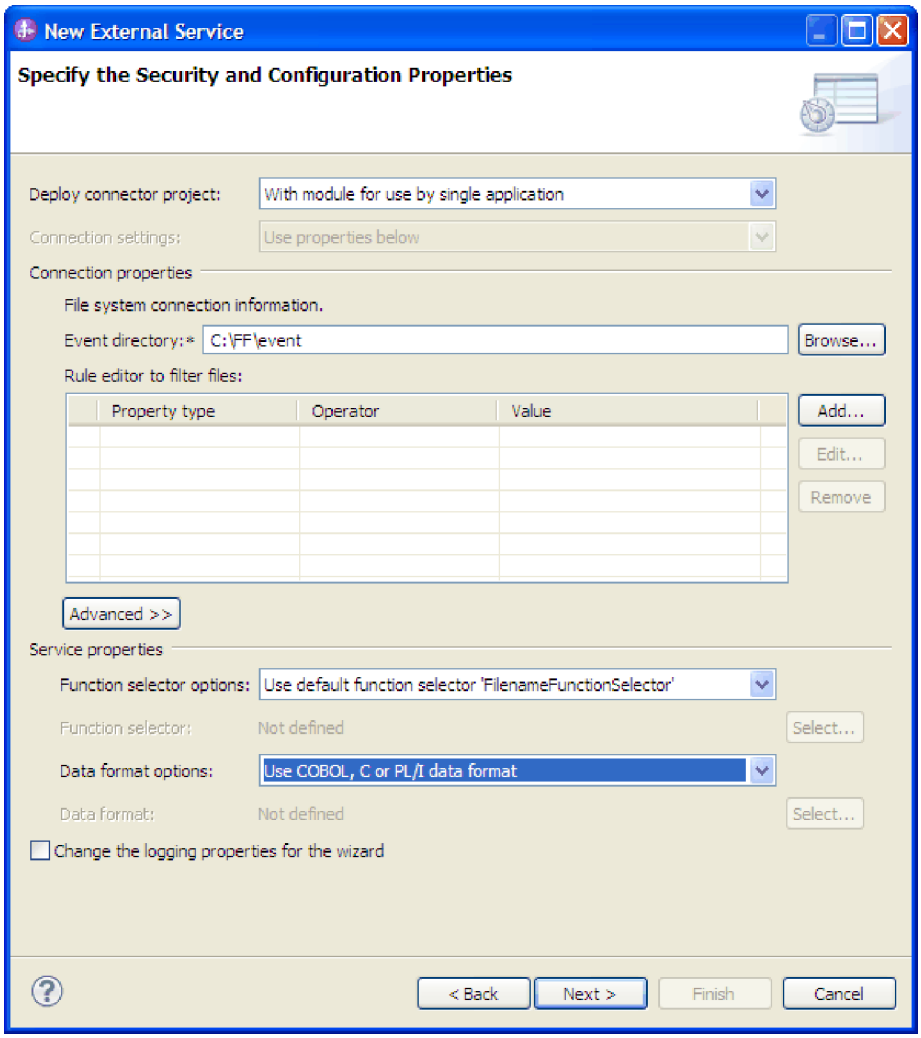


Figure 25. The Specify the Security and Configuration Properties window

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 **Additional configuration**. 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" on page 205.

i. In the Add, Edit, or Remove Operations window, click **Add**.

j. In Specify the I/O Properties window, select **User-defined type** for the **Data type for the operation** field. Click **Next**.

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

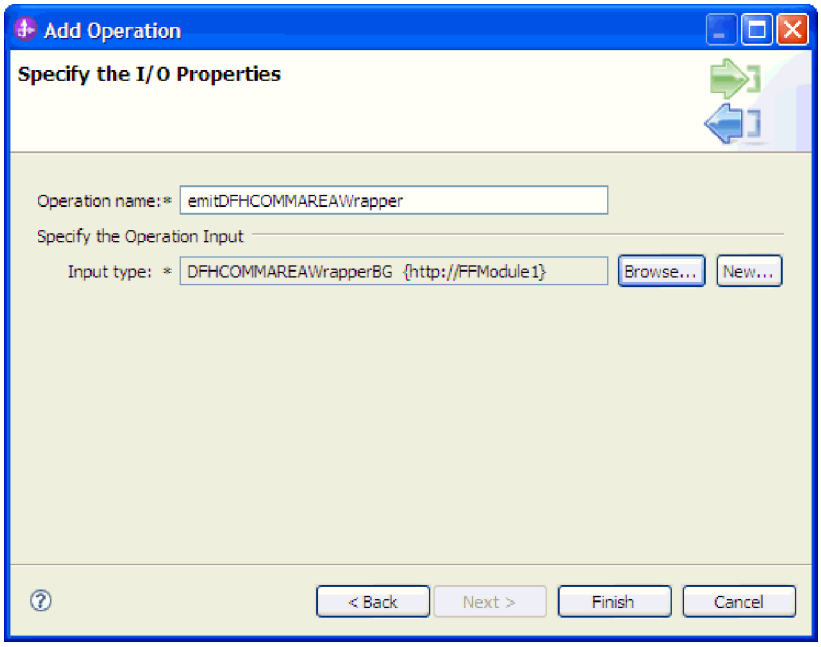


Figure 26. Selecting the input type

l. Click **Finish**.

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.

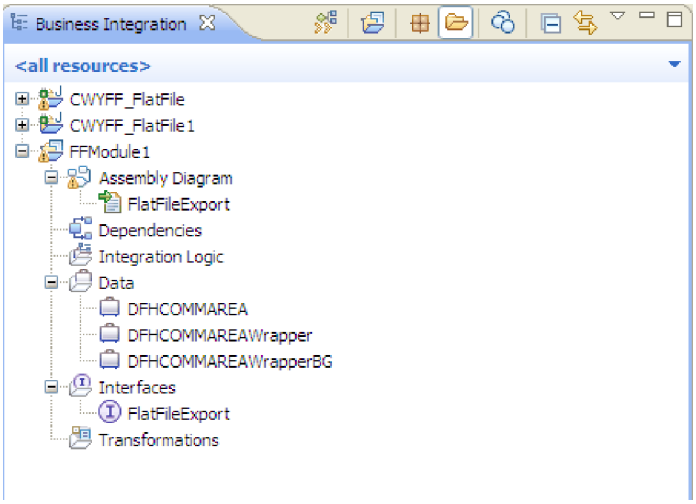


Figure 27. Data bindings used by COBOL copybook, WSDL files, export files, and other artifacts

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 IBM Business Process Manager or WebSphere Enterprise Service Bus and tested for an inbound operation.

What to do next

Deploy the module.

Related reference

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

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 IBM Business Process Manager administrative console.

About this task

The following adapter patterns are available for the WebSphere Adapter for Flat Files:

Table 10.

Adapter pattern	Description
Inbound Flat File pattern	The Flat File inbound pattern creates a service that retrieves a file in a specific directory on the local file system. If the file is not in an XML format, you can specify a data handler that transforms 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 Flat File pattern	The Flat File outbound pattern creates a service that stores data in a file in a specific directory on the local file system. If the required output format is not an XML format, you can specify a data handler that transforms the business object to the file content format.

In this example, we create a Flat File inbound service that receives a file from the file system for processing. The completed service in this example reads in a file and splits 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 IBM Integration Designer window and select **New** > **External Service**. The Select the Service Type or Registry window opens.
2. Select **Simple: Create an inbound Flat File service to read from a local file** and click **Next**.

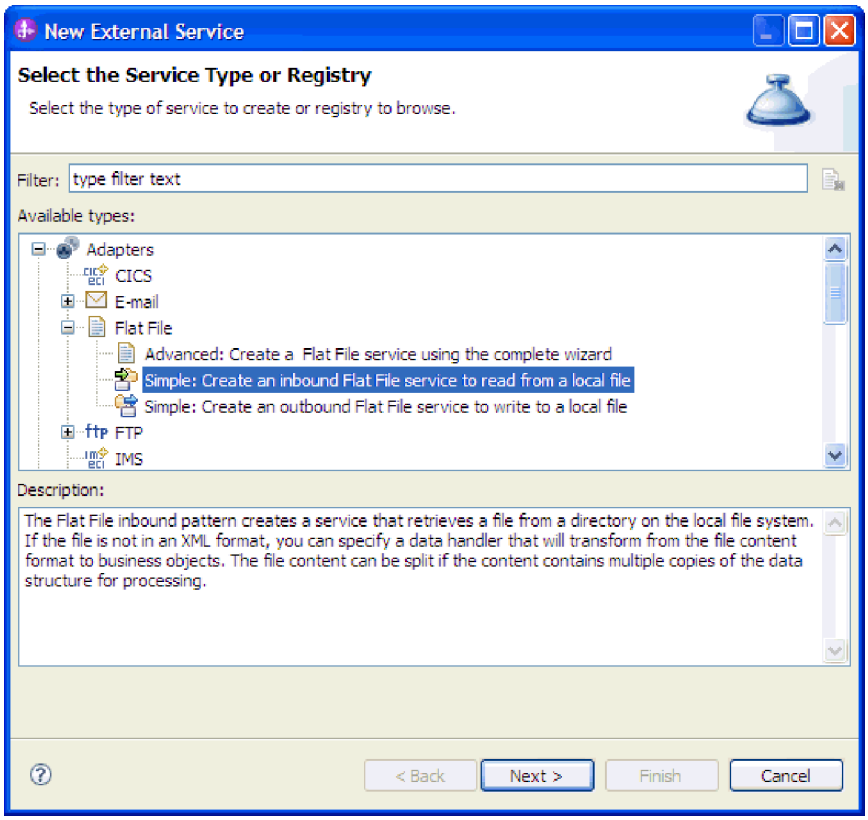


Figure 28. Select the Service Type or Registry window

3. In the Flat File Service window, change the **Name** to something meaningful such as `FlatFileInboundInterface` and click **Next**.
4. In the Business object and directory window, click **Browse** and navigate to the **Customer** business object.
5. Specify the directory where you placed the input file, in this case the `FFInboundEvents` 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: `${FFINBOUNDEVENTS}`.

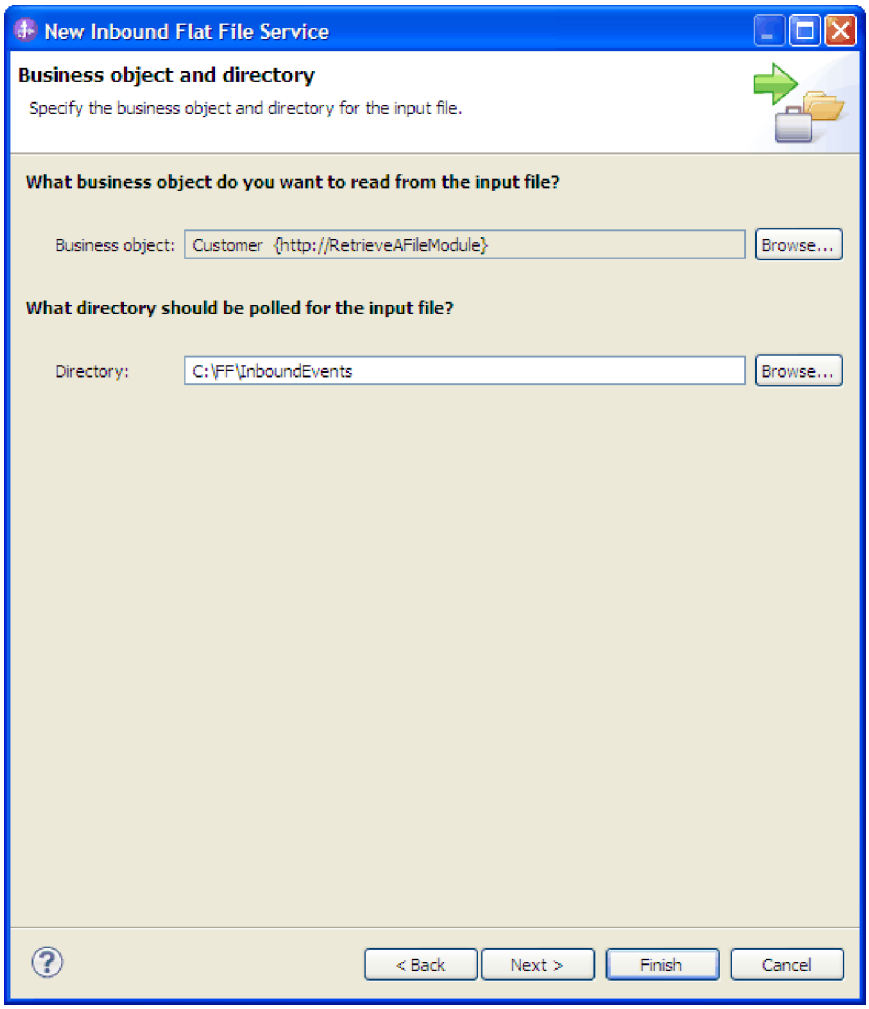


Figure 29. Business object and directory window

6. 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.
7. Select **Split file content by delimiter** and enter your delimiter, which is `####;\r\n` in this example. Click **Next**.

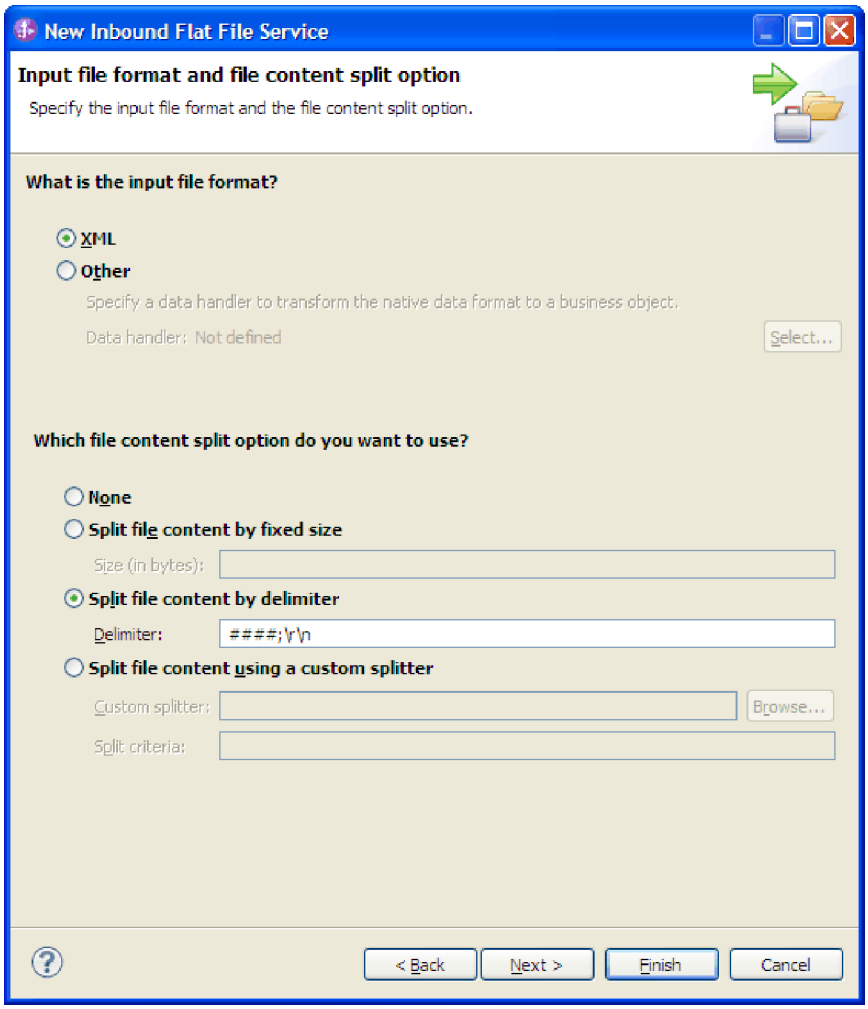


Figure 30. Input file format and file content split option window

8. In the Archive directory and wrapper business object window, specify the **Local archive directory**, which is `FFInboundArchive` 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: `${FFINBOUNDARCHIVE}`. Select **Use a wrapper business object to contain additional input file information** check box, if you want to include the adapter-specific information. Click **Finish**.

Results

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

Table 11.

Artifact	Name	Description
Export	FlatFileInboundInterface	The export exposes the module externally, in this case, to the WebSphere Adapter for Flat Files.

Table 11. (continued)

Artifact	Name	Description
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	FlatFileInboundInterface	This interface contains the operation that can be invoked.
Operation	emitCustomerInput	emitCustomerInput is the only operation in the interface.

Starting the external service wizard

To create and deploy a module, you start the external service wizard in IBM Integration Designer. The wizard creates a project that is used to organize the files associated with the module.

About this task

Start the external service wizard to create a project for the adapter in IBM Integration Designer. 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 IBM Integration Designer, and then click **File** > **New** > **External Service**.
2. In the New external service window, make sure **Adapters** is selected, and click **Next**.
3. From the Select an Enterprise Service Resource Adapter window, create a project or select an existing project.
   - To create a project, perform the following steps:
     a. Select **IBM WebSphere Adapter for Flat Files** and click **Next**.
     b. In the Connector Import window, provide another name for the project (to use a name other than **CWYFF_FlatFile**), select the server (for example, **IBM Business Process Manager**) and click **Next**.
   - To select an existing project, perform the following steps:
     a. Expand **IBM WebSphere Adapter for Flat Files**.
     b. Select a project.

        For example, if you have an existing project named CWYFF_FlatFiles, you can expand **IBM WebSphere Adapter for Flat Files** and select **CWYFF_FlatFile**.
     c. Click **Next**.

Results

A new project is created and is listed in the Business Integration window.

What to do next

Configuring the module for outbound processing

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

Related concepts

"Outbound processing" on page 3
During outbound processing, the adapter receives a request from the module, in the form of a business object, to perform an operation on a file in the local file system. The adapter performs the requested operation and returns a business object, if applicable, that represents the result of the operation to the component.

Setting deployment and runtime properties

After you have decided whether your module is to be used for outbound or inbound communication with the local file system, you must configure the managed connection factory properties so that the adapter can make the connection between the module and the local file system.

Before you begin

Before you can set the properties in this section, you must have created your adapter module. It is displayed in IBM Integration Designer below the adapter project. For more information about creating the adapter project, see "Starting the external service wizard" on page 83.

About this task

To set deployment and runtime properties, follow this procedure. For more information about the properties in this topic, see "Managed connection factory properties" on page 172.

Procedure

1. In the Select the Processing Direction window, select **Outbound** and click **Next**.

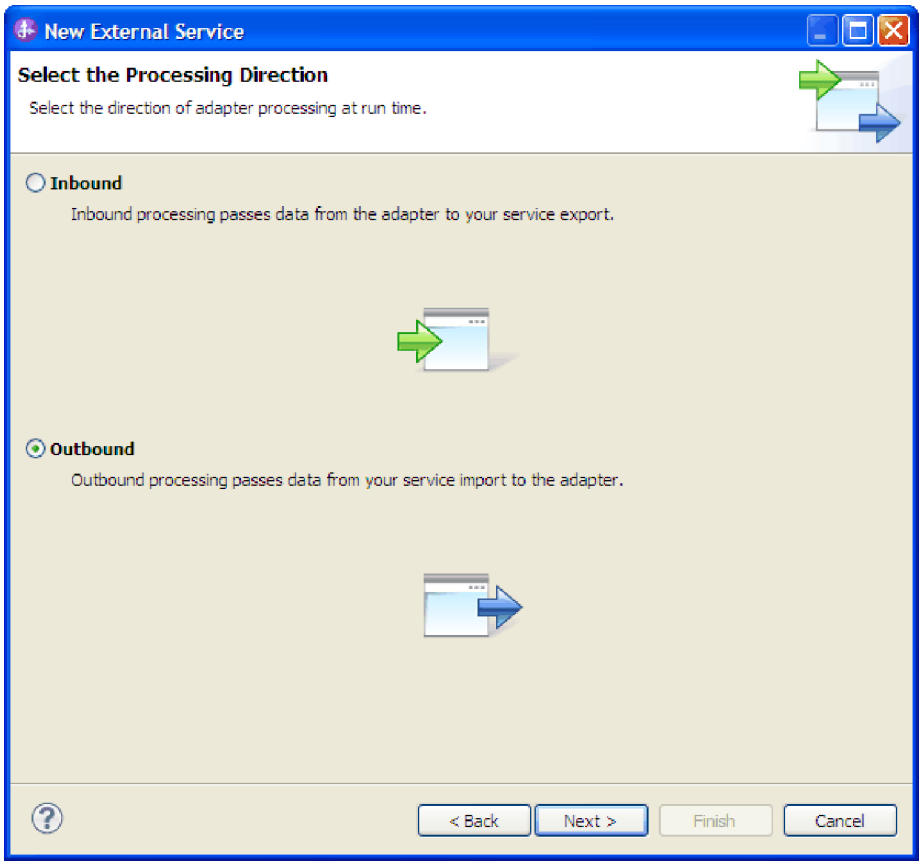


Figure 31. Selecting inbound or outbound processing in the external service wizard

2. In the Specify the Security and Configuration Properties window, in the **Deploy connector project** field, select **With module for use by single application**.
3. Define the Connection properties for your module. For more details on the properties found on this window, see the "Managed connection factory properties" on page 172 topic.

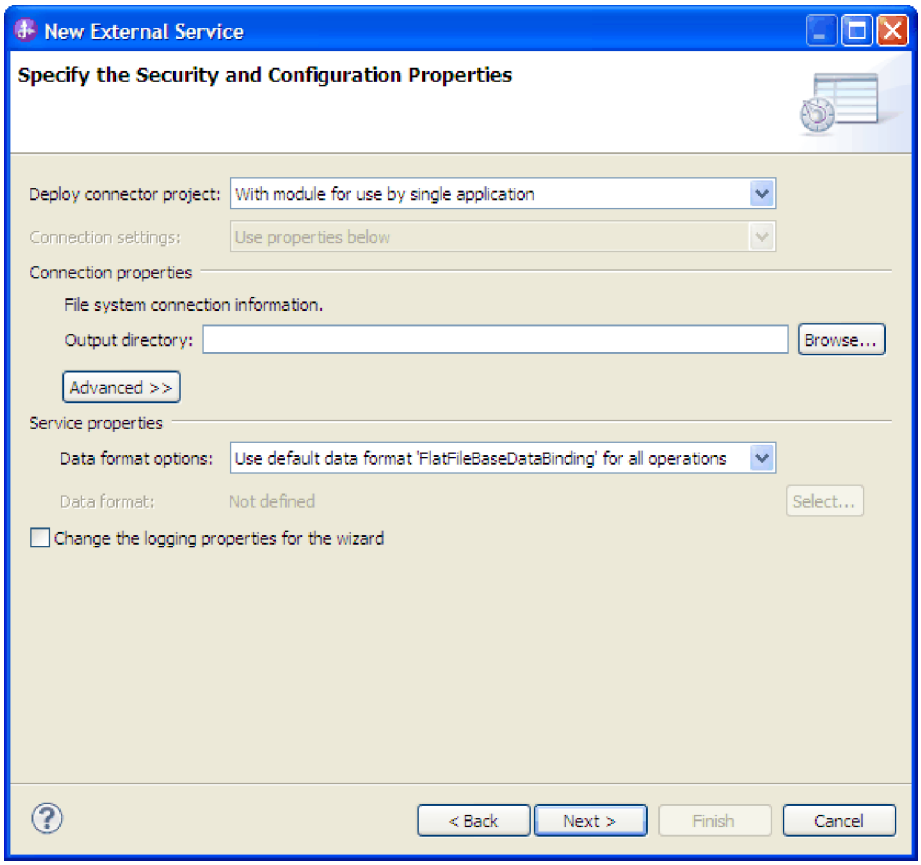


Figure 32. Setting the security and configuration properties

4. In the **Output directory** field, specify the directory that is to be used for all the outbound operations. For example, the directory where the file is to be created during the Create operation.
5. Click **Advanced** and expand the **Logging and tracing**, **Additional configuration**, and **Bidi properties** sections to specify additional properties.
   - **Logging and tracing**
     a. If you have multiple instances of the adapter, enter a value in the **Adapter ID** that is unique for this instance. For more information, see "Adapter ID (AdapterID)" on page 173.
     b. You can select the **Disguise user data as 'XXX' in log and trace files** check box 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) " on page 174.
   - **Additional configuration**
     a. In the **Default target file name** field, specify the name of the file that is created in the output directory, or a WebSphere Application Server environment variable that represents this file. For more information, see "Default target file name" on page 174.
     b. In the **Staging directory** field, specify the full path name of the temporary directory where the adapter writes the initial output files for Create and Overwrite operations during outbound processing, or aWebSphere Application Server environment variable that represents this directory. For more information, see "Staging directory" on page 176.

c. In the **Sequence file** field, specify the full path name of the file where sequences are stored during outbound Create operations, or a WebSphere Application Server environment variable that represents this file. For more information, see "Sequence file" on page 175.

d. Select the **Generate a unique file** check box to enable the adapter to generate unique file names. For more information, see "Generate a unique file" on page 183.

   – In the **Prefix for the unique file** field, specify a predefined prefix to the unique file name. For more information, see "Prefix for the unique file name" on page 175.

   – In the **Suffix (extension) for the unique file** field, specify a predefined extension to the unique file name. For more information, see "Suffix for the unique file name" on page 177.

- **Bidi properties**

  a. Select the **Bidi transformation** check box to specify bidirectional format. For more information about setting the **Bidi properties**, refer to the "Globalization" on page 213 section.

6. Optional: If you want to specify the log file output location or define the level of logging for this module, select the **Change logging properties for wizard** check box. For information about setting logging levels, see the section about configuring logging properties in the Chapter 8, "Troubleshooting and support," on page 145 topic.
7. Click **Next**.

Results

The adapter saves the connection properties.

What to do next

Select a data type for the module and name the operation associated with the chosen data type.

Related reference

"Connection properties for the wizard" on page 169
Connection properties are used to build a service description and save the built-in artifacts. These properties are configured in the external service wizard.

Selecting the operation and data type

Use the external service wizard to select the outbound operation to access functions on the local file system and the data type to be used with it. The operations supported are Create, Append, Overwrite, Delete, Exists, List, and Retrieve. The external service wizard gives you a choice of three data types: generic FlatFile business object, generic FlatFile business object with business graph, and user-defined type. 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 local file system before you can complete the steps given here.

About this task

To select an outbound operation and the data type to be used with it, follow this procedure.

Procedure

1. In the Add, Edit, or Remove Operations window, click **Add** to create an operation.
2. In the Specify the I/O Properties window, select an outbound operation from the **Operation kind** list.
3. In the **Data type for the operation** list, select a data type. Click **Next**. In this example, the user-defined data type is selected. For Delete, Retrieve, Exists, and List operations, only the generic data type (generic FlatFile business object or generic FlatFile business object with business graph) is supported as input. If you select user-defined type with one of these operations, you must provide a user-defined data binding to support it.

   For Create, Append, and Overwrite operations, the choices are user-defined type, generic FlatFile business object, and generic FlatFile business object with business graph. For more information about data types, see, "Business object structures" on page 159.
4. Optional: Select the **Enable response type for the operation** check box for Create, Append, and Overwrite operations, if you want the file name returned or if you are generating a unique file name or have enabled file sequencing. By default, the **Enable response type for the operation** check box is selected for the Exists, List, and Retrieve operations because output is required for these operations. For the Delete operation, select the **Enable response type for the operation** check box if you want a value of `true` returned when the operation is successful. Click **Next**.

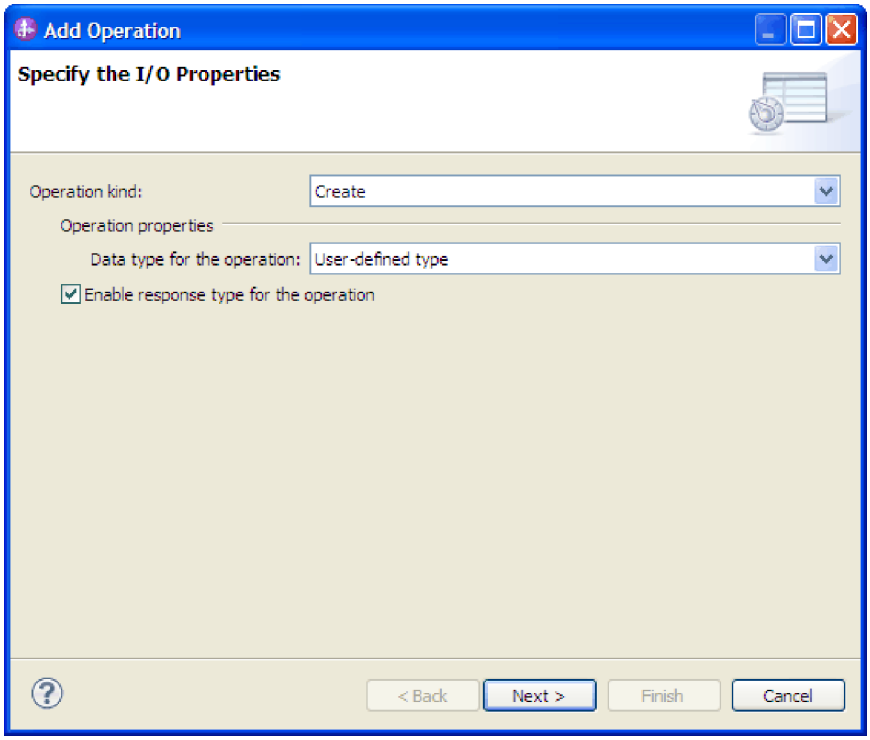


Figure 33. Naming the operation and specifying the input data type

5. In the Specify the I/O Properties window, type an **Operation name**. Name the operation something meaningful.

   Note: Names cannot contain spaces.

   By default, the data type for the output is specified as CreateResponse or CreateResponseBG.
6. Select the Input type. Click **Browse** and select the business object you created earlier.

   Note: If you specified a generic data type such as generic FlatFile business object or generic FlatFile business object with business graph, the **Input type** is specified by default to FlatFile or FlatFileBG.

Results

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

What to do next

Add and configure a data binding to be used with the module.

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 the 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 the file with the information it receives in the business object.

Before you begin

You must have selected an operation and the data type to be used with it.

About this task

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

Note: Data bindings can be configured before running the external service wizard using IBM Integration Designer. To do this configuration, select **New** > **Configure Binding Resource** in IBM Integration Designer and complete the data binding windows described in this documentation.

Procedure

1. In the Specify the I/O Properties window, select either the **Use suggested data format 'FlatFileBaseDataBinding'** or the **Use a data format configuration** option from the **Data format options** list.
2. If you select the **Use a data format configuration** option, click **Select** to configure the data binding.
3. In the Select a Data Format Transformation window, select the **Use existing data format transformation from the list** option. From the list, select **FlatFileBaseDataBinding**. To configure a custom data binding, select the **Select your custom data format transformation from the workspace** option. Click **Next**.

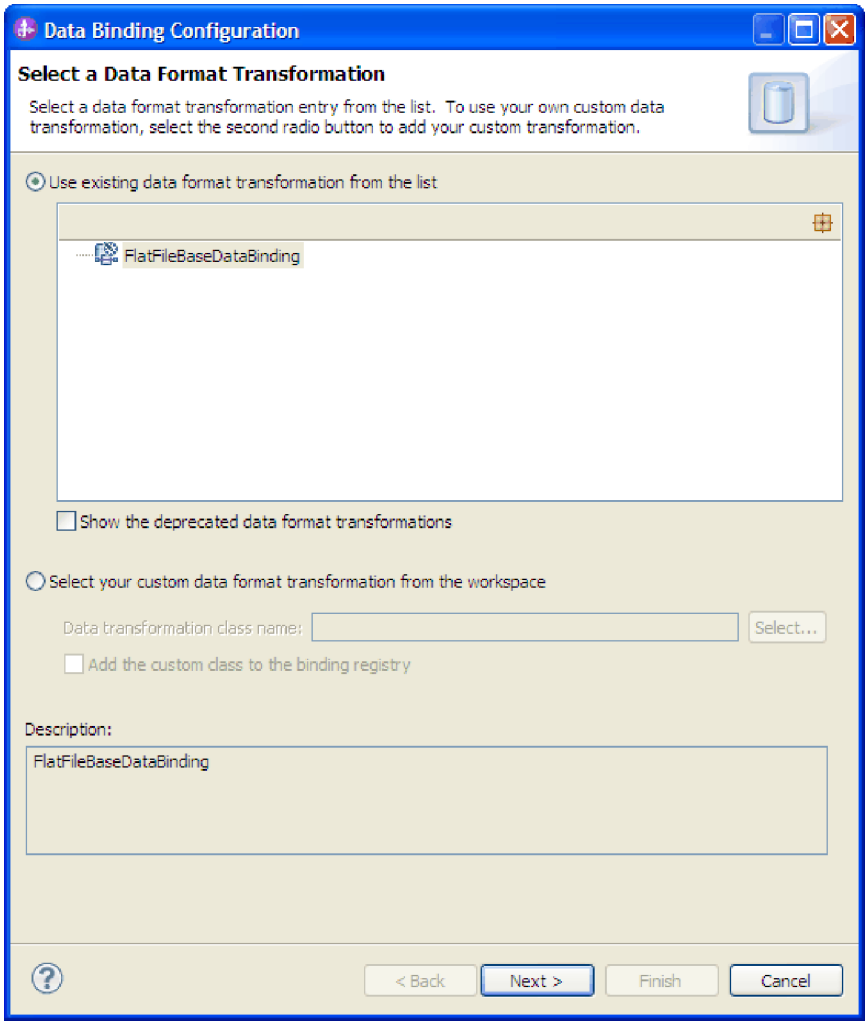


Figure 34. Selecting the data binding

4. In Specify the Data Transformation Properties window, click **Next**.

   Note: This window is used for configuring the data handlers.
5. In Configure a New Data Transformation window, provide the data binding configuration details.
   a. In the Configure a New Data Transformation window, the **Module** defaults to the module name you typed earlier in the external service. If you want to create a data binding for a different module, select **New** to create a module.
   b. If you want to select a new folder for the artifact, click **Browse** and select a new folder location.
   c. Type a **Name** for the data binding configuration. Click **Finish**. The data binding name is populated in the Specify the I/O Properties window.

Results

A data binding is configured for use with the module.

What to do next

Select the data handler configuration.

Configuring data handlers

Data handlers perform the conversions between a business object and a native format. The configuration in this topic is shown using the XML data handler. For comma-separated values (CSV) file format files, you must select the Delimited data handler.

Before you begin

You must have created a data binding before you specify the data handlers for the module. Also, you must have predefined business objects using IBM Integration Designer Business Object Editor. If you stop the wizard here to create business objects, you must start the wizard steps from the beginning.

Note: You can configure data handlers before running the external service wizard using IBM Integration Designer. To do this configuration, select **New** > **Configure Binding Resource** in IBM Integration Designer and complete the data handler windows described in this documentation.

About this task

To specify data handlers, follow this procedure.

Procedure

1. In the Specify the Data Transformation Properties window, ensure that **DataHandler** is selected in the **Binding type** field.
2. In the **Configured data handler** field, click **Select**. Complete the following steps to create and configure a data handler.
   a. In the Select a Data Format Transformation window, click **Use existing data format transformation from the list** option. Select **XML** data handler from the list and click **Next**. To work with data files (CSV) that uses comma to separate the fields of data, select **Delimited**. For more information about working with delimited data files, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.wid.integ.doc/topics/rdelimitedcvs.html.

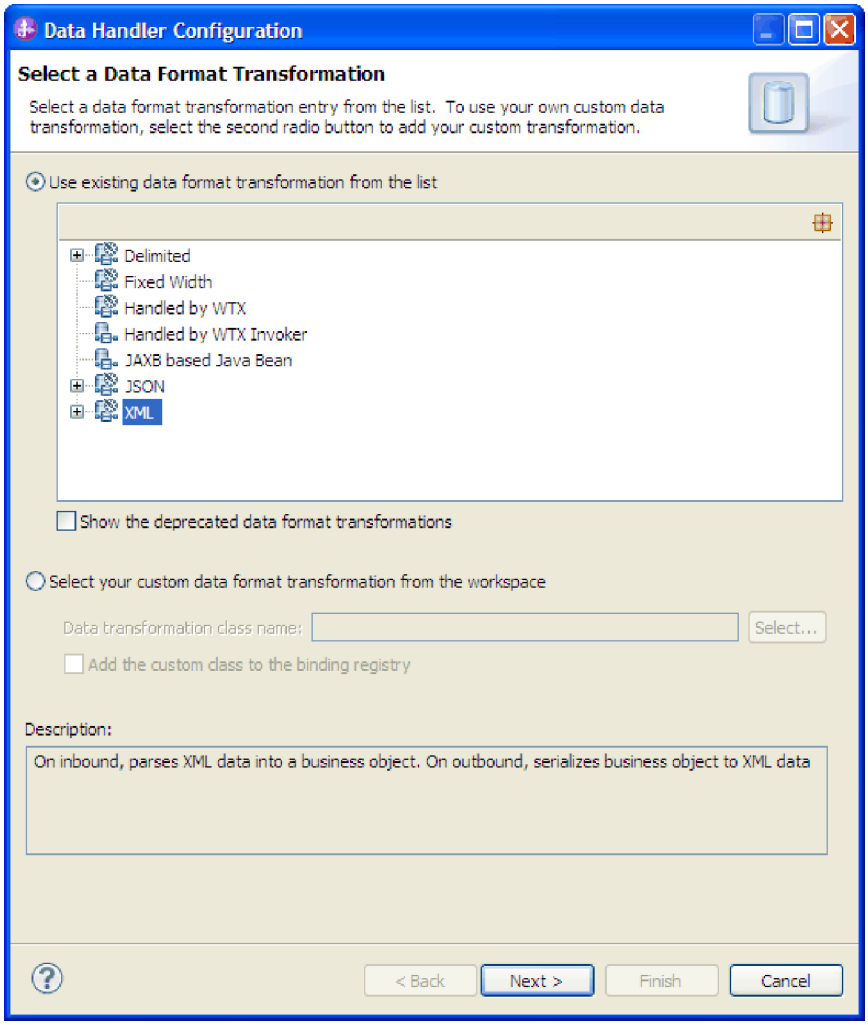


Figure 35. Creating a data handler configuration

3. In the Specify the Data Transformation Properties window, specify the **Encoding**. The default is UTF-8. Click **Next**.

   Note: If **Delimited** option is selected, the Specify the Data Transformation Properties window provides the delimiter data handler configuration fields.

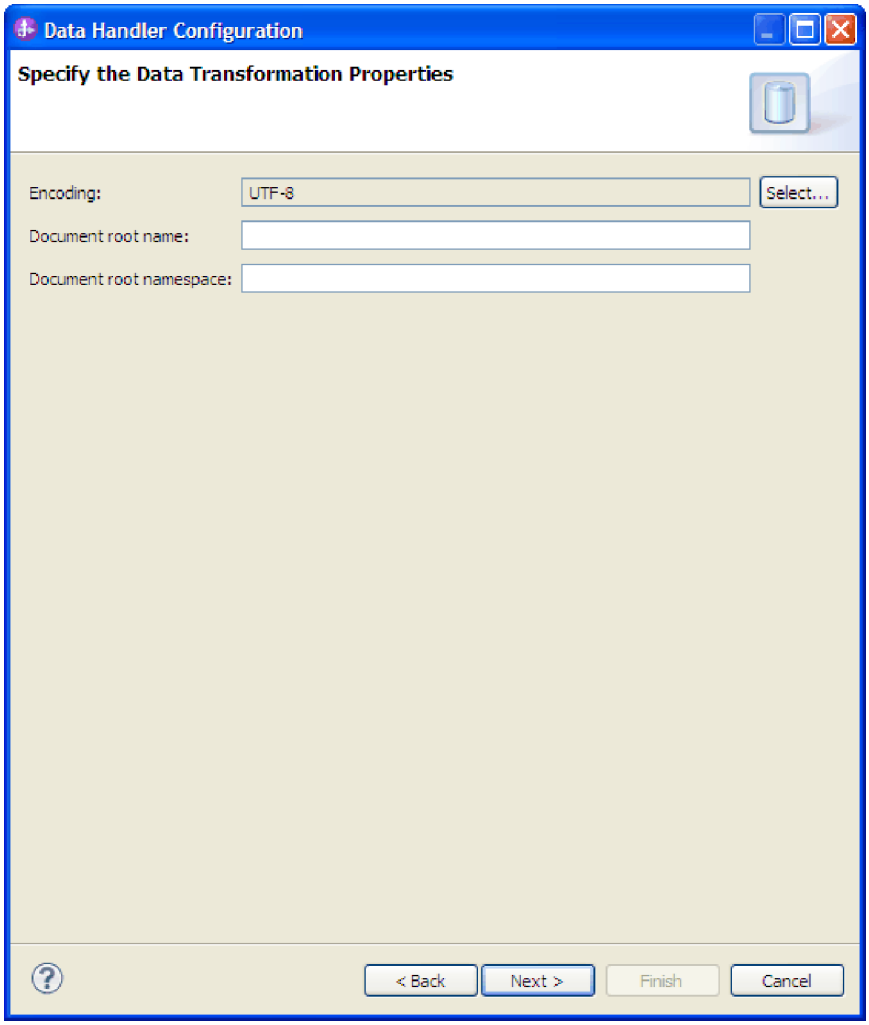


Figure 36. Specifying the encoding for the data handler configuration

4. In the Configure a New Data Transformation window, specify the **Module**, **Namespace**, **Folder**, and **Name** for the data handler configuration. Click **Finish**.

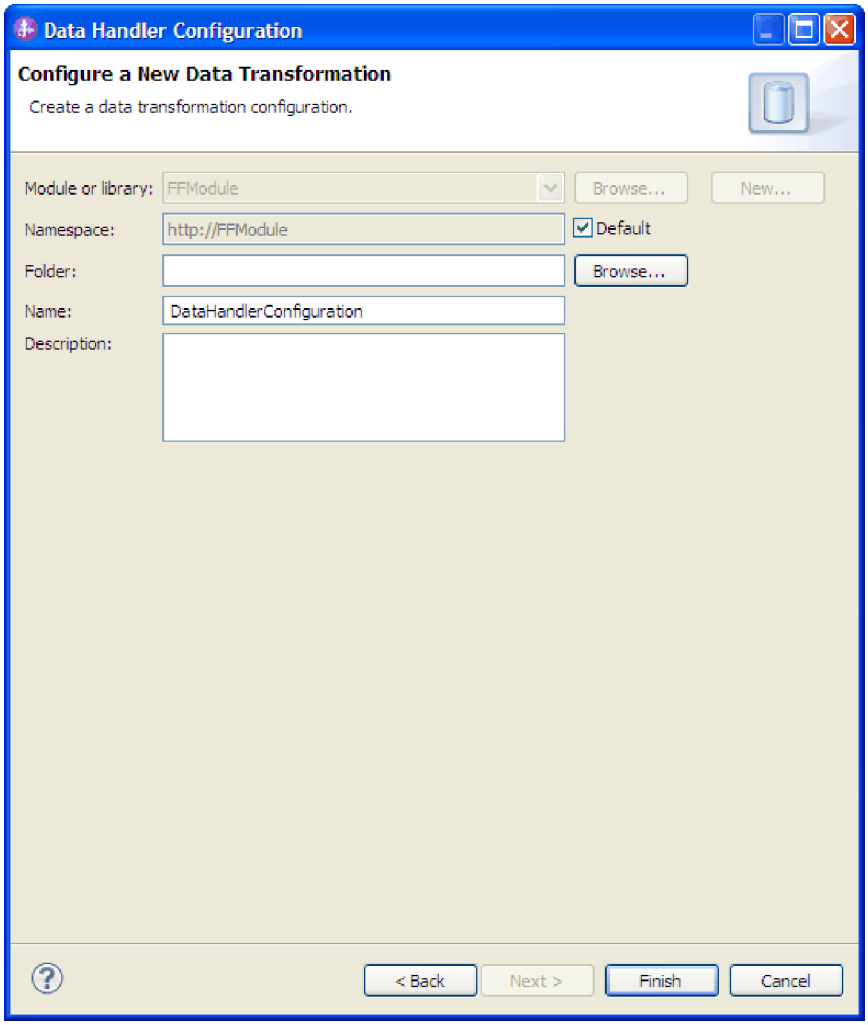


Figure 37. Specifying a name for the data handler configuration

5. Click **Finish**. The configured data handler is populated in the Specify the Data Transformation Properties window.

Results

Data handlers are created.

What to do next

Specify interaction specification properties and generate artifacts for the module.

Setting interaction properties and generating the service

Interaction properties are optional. If you choose to set them, the values you specify appear as defaults in all parent business objects generated by the external service wizard. While creating artifacts for the module, the adapter generates an import file. The import file contains the operation for the top-level business object.

Before you begin

To set interaction specification properties and generate artifacts for your module, you must have already configured data bindings and selected business objects.

About this task

To set interaction specification properties and generate artifacts, follow this procedure. For more information about interaction specification properties, see the “Interaction specification properties” on page 180 topic.

Procedure

1. In the Add, Edit, or Remove Operations window, click **Advanced**.
2. In the **Additional configuration** section, type values for different interaction specification properties. Click **Next**.

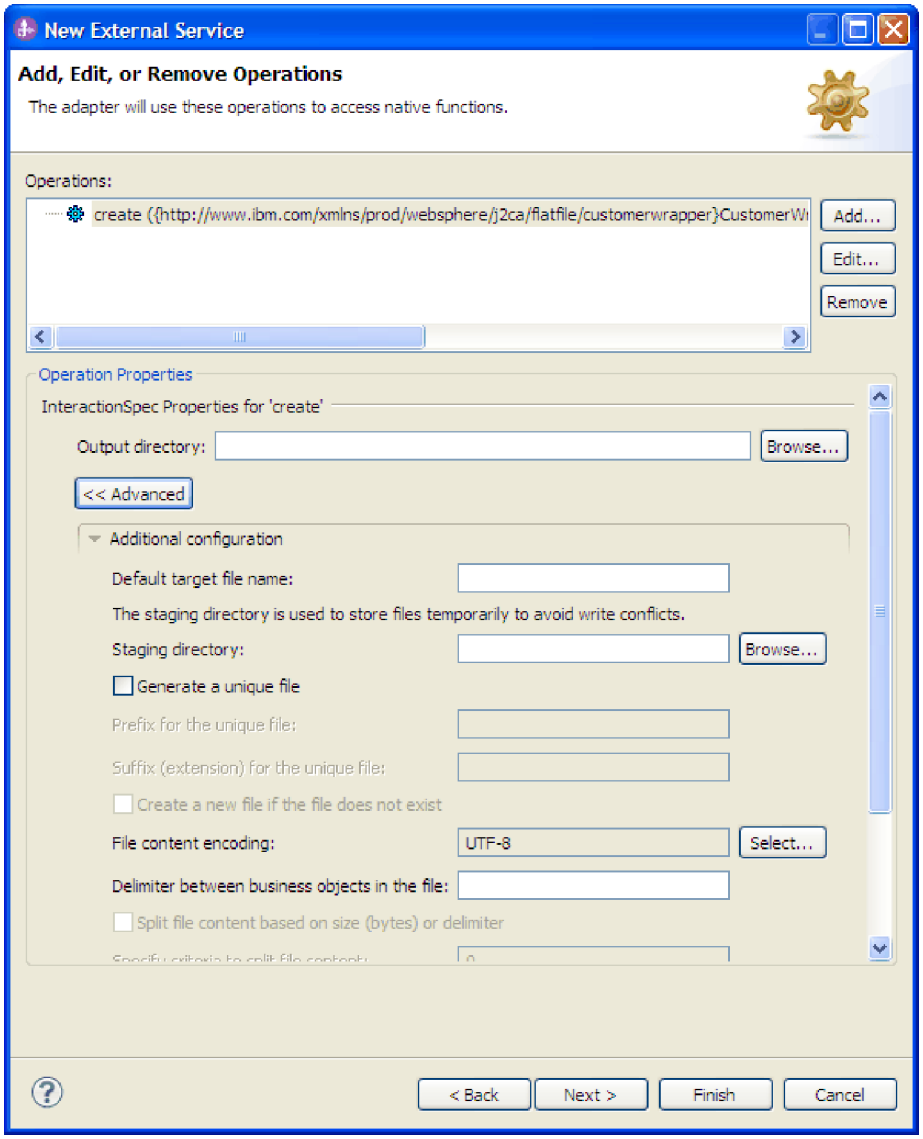


Figure 38. Naming the service

3. In the Operations window, click **Next**. In the Specify the Name and Location window, type a name for the interface. This name displays in the IBM Integration Designer assembly diagram. Click **Finish**.

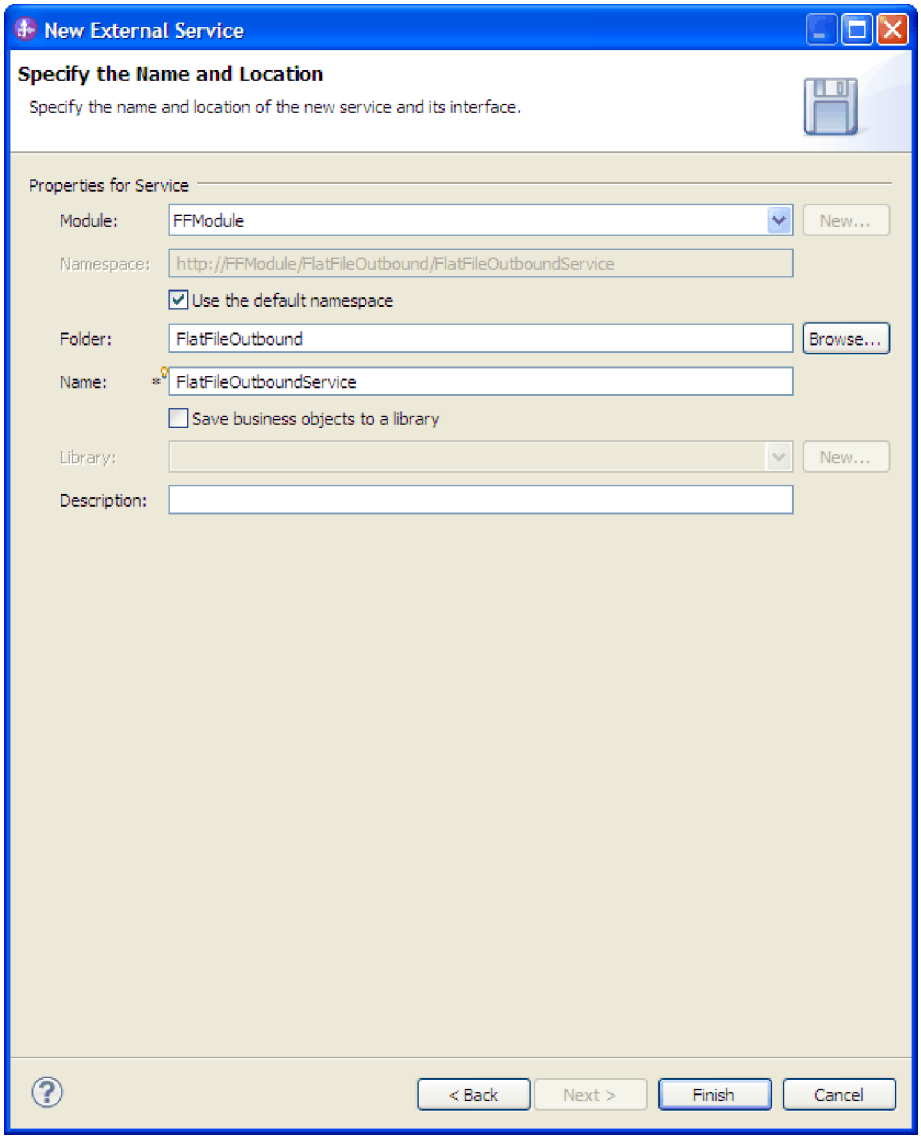


Figure 39. Naming the service

4.

Results

The IBM Integration Designer generates the service and an import. The outbound artifacts that are created are visible in the IBM Integration Designer Project Explorer under your module.

What to do next

Deploy the module.

Related reference

"Outbound configuration properties" on page 167
IBM WebSphere Adapter for Flat Files 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 IBM Business Process Manager or WebSphere Enterprise Service Bus using IBM Integration Designer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

"Inbound configuration properties" on page 186
WebSphere Adapter for Flat Files 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 IBM Integration Designer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

"Globalization" on page 213
WebSphere Adapter for Flat Files 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.

Configuring the module for inbound processing

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

Related concepts

"Inbound processing" on page 13
WebSphere Adapter for Flat Files supports inbound event processing. It polls the local file system at specified intervals for events, such as the creation of a file. When it detects an event, it converts the event data into a business object and sends it to the module for processing.

Related reference

"Custom file splitting" on page 165
You can implement a custom class containing the splitting logic. The adapter provides a Java interface for the class. The version 7.5 of the adapter provides interfaces for both outbound and inbound processes.

Setting deployment and runtime properties

After you have decided whether your module is to be used for outbound or inbound communication with the enterprise information system (local file system), you must configure the activation specification properties, which hold the inbound event processing configuration information for the export.

Before you begin

Before you can set the properties in this section, you must have created your adapter module. It is displayed in IBM Integration Designer below the adapter project. For more information about creating the adapter project, see "Starting the external service wizard" on page 83.

About this task

To set the activation specification properties, follow this procedure. For more information about the properties in this topic, see "Activation specification properties" on page 191.

Procedure

1. In the Select the Processing Direction window, select **Inbound** and click **Next**.

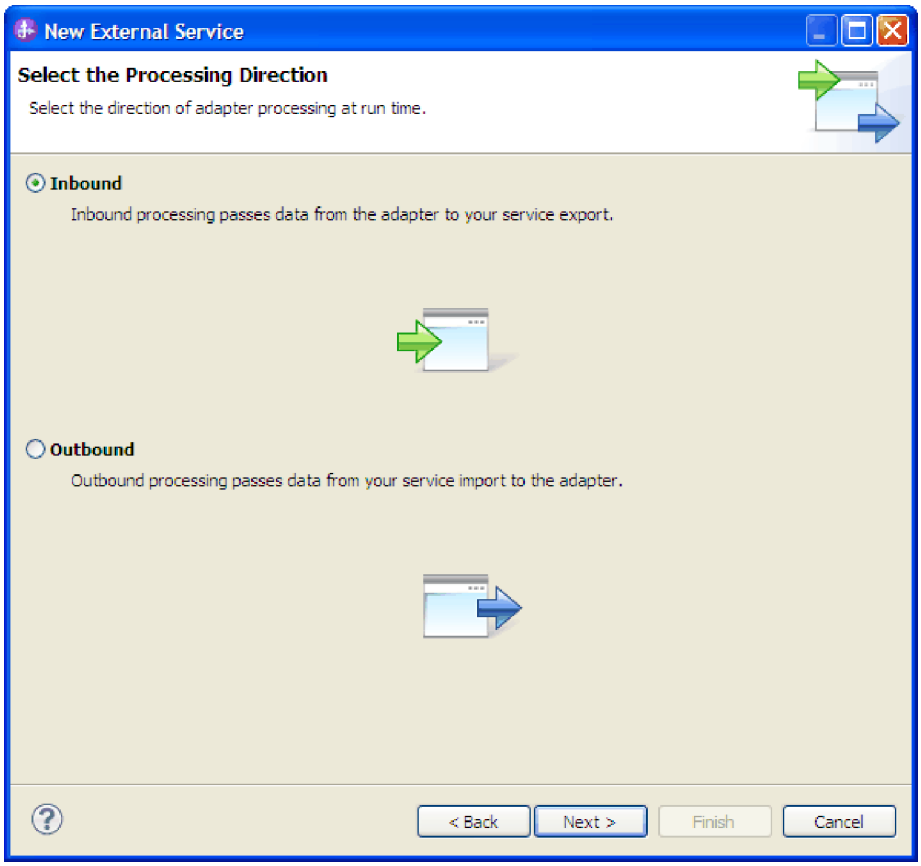


Figure 40. Selecting inbound or outbound in the external service wizard

2. In the Specify the Security and Configuration Properties window, in the **Deploy connector project** field, select **With module for use by single application**.
3. In the Specify the Security and Configuration Properties window, define the activation specification properties for your module. For more details on the properties found on this window, see "Activation specification properties" on page 191.

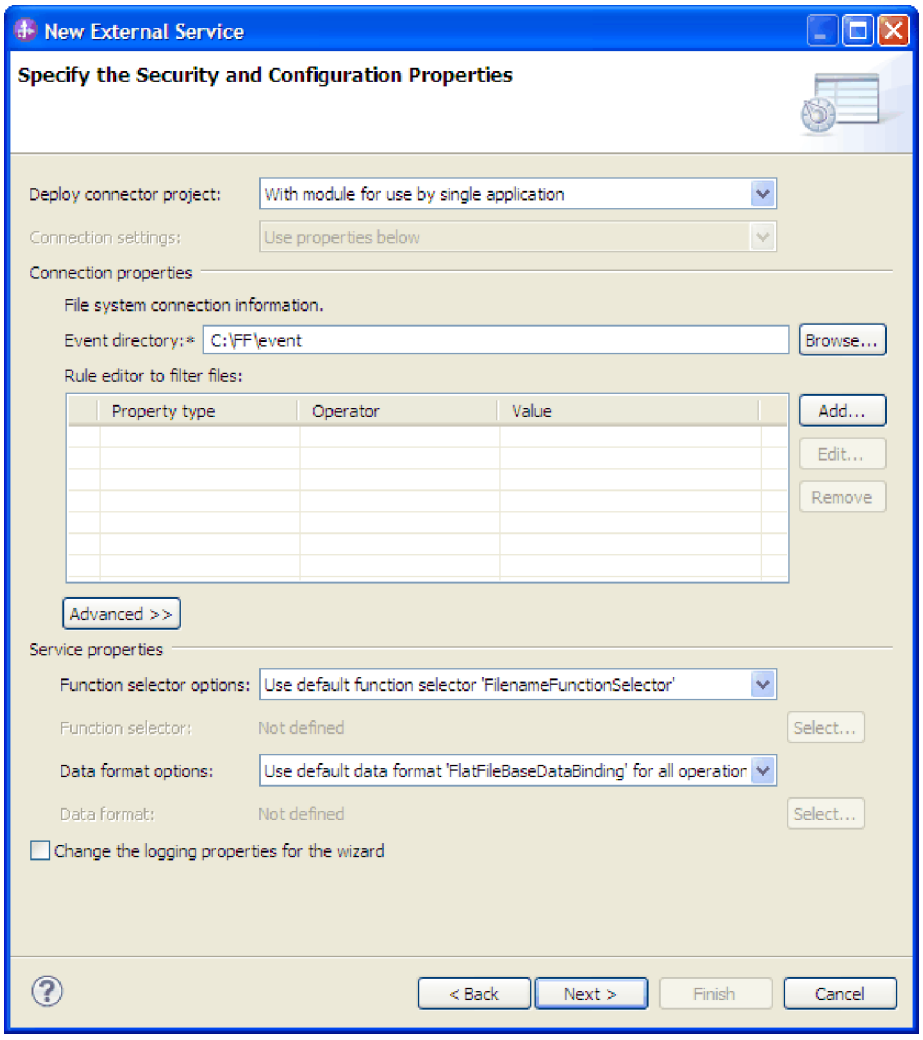


Figure 41. Setting the connection properties

4. In the **Event directory** field, specify the directory in the local file system where the event files are stored.
5. Click **Advanced** and expand the **Event polling configuration**, **Event delivery configuration**, **Event persistence configuration**, **Additional configurationCluster deployment settings**, **File archiving configuration**, **Bidi properties**, and **Logging and tracing** sections to specify values as needed.
   - **Event polling configuration**
     a. In the **Interval between polling periods (milliseconds)** field, type the time in milliseconds, that the adapter waits between polling periods. For more information about the property, see "Interval between polling periods (PollPeriod)" on page 201.
     b. In the **Maximum events in polling period** field, type the number of events to deliver in each polling period. For more information about the property, see "Maximum events in polling period (PollQuantity)" on page 201.
     c. In the **Time between retries in case of case of system connection failure (in milliseconds)** field, type the time in milliseconds, that the adapter

waits before trying to connect after a connection failure during polling. For more information, see “Retry interval if connection fails (RetryInterval)” on page 205.

d. In the **Maximum number of retries in case of system connection failure** field, type the number of times the adapter retries the connection before reporting a polling error. For more information, see “Number of times to retry the system connection (RetryLimit)” on page 202.

e. If you want the adapter to stop if polling errors occur, select the **Stop the adapter when an error is encountered while polling** check box. 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 207.

f. Select **Retry EIS connection on startup**. If you select this property, the adapter continues trying to connect to a system to which it failed to connect when starting. For more information, see “Retry EIS connection on startup (RetryConnectionOnStartup)” on page 205.

g. Select the calendar based scheduling option to create calendar based polling for inbound activities. You can schedule your business activities, when you create a new calendar in IBM Integration Designer. The option of working with the calendar based scheduling feature is only possible with IBM Integration Designer as the tooling environment.

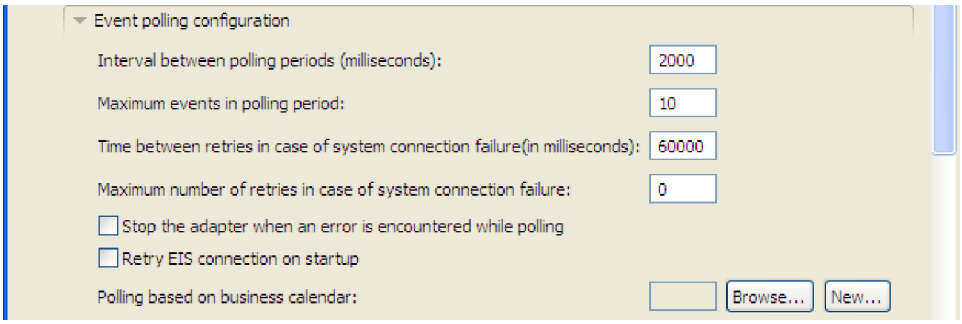


Figure 42. Polling based on calendar

You can either select a blank calendar or create a new calendar for a module or library. When you select a blank calendar, you will not be able to set pre-defined time intervals. You have to define your time intervals. When you create a calendar using a pre-defined template, you can define time intervals for each template.

1) Click **New** to create a new calendar entry for a module or library.

   You can choose an existing calendar, or create a new calendar instance.

   – Click **Browse** to select an existing calendar module. Or click **New** to create a module for the new calendar.
   – Click **Browse** to choose a folder for the calendar. (Optional).
   – Enter a name for the new calendar.
   – Click **Next** if you want to generate the calendar, through a predefined template. Or, click **Finish**, to create a non template calendar.

2) Click **Browse** to select an existing calendar for a module or library. In the **Select a Business Calendar** screen you can search for all currently existing calendar files (`*cal`) in the IBM Integration Designer workspace.
   - In the **Name** field, type the calendar name or click the calendar in the **Matching business calendars** screen. Click **OK** to open the external service wizard.
   - In the **WebSphere Integration workspace**, select the **Calendar** module, and browse **Integration logic->Calendars**, to view or modify the calendar schedules. You can modify the intervals and exceptions, or add new entries for these elements. For more information, refer to the information at http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/index.jsp?topic=/com.ibm.wbpm.wid.bpel.doc/topics/cbuscal.html.

     Note: You must deploy the Business Calendar module to the same IBM Business Process Manager or WebSphere Enterprise Service Bus instance, along with the inbound application. If you do not map these two connections to the same server instance, the inbound application using the business calendar will by default, poll as there is no calendar configured.

h. In the **FlatFile specific polling properties** section, you can retrieve a file by using one of the following properties. When any one of the properties is selected, the other property is automatically disabled.
   - In the **Time interval for polling unchanged files (in milliseconds)** field, you can specify if the adapter retrieves only those files that are not changed during the specified time interval. For more information, see "Time interval for polling unchanged files (in milliseconds)" on page 208.
   - Select **Poll event files for modified content** check box to specify if the adapter polls the files that have changed since the last recorded time stamp. For more information, see "Poll event files for modified content" on page 202.

     Note: This property is disabled if you select the **Pass only file name and directory, not the content** property.
   - Select the **Include only the newly appended content** check box to deliver the appended file contents to the endpoint. This property is enabled when you select the **Poll event files for modified content** check box. For more information, see "Include only the newly appended content" on page 199.

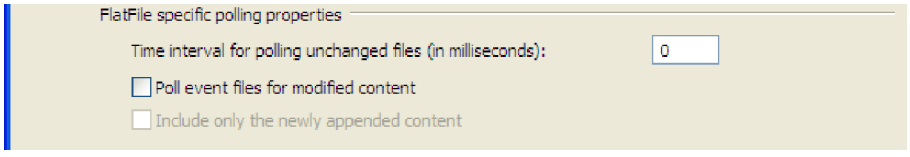


Figure 43. Configuring the polling properties

- **Event delivery configuration**

  a. In the **Type of delivery** field, select the delivery method. The methods are described in "Delivery type (DeliveryType)" on page 196.

Note: The HA Active-Active configuration supports only unordered delivery type events. If the delivery type specified is `ORDERED`, then the adapter throws a runtime exception error.

b. If you want to ensure that events are delivered only once and to only one export, select the **Ensure assured-once event delivery** check box. This option might reduce performance but does not result in duplicate or missing event delivery. For more information, see "Ensure once-only event delivery (AssuredOnceDelivery)" on page 196.

c. In the **Retry limit for failed events** field, specify the number of times to try to deliver an event after a delivery failure. For more information, see "Retry limit for failed events (FailedEventRetryLimit)" on page 198.

- **Event persistence configuration**

  Note: In a HA Active-Active configuration, ensure that you provide values for the mandatory event persistence properties. If a value is not assigned to any of the event persistence properties, then the adapter throws a runtime exception error.

  a. Select **Auto create event table** check box if you want the adapter to create the Event Persistence table. For more information, see "Auto create event table" on page 195.

  b. In the **Event recovery table name** field, you can specify the name of the table that the adapter uses for event persistence. If you specify the **Event recovery table name** field, you must enter value in the **Table name to store the file processing status** field, to specify the table name that the adapter uses for file processing. For more information, see "Event recovery table name" on page 197 and "Table name to store the file processing status (EP_FileTableName) " on page 208 properties.

  c. In the **Event recovery data source (JNDI) name** field, specify the JNDI name of the data source that event persistence uses to connect to the JDBC database. For more information, see "Event recovery data source (JNDI) name" on page 197.

  d. In the **User name used to connect to event data source** field, specify the user name that the event persistence uses to connect to the database from the data source. For more information, see "User name used to connect to event data source" on page 209.

  e. In the **Password used to connect to event data source** field, specify the password that the event persistence uses to connect to the database from the data source. For more information, see "Password used to connect to event data source" on page 203.

  f. In the **Database schema name** field, specify the schema name of the database that the event persistence uses. For more information, see "Database schema name" on page 195.

  g. In the **Time interval for processing the fetched events (in seconds)** field, specify the time interval for the adapter to process the fetched events. For more information, see "Time out period for HA Active-Active event processing change (in seconds) (EP_Timeout)" on page 209.

- **Additional configuration**

  a. In the **Retrieve files with this pattern** field, specify the filter for the event files. For more information, see "Retrieve files with pattern" on page 204.

  b. Select **Include business object delimiter in the file content** check box to specify if the delimiter value specified in the SplitCriteria property is sent

with the business object content for further processing. For more information, see “Include business object delimiter in the file content” on page 200.

c. In the **Retrieve files in sorted order** list, specify sorting order of polled event files. For more information, see “Retrieve files in sorted order” on page 204.

   Note: In a HA Active-Active configuration, sorting of event files is not supported. If the default value (**no sort**) is changed, then the adapter throws a runtime exception error.

d. Select a value for the **File content encoding** field. If you are working with binary event data, select `BINARY`. If you are working with non-binary event data, such as text or XML, select a valid file encoding value, such as `UTF-8` (the default value). For more information, see “File content encoding” on page 199.

e. Select **Include total business object count in the ChunkInfo property** to specify that the total business object count is included in the chunk information of the dataobject being sent to the endpoint. For more information, see “Include total business object count in the ChunkInfo (includeBOCountInChunkInfo)” on page 200.

f. Select **Split file content based on size (bytes) or delimiter** check box to specify file splitting either by size or delimiter. For more information, see “Split file content based on size (bytes) or delimiter” on page 206.

   Note: This property is disabled if you select the **Pass only file name and directory, not the content** property.

g. In the **Split function class name** field, specify how the event file is to be split, by delimiter or by size. For more information, see “Split function class name” on page 207.

h. In the **Specify criteria to split file content** field, specify the delimiter that separates the business objects in the event file or the maximum size of the event file, depending on the value that is set in the Splitting Function Class Name. For more information, see “Specify criteria to split file content” on page 205.

i. Select **Poll subdirectories in event directory** check box to specify if the adapter polls the subdirectories within the event directory. For more information, see “Poll subdirectories in event directory” on page 203.

- **File archiving configuration**

  a. Select **Pass only file name and directory, not the content** check box to specify if the adapter sends the directory name and file name to the endpoint. For more information, see “Pass only file name and directory, not the content” on page 202.

     Note: In the external service wizard, you can use this property if both the **Split file content based on size (bytes) or delimiter** property and **Poll event files for modified content** properties are not selected.

  b. In the **Archive directory** field, specify the directory where the adapter archives processed event files. For more information, see “Archive directory” on page 194.

  c. In the **File extension for archive** field, specify the file extension used to archive unsuccessfully processed business objects in the input event file. For more information, see “File extension for archive” on page 199.

d. In the **Success file extension for archive** field, specify the file extension used to archive successfully processed business objects. For more information, see "Success file extension for archive" on page 208.

e. In the **Failure file extension for archive** field, specify the file extension used to archive unsuccessfully processed business objects in the input event file. For more information, see "Failure file extension for archive" on page 198.

- **Bidi properties**
  a. Select the **Bidi transformation** check box to specify bidirectional format. For more information about setting the **Bidi properties**, refer to the "Globalization" on page 213 section.
- **Logging and tracing**
  a. If you have multiple instances of the adapter, expand **Logging and tracing** and enter a value in the **Adapter ID** field that is unique for this instance. For more information about this property, see "Adapter ID (AdapterID)" on page 178.
  b. If you want to mask certain information so that the information is not displayed in the logs or traces, select **Disguise user data as "XXX" in log and trace files**.
  c. To specify the log file output location or define the level of logging for this module, select the **Change logging properties for wizard** check box. For information about setting logging levels, see "Configuring logging properties" on page 146.

6. For the **Function selector** field, select whether to use the default function selector configuration or create a new one.
   a. To create a function selector configuration, click **New**.
   b. In the Configure a New Function Selector window, click **Next**.
   c. Select the required function selector from the list of available function selectors.

   Note: A function selector assigns incoming messages or requests to the correct operation on the service.

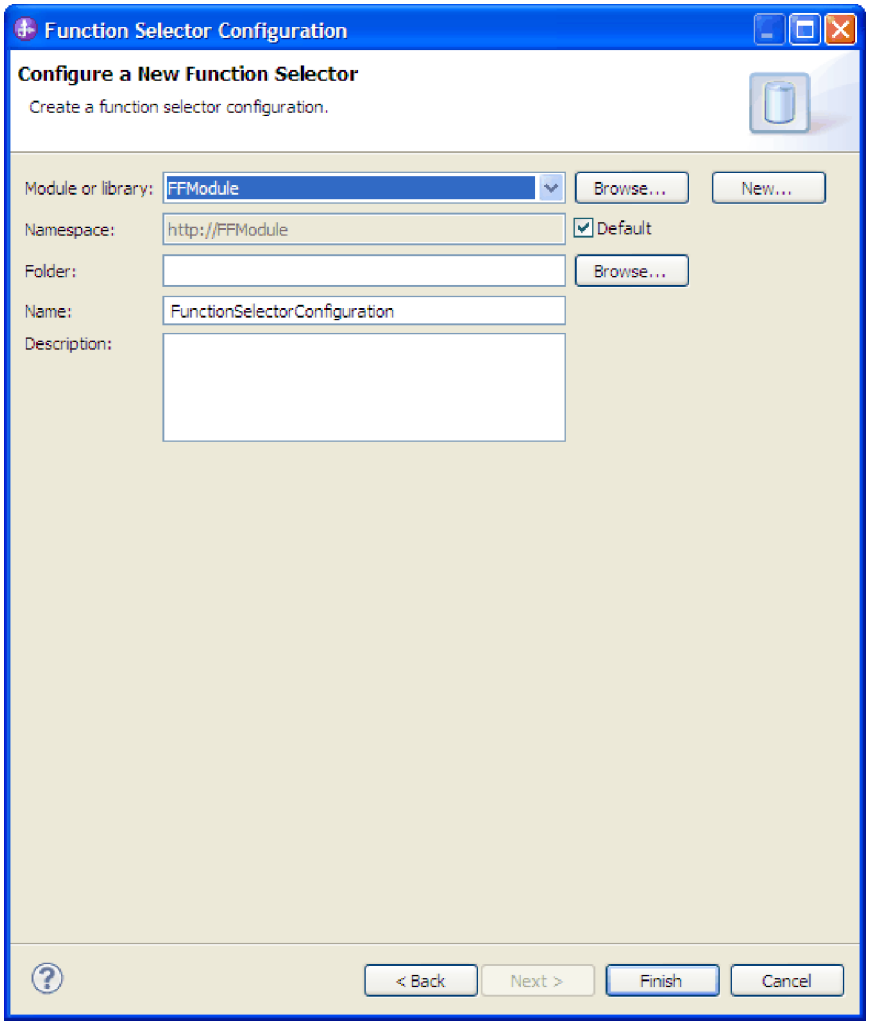


Figure 44. Creating a function selector configuration

Note: The enterprise information system (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 (base classes), you can edit it using the assembly editor.

7. To filter the inbound event file by configuring rules, click **Add** or **Edit** in the Rule editor table. The rule constitutes three parameters, namely, Property type, Operator and Value.

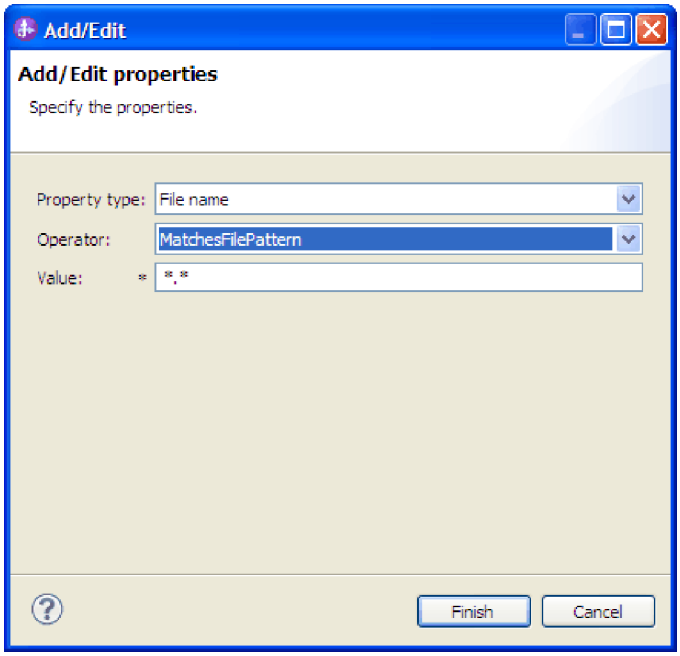


Figure 45. Adding or editing a rule

a. Select any of the following metadata filtering property types from **Property type** list.
   - FileName
   - FileSize
   - Directory
   - LastModified

b. Select the operator for the property type from the **Operator** list. Each of the property type metadata has its own operators.
   1) FileName contains the following operators:
      - Matches_File_Pattern (matches pattern)
      - Matches_RegExp (matches regular expression)
   2) FileSize metadata contains the following operators:
      - Greater than
      - Less than
      - Greater than or equal to
      - Less than or equal to
      - Equal to
      - Not equal to
   3) Directory contains Matches_RegExp as its operators.
   4) LastModified metadata contains the following operators:
      - Greater than
      - Less than
      - Greater than or equal to
      - Less than or equal to
      - Equal to
      - Not equal to

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_RegExp operator.

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

Note: The rules are grouped by using the logical **OR** operator, unless **END-OF-RULE** is selected in the property field. If an **END-OF-RULE** is selected between expressions (an expression can be a single rule or multiple rules grouped by an OR operator), it will be grouped using the logical **AND** operator. For example, If the rule A (FileName) is grouped with rule B (FileSize) using the logical **OR** operator, and on selecting the **END-OF-RULE** option, this expression will be grouped with another rule C (LastModified) using an **AND** operator. This can be represented as follows: ((A) OR (B)) AND (C)

For more information see, "Rule editor to filter files (ruleTable)" on page 209.

8. Click **Finish**.

Results

The adapter saves the activation specification properties.

What to do next

Select a data type for the module and name the operation associated with the chosen data type.

Related concepts

"Known issues in editing the Rule Table" on page 149
When configuring the adapter to filter event files based on a set of rules, some known issues can occur while editing the Rule Table in the Properties view. To correct the problem follow the solutions described here for each of these issues.

"File retrieval" on page 23
During inbound processing, you can manage the retrieval of the files by using the `Poll event files for modified content` or the `Time interval for polling unchanged files` property. You can also use the `Include only the newly appended content` property to retrieve only the appended file contents.

Related reference

"Connection properties for the wizard" on page 169
Connection properties are used to build a service description and save the built-in artifacts. These properties are configured in the external service wizard.

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

Selecting the operation and data type

Use the external service wizard to select a data type and name the operation associated with this data type. The external service wizard gives you a choice of

three data types: generic FlatFile business object, generic FlatFile business object with business graph, and user-defined type. 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 local file system before you can complete the steps given here.

About this task

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

Procedure

1. In the Add, Edit, or Remove Operations window, click **Add** to create an operation.
2. In the Specify the I/O Properties window, select a data type from the **Data type for the operation** list. Click **Next**. The three available data types are: Generic FlatFile business object, Generic FlatFile business object with business graph, and User-defined type. For more information about data types, see, "Business object structures" on page 159. In this example, **Generic FlatFile business object** is selected.

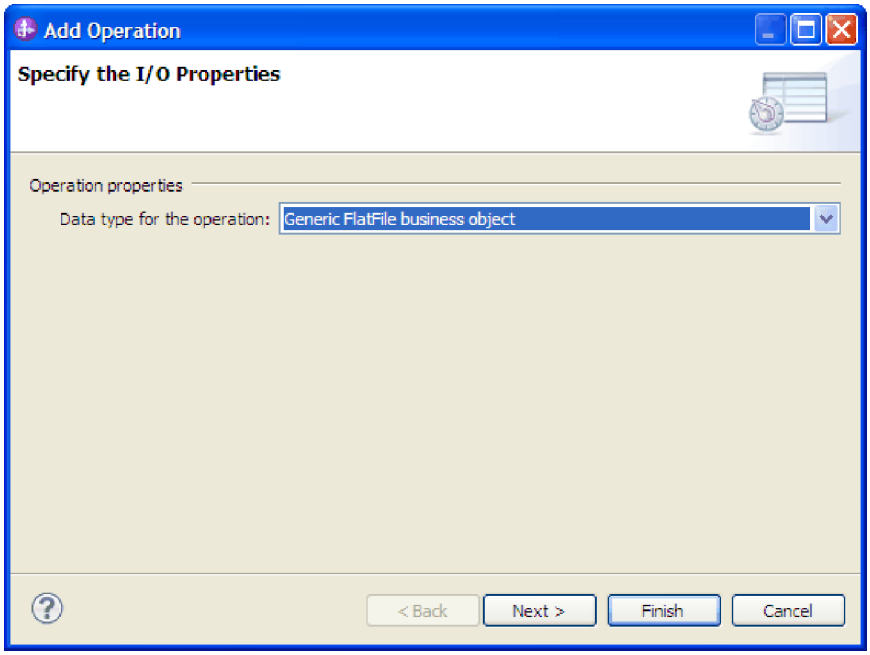


Figure 46. Naming the operation and specifying the input data type

3. In the Specify the I/O Properties window, the emitFlatFile is displayed as the inbound operation name in the **Operation name** field.

   Note: The emit operation is the only operation available during inbound processing.

Results

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

What to do next

Add and configure a data binding to be used with the module.

Configuring 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 the 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 the file with information it receives in the business object.

Before you begin

You must have selected a data type and chosen an operation 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 before running the external service wizard using IBM Integration Designer. To do this configuration, select **New** > **Configure Binding Resource** in IBM Integration Designer and complete the data binding screens described in this documentation.

Procedure

1. In the Specify the I/O Properties window, select either the **Use suggested data format 'FlatFileBaseDataBinding'** or the **Use a data format configuration** option from the **Data format options** list.
2. If you select the **Use a data format configuration** option, click **Select** to configure the data binding.
3. In the Select a Data Format Transformation window, select the **Use existing data format transformation from the list** option. From the list, select **FlatFileBaseDataBinding**. To configure a custom data binding, select the **Select your custom data format transformation from the workspace** option. Click **Next**.

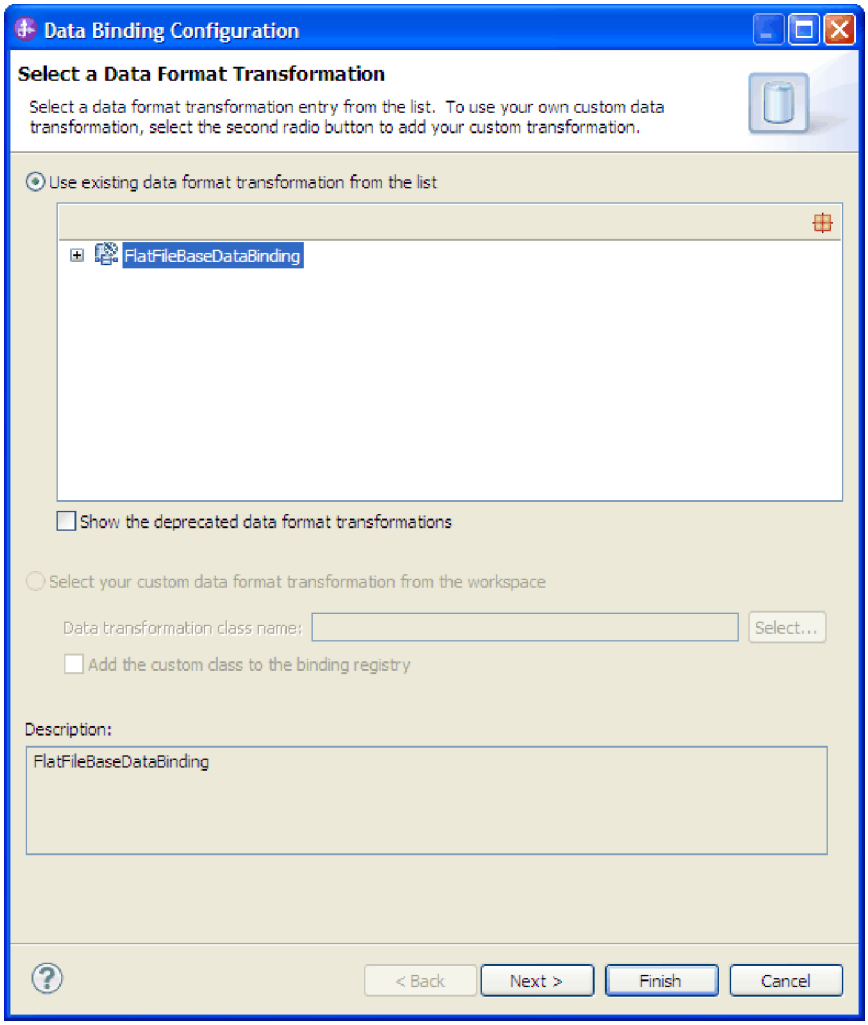


Figure 47. Selecting the data binding

4. In Specify the Data Transformation Properties window, click **Next**.

   Note: This window is used for configuring the data handlers.
5. In Configure a New Data Transformation window, provide the data binding configuration details.
   a. In the Configure a New Data Transformation window, the **Module** defaults to the module name you typed earlier in the external service. If you want to create a data binding for a different module, select **New** to create a module.
   b. If you want to select a new folder for the artifact, click **Browse** and select a new folder location.
   c. Type a **Name** for the data binding configuration. Click **Finish**. The data binding name is populated in the Specify the I/O Properties window.

Results

A data binding is configured for use with the module.

What to do next

Select the data handler configuration.

Configuring data handlers

Data handlers perform the conversions between a business object and a native format. The configuration in this topic is shown using the XML data handler. For comma-separated values (CSV) file format files, you must select the Delimited data handler.

Before you begin

You must have created a data binding before you specify data handlers for the module. Also, you must have predefined business objects using IBM Integration Designer Business Object Editor. If you stop the wizard here to create business objects, you need to start the wizard steps from the beginning.

Note: Data handlers can be configured before running the external service wizard using IBM Integration Designer. To do this configuration, select **New** > **Configure Binding Resource** in IBM Integration Designer and complete the data handler windows described in this documentation.

About this task

To specify data handlers, follow this procedure.

Procedure

1. In the Specify the Data Transformation Properties window, ensure that **DataHandler** is selected in the **Binding type** field.
2. In the **Configured data handler** field, click **Select**. Complete the following steps to create and configure a data handler.
   a. Select the class name for the data handler. In Select a Data Format Transformation window, click **XML** for data handler class name. This example uses **XML** data handler. Click **Next**.
3. Select the class name for the data handler. In Select a Data Format Transformation window, click **Use existing data format transformation from the list** option. A list of available data handler classes is displayed. Select **XML** data handler from the list and click **Next**.

   Note: To work with data files (CSV) that uses comma to separate the fields of data, select the **Delimited** data handler. For more information about working with delimited data files, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.wid.integ.doc/topics/rdelimitedcvs.html.

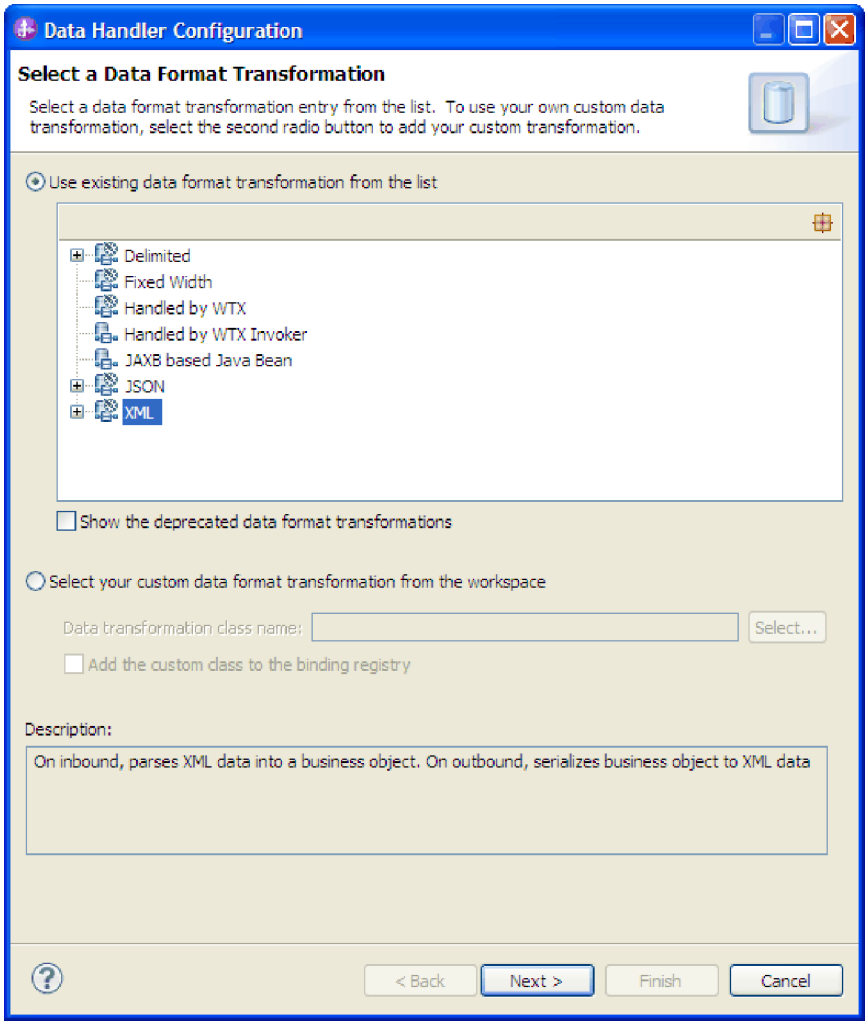


Figure 48. Selecting the data handler class

4. In the Specify the Data Transformation Properties window, specify the **Encoding**. The default is UTF-8. Click **Next**.

   Note: If **Delimited** option is selected, the Specify the Data Transformation Properties window provides the delimiter data handler configuration fields.
5. In the Configure a New Data Transformation window, specify the **Module**, **Namespace**, **Folder**, and **Name** for the data handler configuration. Click **Finish**.

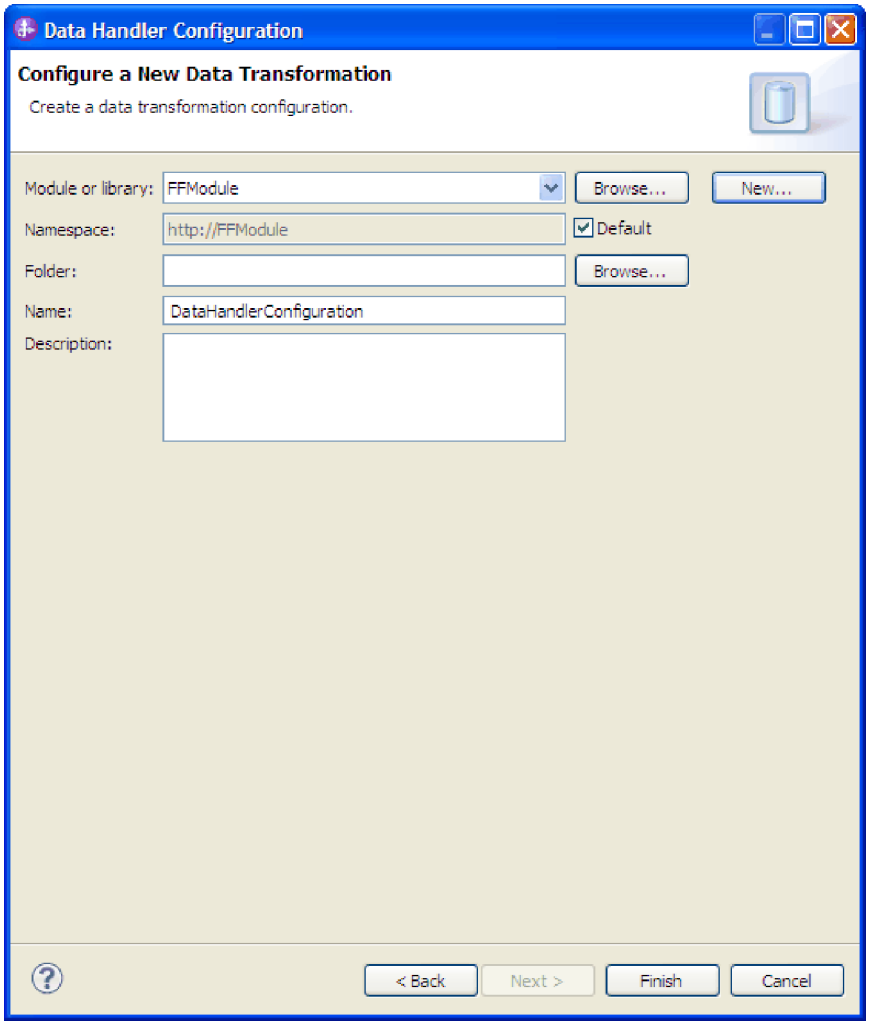


Figure 49. Specifying a name for the data handler configuration

6. Click **Finish**. The configured data handler is populated in the Specify the Data Transformation Properties window.

Results

Data handlers are created.

What to do next

Specify interaction specification properties and generate artifacts for the module.

Setting deployment properties and generating the service

Use the external service wizard to set activation specification properties and generate artifacts for use with your module. Artifacts are the business objects, WSDL files, and import and export files that are created as part of the external 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.

Before you begin

To set activation specification properties and generate artifacts for your module, you must have already configured data bindings and selected business objects.

About this task

To set activation specification properties and generate artifacts, follow this procedure. For more information about activation specification properties, see the "Activation specification properties" on page 191 topic.

Procedure

1. In the Add, Edit, or Remove Operations window, click **Advanced**.
2. In the **Additional configuration** section, type values for different interaction specification properties. Click **Next**.

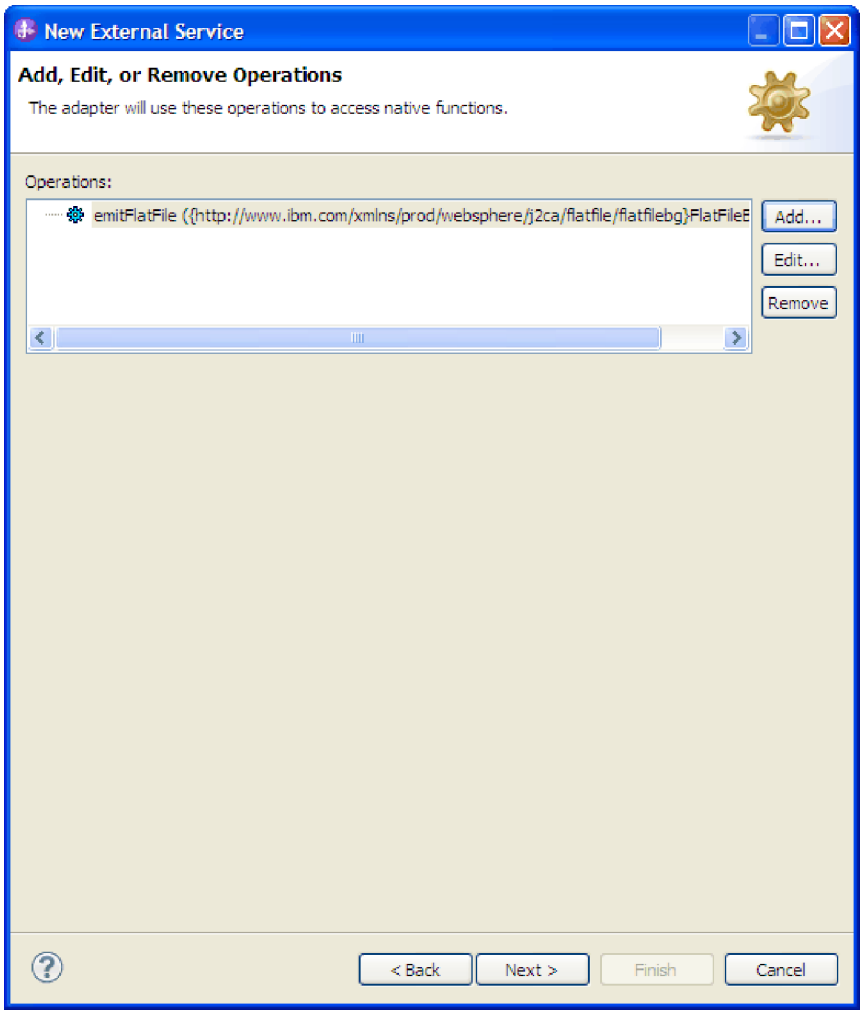


Figure 50. The inbound operation with InteractionSpec properties

3. In the Specify the Name and Location window, type a name for the interface. This name displays in the IBM Integration Designer assembly diagram. Click **Finish**.

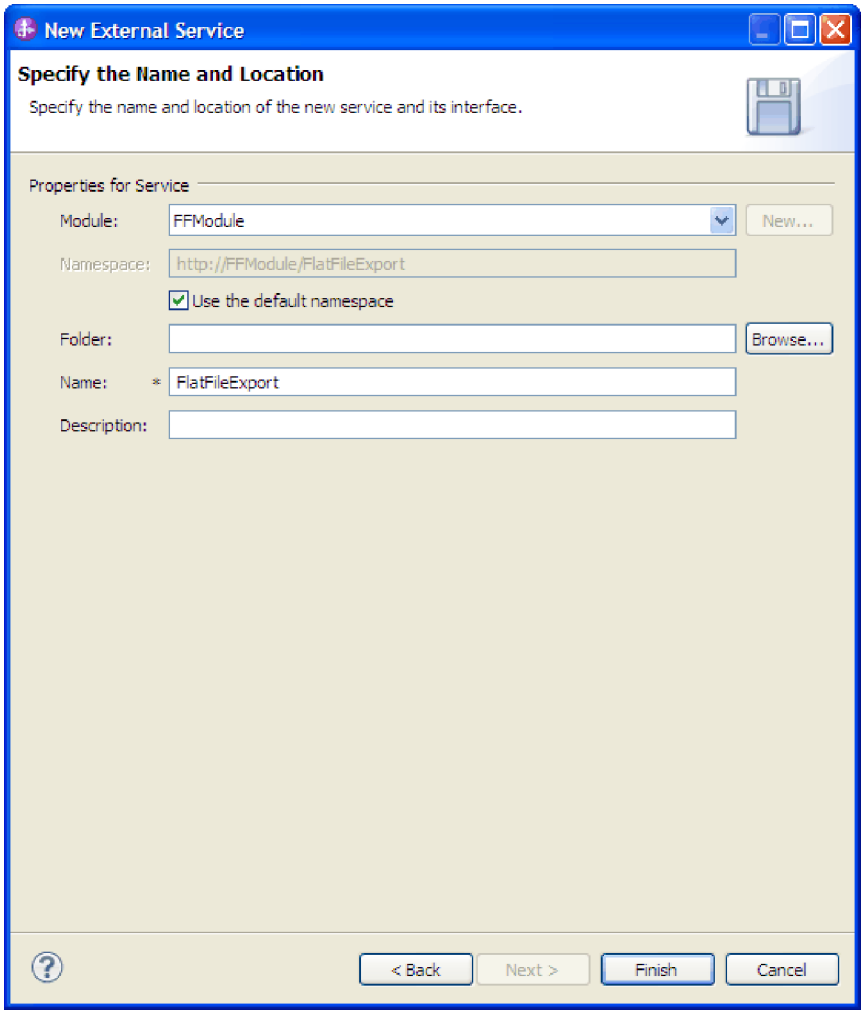


Figure 51. Naming the artifact

Results

The IBM Integration Designer generates the artifacts and an import. The inbound artifacts that are created are visible in the IBM Integration Designer Project Explorer under your module.

What to do next

Deploy the module.

Related reference

"Outbound configuration properties" on page 167
IBM WebSphere Adapter for Flat Files 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 IBM Business Process Manager or WebSphere Enterprise Service Bus using IBM Integration Designer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

"Inbound configuration properties" on page 186
WebSphere Adapter for Flat Files 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 IBM Integration Designer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

"Globalization" on page 213
WebSphere Adapter for Flat Files 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.

Chapter 5. Changing interaction specification properties

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

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 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 IBM Integration Designer, expand the module name.
2. Expand **Assembly Diagram** and double-click the interface.
3. Click the interface in the assembly editor. The module properties are displayed.
4. Click the **Properties** tab. You can also right-click the interface in the assembly 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.

Related reference

"Interaction specification properties" on page 180
Interaction specification properties contain the outbound connection properties the adapter uses to interface with the file system. You configure these properties using the external service wizard. To change the interaction specification properties after the application has been deployed, use the assembly editor in IBM Integration Designer.

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 IBM Integration Designer, the integrated test environment features runtime support for IBM Business Process Manager 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 IBM Integration Designer, 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 IBM Business Process Manager or WebSphere Enterprise Service Bus as EAR files using the administrative console or command-line tools.

Deploying the module for testing

In IBM Integration Designer, 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.

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 IBM Integration Designer. 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 IBM Integration Designer, 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.
   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** > **Java**.
   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 IBM Integration Designer, 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, generate and wire a *target component* to which the adapter sends the 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 servers settings.
   d. Click **Finish** to publish the server.

2. Add the module to the server.
   a. Switch to the servers view. In IBM Integration Designer, select **Windows** > **Show View** > **Servers**.
   a. Start the server. In the **Servers** tab in the lower-right pane of the IBM Integration Designer screen, right-click the server, and then select **Start**.
3. When the server status is *Started*, right-click 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 IBM Integration Designer integration test client.

Before you begin

You need to add the module to the server first.

About this task

Testing a module is 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 about testing a module using the test client, see the *Testing modules and components* topic in the IBM Integration Designer 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 IBM Business Process Manager or WebSphere Enterprise Service Bus in a production environment is a two-step process. First, you export the module in IBM Integration Designer as an enterprise archive (EAR) file. Second, you deploy the EAR file using the IBM Business Process Manager or WebSphere Enterprise Service Bus administrative console.

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 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 set **Deploy connector project** to **On server for use by multiple adapters** in the Specify the Service Generation and Deployment Properties 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. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources** > **Resource Adapters** > **Resource adapters**.
5. In the Resource adapters page, click **Install RAR**.

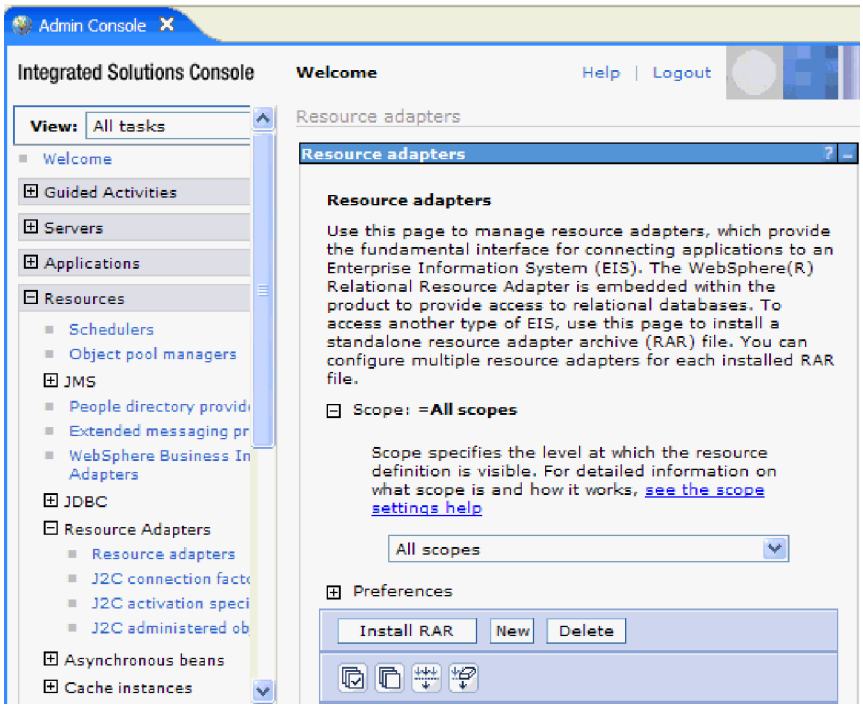


Figure 52. The Install RAR button on the Resource adapters page

6. 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:
   IID_installation_directory/ResourceAdapters/*adapter_name*/*adapter*.rar
7. Click **Next**.
8. Optional: In the Resource adapters page, change the name of the adapter and add a description.
9. Click **OK**.
10. 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 IBM Integration Designer, 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 IBM Business Process Manager 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 IBM Integration Designer 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. To export the source files, select the **Export source files** check box. 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 existing file**.
8. Click **Finish**.

Results

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

What to do next

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

About this task

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

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Applications** > **New Application** > **New Enterprise Application**.

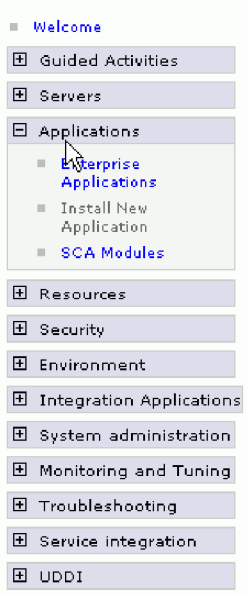


Figure 53. Preparing for the application installation window

5. Click **Browse** to locate your EAR file and click **Next**. The EAR file name is the name of the module followed by "App."

6. Optional: If you are deploying to a clustered environment, complete the following steps.
   a. On the **Step 2: Map modules to servers** window, select the module and click **Next**.
   b. Select the name of the server cluster.
   c. Click **Apply**.
7. Click **Next**. In the Summary page, verify the settings and click **Finish**.
8. Optional: If you are using an authentication alias, complete the following steps:
   a. Expand **Security** and select **Business Integration Security**.
   b. Select the authentication alias that you want to configure. You must have administrator or operator rights 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 the 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).

Related reference

"Inbound configuration properties" on page 186
WebSphere Adapter for Flat Files 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 IBM Integration Designer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

"Outbound configuration properties" on page 167
IBM WebSphere Adapter for Flat Files 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 IBM Business Process Manager or WebSphere Enterprise Service Bus using IBM Integration Designer or the administrative console, but connection properties for the external service wizard cannot be changed after deployment.

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 IBM Business Process Manager or WebSphere Enterprise Service Bus.

About this task

Custom properties are default configuration properties shared by all IBM WebSphere Adapters.

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

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.

2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications** > **Application Types** > **WebSphere enterprise application**.
5. From the Enterprise Applications list, click the name of the adapter module whose properties you want to change. The **Configuration** page is displayed.

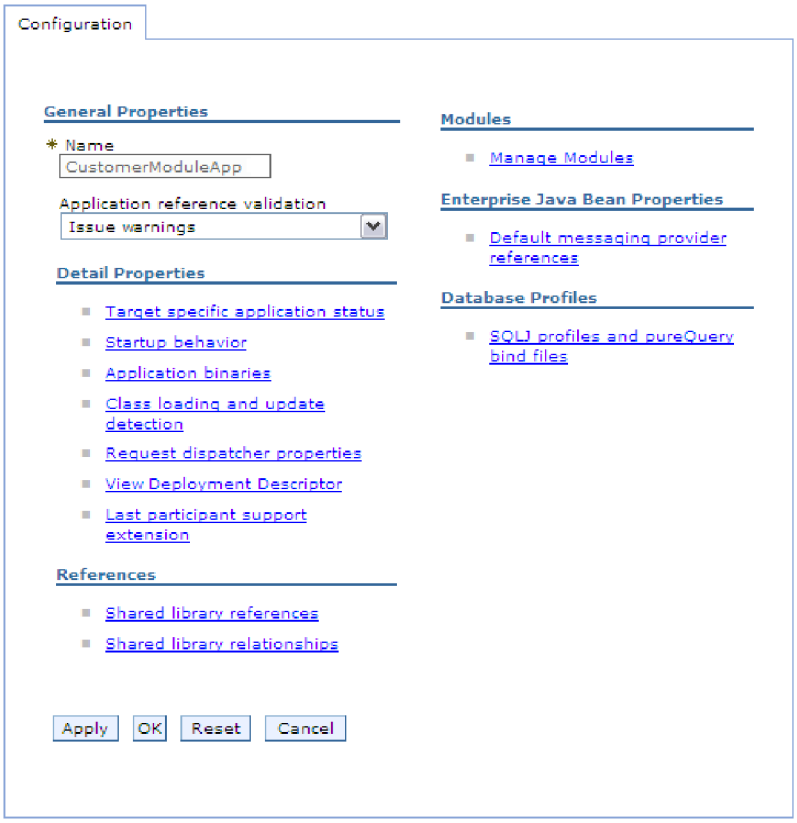


Figure 54. The Manage Modules selection in the Configuration tab

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

    Note: For more information about,
    - Inbound resource adapter properties, see"Resource adapter properties" on page 210
    - Outbound resource adapter properties, see "Resource adapter properties" on page 177

    a. Click the name of the property. The **Configuration** page for the selected property is displayed.
    b. Change the contents of the **Value** field or type a value, if the field is empty.

c. Click **OK**.

11. In the Messages area, click **Save**.

Results

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

Related reference

"Resource adapter properties" on page 177
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.

"Resource adapter properties" on page 210
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.

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 IBM Business Process Manager or WebSphere Enterprise Service Bus.

About this task

You use managed connection factory properties to configure the target local file system 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. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications** > **Application Types** > **WebSphere enterprise application**.
5. In the Enterprise Applications list, click the name of the adapter module whose properties you want to change.

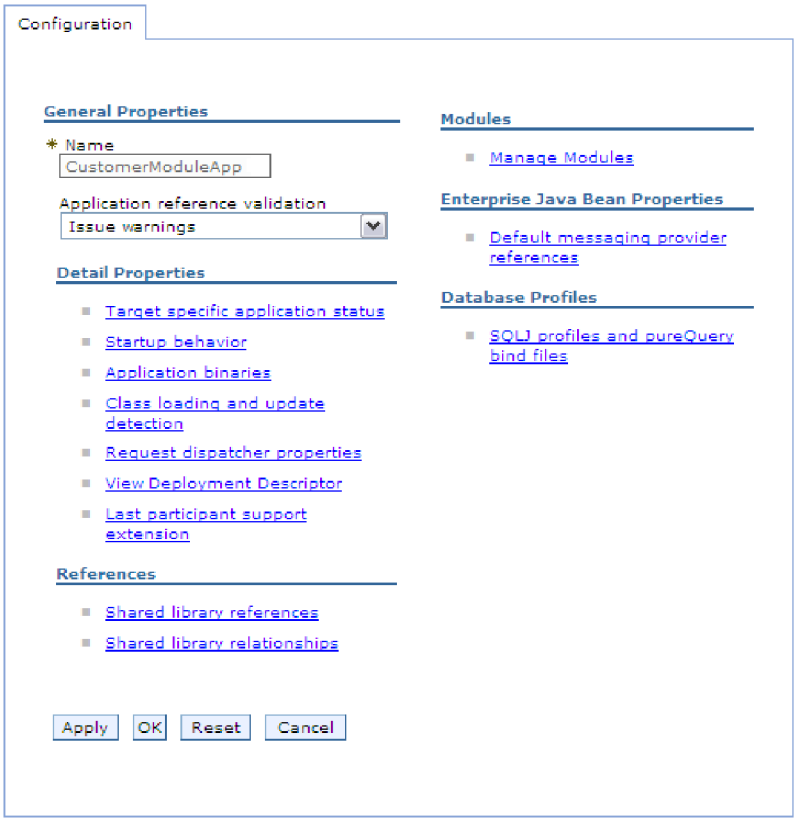


Figure 55. The Manage Modules selection in the Configuration tab

6. Under **Modules**, click **Manage Modules**.
7. Click **IBM WebSphere Adapter for Flat Files**.
8. In the **Additional Properties** list, click **Resource Adapter**.
9. On the next page, from the **Additional Properties** list, click **J2C connection factories**.
10. Click the name of the connection factory associated with your adapter module.
11. In the **Additional Properties** list, click **Custom properties**.

    Custom properties are those J2C connection factory properties that are unique to IBM WebSphere Adapter for Flat Files. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
12. For each property you want to change, perform the following steps.

    Note: See "Managed connection factory properties" on page 172 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**.
13. In the Messages area, click **Save**.

Results

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

Related reference

"Managed connection factory properties" on page 172
Managed connection factory properties specify information the adapter needs at run time for outbound communication with the local file system.

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 IBM Business Process Manager 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. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications** > **Application Types** > **WebSphere enterprise application**.
5. From the Enterprise Applications list, click the name of the adapter module whose properties you want to change.

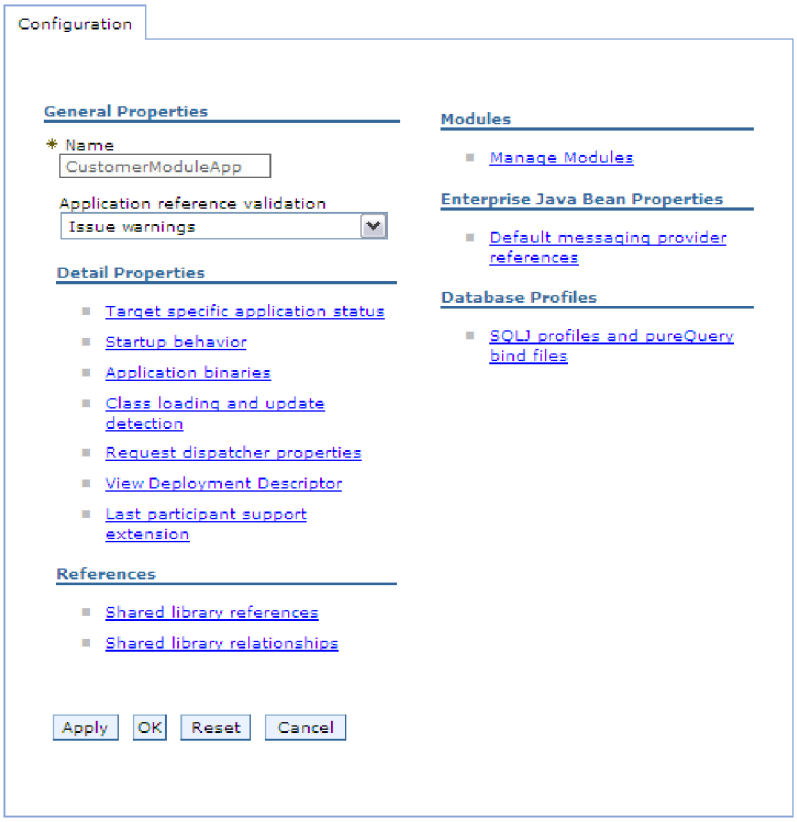


Figure 56. The Manage Modules selection in the Configuration tab

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

    Note: See "Activation specification properties" on page 191 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**.
13. In the Messages area, click **Save**.

Results

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

Related reference

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

Changing configuration properties for stand-alone adapters

To set configuration properties after you install a stand-alone adapter, use the administrative console of the runtime environment. Provide the general information about the adapter and then set the resource adapter properties (which are used for general adapter operation). If the adapter is used for outbound operations, create a connection factory and then set the properties for it. If the adapter is used for inbound operations, create an activation specification and then set the 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 IBM Business Process Manager 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 IBM Business Process Manager or WebSphere Enterprise Service Bus.

About this task

Custom properties are default configuration properties shared by all IBM WebSphere Adapters.

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

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources** > **Resource Adapters** > **Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for Flat Files**.
6. In the **Additional Properties** list, click **Custom properties**.
7. For each property you want to change, perform the following steps.

   Note: For more information about,
   - Inbound resource adapter properties, see"Resource adapter properties" on page 210
   - Outbound resource adapter properties, see "Resource adapter properties" on page 177

   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**.

8. In the Messages area, click **Save**.

Results

The resource adapter properties associated with your adapter are changed.

Related reference

"Resource adapter properties" on page 177
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.

"Resource adapter properties" on page 210
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.

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 IBM Business Process Manager 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 IBM Business Process Manager or WebSphere Enterprise Service Bus.

About this task

You use managed connection factory properties to configure the target local file system 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. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources** > **Resource Adapters** > **Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for Flat Files**.
6. In the **Additional Properties** list, click **J2C connection factories**.

7. If you are going to use an existing connection factory, skip ahead to select from the list of existing connection factories.

   Note: If you have selected **Specify connection properties** when you use 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 can type `AdapterCF`.
   c. Type a value for **JNDI name**. For example, you can type `com/eis/AdapterCF`.
   d. Optional: Select an authentication alias from the **Component-managed authentication alias** list.
   e. Click **OK**.
   f. In the Messages area, click **Save**.

      The newly created connection factory is displayed.

Figure 57. User-defined connection factories for use with the resource adapter

8. In the list of connection factories, click the one you want to use.
9. In the **Additional Properties** list, click **Custom properties**.

   Custom properties are those J2C connection factory properties that are unique to WebSphere Adapter for Flat Files. 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 connection factory properties" on page 172 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. After you have finished setting properties, click **Apply**.
12. In the Messages area, click **Save**.

Results

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

Related reference

"Managed connection factory properties" on page 172
Managed connection factory properties specify information the adapter needs at run time for outbound communication with the local file system.

Setting activation specification properties for stand-alone adapters

To set activation specification properties for your stand-alone adapter after it has been installed on IBM Business Process Manager 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 IBM Business Process Manager 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. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources** > **Resource Adapters** > **Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for Flat Files**.
6. In the **Additional Properties** list, click **J2C activation specifications**.
7. If you are going to use an existing activation specification, skip ahead to select from an existing list of activation specifications.

   Note: If you have selected **Use predefined connection properties** when you use the external service wizard to configure the adapter module, you must 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 can type `AdapterAS`.
   c. Type a value for **JNDI name**. For example, you can type `com/eis/AdapterAS`.
   d. Optional: 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.

8. In the list of activation specifications, click the one you want to use.
9. In the Additional Properties list, click **J2C activation specification custom properties**.
10. For each property you want to set, perform the following steps.

    Note: See "Activation specification properties" on page 191 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. After you have finished setting properties, click **Apply**.
12. In the Messages area, click **Save**.

Results

The activation specification properties associated with your adapter are set.

Related reference

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

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. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Applications** > **Application Types** > **WebSphere enterprise applications**.

   Note: The administrative console is labeled "Integrated Solutions Console".
5. Select the application that you want to start. The application name is the name of the EAR file you installed, without the .EAR file extension.
6. 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. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Applications** > **Application Types** > **WebSphere enterprise applications**.

   Note: The administrative console is labeled "Integrated Solutions Console".
5. Select the application that you want to stop. The application name is the name of the EAR file you installed, without the .EAR file extension.
6. Click **Stop**.

Results

The status of the application changes to Stopped, and a message stating that the application has stopped is displayed 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 IBM WebSphere Adapter for Flat Files. 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 IBM Business Process Manager or WebSphere Enterprise Service Bus.

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 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 IBM Business Process Manager or WebSphere Enterprise Service Bus website: 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 to gather the 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, see "Enabling tracing with the Common Event Infrastructure" on page 142.
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.

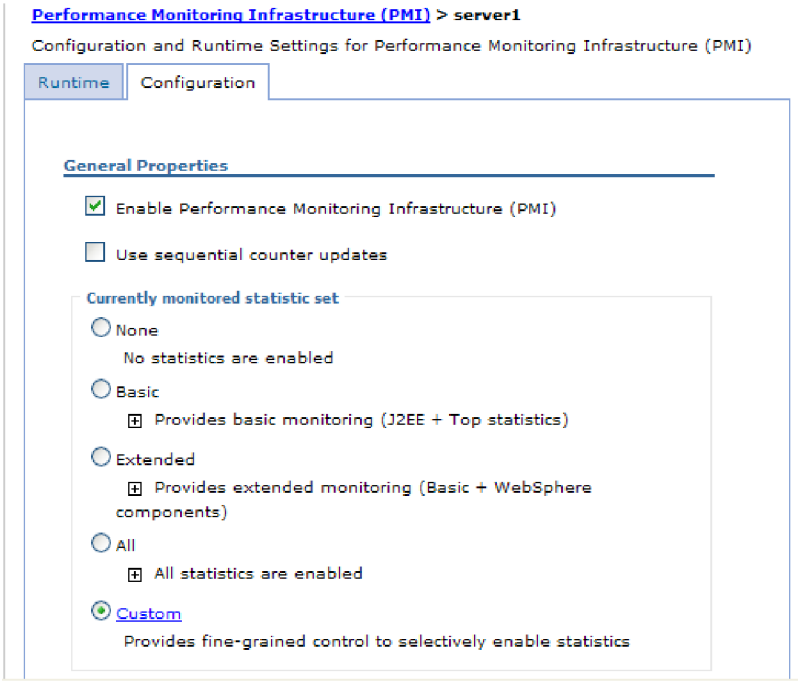


Figure 58. Enabling Performance Monitoring Infrastructure

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.

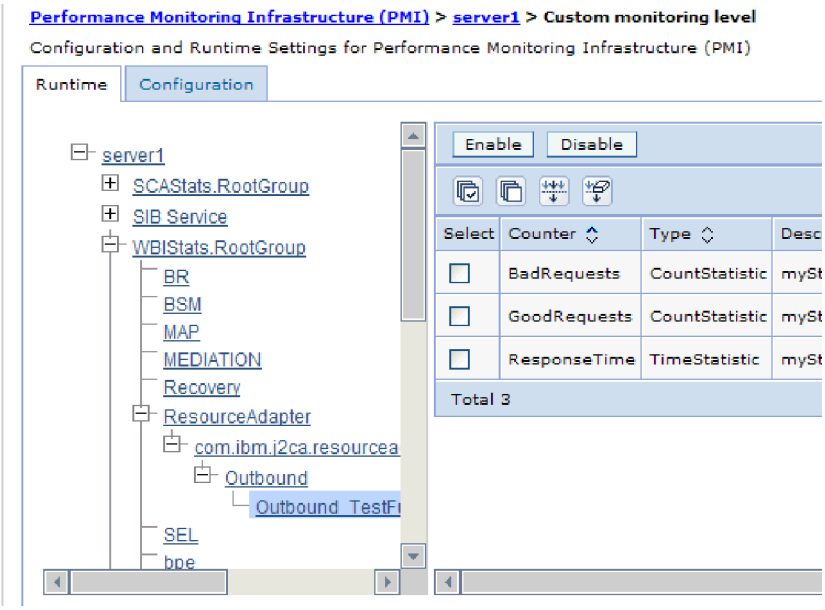


Figure 59. Runtime tab used for configuring PMI

e. Click **WBIStats.RootGroup**. This is a PMI sub module for data collected in the root group. This example uses the name WBIStats 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.

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 IBM Business Process Manager or WebSphere Enterprise Service Bus.

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 **WBIStatsRootGroup**.
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.

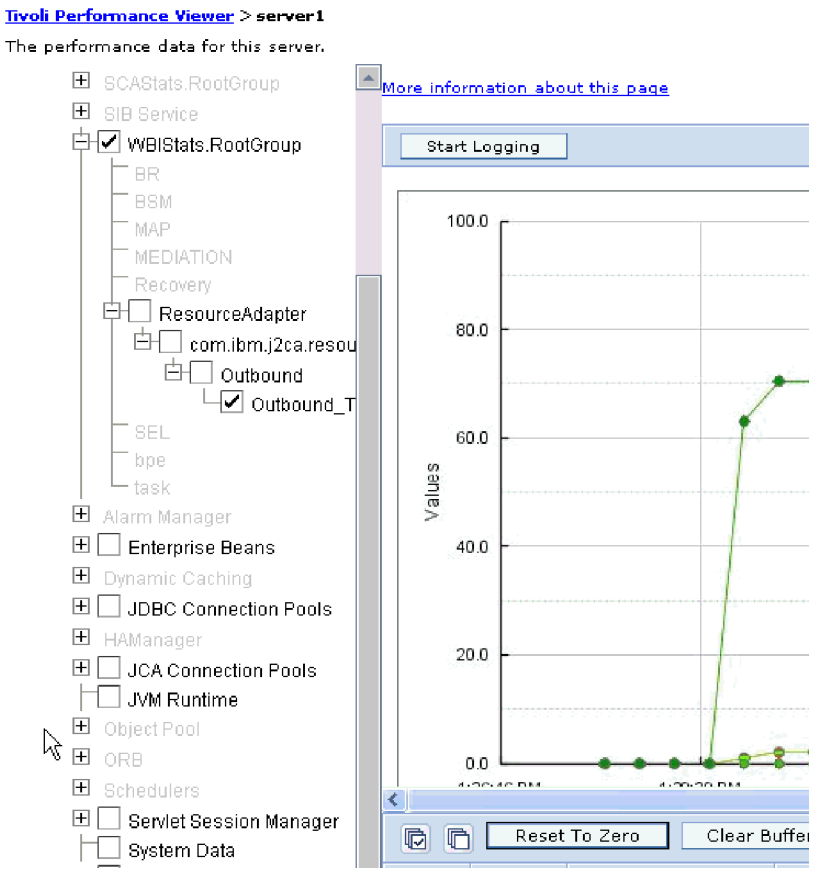


Figure 60. Adapter performance statistics, using graph view

Enabling tracing with the Common Event Infrastructure

The adapter can use the Common Event Infrastructure (CEI), 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

Use this procedure to report CEI entries in the trace log file by using the Common Base Event Browser within the administrative console.

Procedure

1. In the administrative console, click **Troubleshooting**.
2. Click **Logs and Trace**.
3. From 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, select one of the following options:

     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.

Chapter 8. Troubleshooting and support

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

Related reference

"Adapter messages" on page 218
View the messages issued by WebSphere Adapter for Flat Files at the following location.

Adapter returns version conflict exception message

Adapter returns version conflict exception message

Problem

When you install multiple adapters with different versions of CWYBS_AdapterFoundation.jar, and if a lower version of the CWYBS_AdapterFoundation.jar is loaded during runtime, the adapter will return the ResourceAdapterInternalException error message, due to a version conflict. For example, when you install Oracle E-Business Suite adapter version 7.0.0.3 and WebSphere Adapter for Flat Files version 7.5, the following error message is displayed: IBM WebSphere Adapter for Flat Files has loaded file:/C:/IBM/WebSphere/ProcServer7/profiles/ProcSrv01/installedConnectors/CWYOE_OracleEBS.rar/CWYBS_AdapterFoundation.jar with version 7.0.0.3. However, the base level of this jar required is version 7.5. When you install multiple adapters with different CWYBS_AdapterFoundation.jar versions, the adapter returns the ResourceAdapterInternalException message, due to a version conflict. To overcome this conflict, you must migrate all adapters to the same version level. For further assistance, contact WebSphere Adapters Support for help.

Solution

Migrate all adapters to the same version level.

For further assistance, visit http://www-947.ibm.com/support/entry/portal/Overview/Software/WebSphere/WebSphere_Adapters_Family.

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 `FFRA` plus the value of the adapter ID property. For example, if the adapter ID property is set to `001`, the component ID is `FFRA001`.

If you run multiple instances of the same adapter, ensure that the first nine characters of the adapter ID property are unique for each instance so that you can

correlate the 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 the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for Flat Files to `001` and `002`. The component IDs for those instances, `FFRA001` and `FFRA002`, 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. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `FFRAInstance0`.

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. It prevents 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, see the "Adapter ID (AdapterID)" on page 178 property.

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.

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 IBM 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.

Note: For more information about monitoring on a IBM Process Server, including service components and event points, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.admin.doc/topics/welcome_wps_mon.html.

You can change the log configuration statically or dynamically. Static configuration takes effect when you start or restart the application server. Dynamic or run time 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 child logs, which recursively propagate the change to their child log, 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, select **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 WebSphere Adapter for Flat Files only, select the **com.ibm.j2ca.flatfile.*** 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 IBM 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 IBM Process Server is written to the `SystemOut.log` and `trace.log` files.

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 Flat Files.
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.

Known issues in editing the Rule Table

When configuring the adapter to filter event files based on a set of rules, some known issues can occur while editing the Rule Table in the Properties view. To correct the problem follow the solutions described here for each of these issues.

Symptoms:

When an existing Rule Table row is configured in the Properties view, the following issue can occur:

The **Finish** option is not enabled sometimes.

Problem:

After you have completed entering all the required properties, the **Finish** option is not enabled for you to complete the editing of the Rule Table.

Solution:

To correct this problem, use either of the following workaround:

- Use **Tab** to move between the fields.
- Keep the focus away from the **Value** field either to **Operator** or the **Property** field.

Related tasks

"Setting deployment and runtime properties" on page 97
After you have decided whether your module is to be used for outbound or inbound communication with the enterprise information system (local file system), you must configure the activation specification properties, which hold the inbound event processing configuration information for the export.

Related reference

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

Support for global elements without wrapper

When global element without wrapper is used as input type, you need to take care of using the correct configuration described for the below listed scenarios to get the expected result.

Global element of named type without wrapper during outbound processing

When global element of named type without wrapper is used as input type in adapter outbound using UTF8XML Datahandler, the file is serialized with global element type name as root element name, instead of the global element name.

To serialize file to get the global element name as the root element name, you need to use the XML Datahandler and specify the global element name as the root element name in XML datahandler configuration.

Global element of anonymous type without wrapper

When global element of anonymous type without wrapper is used as input type in adapter inbound or outbound retrieve, the data object is emitted back to SCA component. When this data object is serialized, it returns the type name of dataobject as 'globalelementname_._type'.

To get the correct data object type, in order to be used for a global element of anonymous type without wrapper, for inbound as well as outbound retrieve, you need to use the following code snippet.

```
The following sample code can be used to get the correct
dataobject details for global element of anonymous type
without wrapper, which is named as GlobalElementExample1.

import java.io.ByteArrayOutputStream;
import java.io.IOException;

import commonj.sdo.DataObject;
import commonj.sdo.Type;

import com.ibm.websphere.bo.BOFactory;
import com.ibm.websphere.bo.BOXMLSerializer;
import com.ibm.websphere.sca.ServiceManager;

public void emit(DataObject globalElementExample1) {
ServiceManager s = ServiceManager.INSTANCE;
BOFactory factory= (BOFactory) s.locateService
("com/ibm/websphere/bo/BOFactory");
DataObject dobj= factory.createByElement
(globalElementExample1.getType().getURI(), "GlobalElementExample1");
final Type type = dobj.getType();
String typeName = type.getName();
if (typeName.endsWith("_._type"))
         typeName = typeName.substring(0, typeName.indexOf("_._type"));
BOXMLSerializer serializer = BOXMLSerializer)s.locateService
("com/ibm/websphere/bo/BOXMLSerializer");
ByteArrayOutputStream baos = new ByteArrayOutputStream();
serializer.writeDataObject(globalElementExample1, type.getURI(), typeName, baos);
String bo = new String(baos.toByteArray());
System.out.println("bo : "+bo);
}
```

Global elements in SDOX mode throw exceptions

In SDOX (Service Data Objects - XML Cursor Interface) mode, the adapter throws the DataBindingException or IllegalArgumentException exception when the global element feature is used in the business object structure.

DataBindingException when using anonymous complex type global element

The adapter throws a DataBindingException exception when running in SDOX mode during the outbound operations, if it uses the following settings:

- the business object structure contains an anonymous complex type global element with no name space definition, and
- the business object is directly specified as the data type in outbound artifacts.

Note: The exception can occur when running the Create, Append, Overwrite, or Retrieve outbound operation.

The stack trace of WebSphere Adapter for Flat Files contains a trace report. An example of a trace report is shown here.

```
[12/3/09 10:26:00:156 CST] 00000058 FfdcProvider I com.ibm.ws.ffdc.impl.FfdcProvider logIncident FFDC100
FFDC Incident emitted on C:\W7\profiles\ProcSrv01\logs\ffdc\server1_71327132_09.12.03_10.26.00.140451242
txt com.ibm.j2ca.flatfile.emd.runtime.FlatFileBaseDataBinding getBiDiContext
[12/3/09 10:26:00:156 CST] 00000058 FfdcProvider I com.ibm.ws.ffdc.impl.FfdcProvider logIncident FFDC100
FFDC Incident emitted on C:\W7\profiles\ProcSrv01\logs\ffdc\server1_71327132_09.12.03_10.26.00.156422027
txt com.ibm.j2ca.flatfile.emd.runtime.FlatFileBaseDataBinding getRecord
[12/3/09 10:26:00:156 CST] 00000058 FFRADB E Error on getRecord(): commonj.connector.runtime.DataBinding
Error while bidi format
at com.ibm.j2ca.flatfile.emd.runtime.FlatFileBaseDataBinding.getBiDiContext(FlatFileBaseDataBinding.java
at com.ibm.j2ca.flatfile.emd.runtime.FlatFileBaseDataBinding.getRecord(FlatFileBaseDataBinding.java:134)
```

To correct the problem, while using anonymous complex type global element for outbound operations, use the `BOWrapper` instead of business object as the data type.

IllegalArgumentException during the outbound operations

The adapter throws a IllegalArgumentException exception when running in SDOX mode during the outbound operations, if it uses the following settings:

- the business object structure contains a global element, and
- the `BOWrapperBG` is used as the data type in the outbound artifacts.

Note: The exception can occur when running the Create, Append, Overwrite, or Retrieve outbound operation.
The stack trace of WebSphere Adapter for Flat Files contains a trace report. An example of a trace report is shown here.

```
[12/8/09 18:22:00:906 CST] FFDC Exception:java.lang.IllegalArgumentException SourceId:com.ibm.j2ca.
flatfile.emd.runtime.FlatFileBaseDataBinding ProbeId:getRecord Reporter:java.lang.Class@61e461e4
java.lang.IllegalArgumentException: Expected a DataObject GlobalElementExample1Wrapper
inside GlobalElementExample1WrapperBG but found none.
at com.ibm.j2ca.extension.emd.runtime.internal.DataBindingUtil.getBOFromBG(DataBindingUtil.java:459)
at com.ibm.j2ca.flatfile.emd.runtime.FlatFileBaseDataBinding.getContentObject(FlatFileBaseDataBinding.ja
at com.ibm.j2ca.flatfile.emd.runtime.FlatFileBaseDataBinding.getRecord(FlatFileBaseDataBinding.java:118)
at com.ibm.ws.sca.binding.j2c.J2CMethodBindingImpl.invoke(J2CMethodBindingImpl.java:1202)
at com.ibm.ws.sca.binding.j2c.J2CInterfaceBindingImpl.invoke(J2CInterfaceBindingImpl.java:152)
at com.ibm.ws.sca.binding.j2c.handler.J2CImportHandler.invokeDynamicImport(J2CImportHandler.java:1314)
```

To correct the problem, you can use either of the following workaround:

- When running this scenario using a component to start the outbound operation, use the `BOWrapper` instead of `BOWrapperBG` as the data type.
- Call the outbound operations directly from the Java code, BPEL (Business Process Execution Language), and other mediation flows.

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 IBM Business Process Manager 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 IBM Business Process Manager or WebSphere Enterprise Service Bus documentation.

Incomplete file processing in UNIX environments

In UNIX environments, such as AIX, the files being copied to the event directory are made available to the adapter for processing even before the files are completely copied. This leads to incorrect business data being sent to the downstream components.

Symptoms:

This problem occurs due to absence of locking mechanism in the UNIX environments, when files are being written to a directory. The other applications are not prohibited during the copying process from accessing the files.

Problem:

During inbound polling, the adapter picks up the files that are not completely copied, causing erroneous results because of processing of the incomplete files.

Solution:

To resolve this problem, use any one of the following workaround.

- Use a staging directory to copy the event files. After the event files are copied in the staging directory, use the **Move** command to transfer the file to the event directory. You can use some of the sample UNIX scripts that are provided for your use as part of the adapter. The script file named CheckIfFileIsOpen.sh is available in the UNIX script file folder in the adapter installer. The script performs a check to see if a file has been copied to the staging directory. After the file copy is completed, the script moves the file to the event directory for further processing.

  Note: The staging directory and event directory must be on the same file system.
- You can also use the **Time interval for polling unchanged files (in milliseconds)** property to ensure that adapter processes the completely copied files. This property enables the adapter to retrieve only those files that are not modified in the event directory for a specified time interval. When this property is selected, the adapter retrieves only the unchanged files during the poll cycles.

Out of memory exception

Out of memory exception error while polling large-size files during inbound processing

Problem

During inbound polling, the adapter fails to poll large-size files and generates an out of memory exception error.

Solution

If you split an event file by size, ensure that the SplitCriteria property contains a valid chunk value. A non-negative integer is considered as a valid chunk. If the value in the SplitCriteria property is not configured, the whole file is processed as a single business object and can throw exceptions with large-size files. When you specify the split size value, the file is processed in split sized chunks resulting in a successful poll. For more information about the splitting of files, see "File splitting" on page 25.

Out of memory exception error while retrieving large-size files during outbound processing

Problem

When retrieving content for large-size files during the retrieve operation, the adapter generates an out of memory exception error.

Solution

If an out of memory exception error is generated, it indicates that the machine configuration does not support processing of large-size files. For more information about the retrieve operation, see "Retrieve operation" on page 7.

XAResourceNotAvailableException

When the IBM Process Server log contains repeated reports of the com.ibm.ws.Transaction.XAResourceNotAvailableException exception, remove transaction logs to correct the problem.

Symptom:

When the IBM Process Server is started after a server failure, the following exception is repeatedly logged in the IBM Process Server log file:

com.ibm.ws.Transaction.XAResourceNotAvailableException

The stack trace of WebSphere Adapter for Flat Files contains a trace report. An example of a trace report is shown here.

```
00000015 XARecoveryDat W   WTRN0005W: The XAResource for a transaction participant
could not be recreated and transaction recovery may not be able to complete properly.
The resource was com.ibm.ws.Transaction.JTA.ASWrapper@61c461c4.
The exception stack trace follows: com.ibm.ws.Transaction.XAResourceNotAvailableException:
java.security.PrivilegedActionException: java.lang.ClassNotFoundException:
com.ibm.j2ca.flatfile.FlatFileResourceAdapter
 at com.ibm.ws.Transaction.JTA.ASXAResourceFactoryImpl.getXAResource
  (ASXAResourceFactoryImpl.java:97)
 at com.ibm.ws.Transaction.JTA.XARecoveryData.getXARminst(XARecoveryData.java:529)
 at com.ibm.ws.Transaction.JTA.XARecoveryData.recover(XARecoveryData.java:644)
 at com.ibm.ws.Transaction.JTA.PartnerLogTable.recover(PartnerLogTable.java:524)
 at com.ibm.ws.Transaction.JTA.RecoveryManager.resync(RecoveryManager.java:1859)
 at com.ibm.ws.Transaction.JTA.RecoveryManager.run(RecoveryManager.java:2580)
 at java.lang.Thread.run(Thread.java:810)
  Caused by: java.security.PrivilegedActionException: java.lang.ClassNotFoundException:
  com.ibm.j2ca.flatfile.FlatFileResourceAdapter
 at com.ibm.ws.security.util.AccessController.doPrivileged(AccessController.java:122)
 at com.ibm.ejs.j2c.RAWrapperImpl.createAndConfigureRA(RAWrapperImpl.java:2064)
```

```
at com.ibm.ejs.j2c.RAWrapperImpl.startRA(RAWrapperImpl.java:584)
at com.ibm.ejs.j2c.RAWrapperImpl.getStartedRA(RAWrapperImpl.java:1662)
at com.ibm.ejs.j2c.ActivationSpecWrapperImpl.getStartedRA
 (ActivationSpecWrapperImpl.java:1368)
at com.ibm.ws.Transaction.JTA.ASXAResourceFactoryImpl.getXAResource
 (ASXAResourceFactoryImpl.java:92)
... 6 more
```

Problem:

A resource deployed at the node level was being removed and there was an abrupt shutdown of the IBM Process Server. This results in killing the Java process while the IBM Process Server was committing or rolling back a transaction for that resource. When the IBM Process Server is restarted, it tries to recover the transaction but cannot do so as the resource was removed.

Solution:

To correct this problem, use the following procedure:

1. Reinstall the WebSphere Adapter for Flat Files at the node level.

   Note: The IBM Process Server is already running, therefore you do need to restart it.
2. From the administrative console, restore all the activation specification properties. When adding the properties, you must use the same JNDI reference that the adapter was referencing for the module.
3. Stop the IBM Process Server.
4. Restart the IBM Process Server in the `Recovery` mode. In administrative console, move to the bin directory of the IBM Process Server installation and type the following command: **`startServer <server name> -recovery`**

   Note: The IBM Process Server stops after the IBM Process Server recovery is completed.
5. Start the IBM Process Server in the normal mode.

   Note: After the IBM Process Server is started, the event processing will start again.
6. Verify the stack trace file to ensure that there is no occurrence of the XAResourceNotAvailableException exception.

Uninstall the WebSphere Adapter for Flat Files deployed at the node level

The XAResourceNotAvailableException exception can occur when there are some typical events, such as deleting an adapter, that have unprocessed events from the server.

1. Use the administrative console to stop all applications that are using the adapter.
2. Uninstall all the applications that are using the adapter.
3. Uninstall the WebSphere Adapter for Flat Files.
4. Stop and start the IBM 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.

Disabling end point applications of the passive adapter

Problem

In the active-passive configuration mode of the adapters, the endpoint application of the passive adapter instance also listens to the events or messages even if the enableHASupport property is set to `True`.

Cause

By default, in WebSphere Application Server, version 7.0, the `alwaysactivateAllMDBs` property in the JMS activation specification is set to `True`. This enables the endpoint application of all the adapter (active or passive) instances to listen to the events.

Solution

To stop the endpoint application of the passive adapter instance from listening to the events, you must set the `alwaysactivateAllMDBs` property value to `False`. The JMS activation specification is associated with one or more MDBs and provides the necessary configuration to receive events. If the `alwaysActivateAllMDBs` property is set to `False`, then the endpoint application of only the active adapter instance receives the events.

Perform the following procedure, to set the `alwaysActivateAllMDBs` property to `False`.

1. Log on to the administrative console.
2. Go to **Resources> JMS > Activation specifications**.
3. Click the activation specification corresponding to the application from the list.

4. Click **Custom properties** under **Additional properties**.
5. Click **alwaysActivateAllMDBs**.
6. Change the value to **False**.
7. Click **Apply** and **OK**.

Result

The endpoint application of only the active adapter instance listens to the events.

Solutions to common problems

Some of the problems you might encounter while running WebSphere Adapter for Flat Files with your database are described along with solutions and workaround. These problems and solutions are in addition to those documented as technotes on the software support website.

For a complete list of technotes about WebSphere Adapters, see http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm.

Adapter throws an exception requesting users to migrate their existing tables

Problem

When you install the version 7.5 of the adapter without migrating the existing event table structure with the same name as specified in the activation specification properties, the adapter throws an exception error at run time.

Solution

Migrate the existing event table while retaining the same table name as provided in the previous version of the activation specification properties. You can use the database scripts available at `<IID-HOME>\Resource Adapters\FlatFile_7.5.0.0\SQLScripts`, to migrate the existing event table.

Use the script corresponding to your database:

- `scripts_db2_upgrade.sql` – for DB2 and Derby database
- `scripts_mssql_upgrade.sql` – for Microsoft SQL Server database
- `scripts_oracle_upgrade.sql` – for Oracle database

For more information, see “Migrating databases” on page 47.

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 website

The WebSphere Adapters software support website at http://www-947.ibm.com/support/entry/portal/Overview/Software/WebSphere/

WebSphere_Adapters_Family provides links to many resources to help you learn about, use, and troubleshoot WebSphere Adapters, including:
- Flashes (alerts about the product)
- Technical information including the product information center, manuals, IBM Redbooks®, and whitepapers
- Educational offerings
- Technotes

Recommended fixes

A list of recommended fixes you must 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 WebSphere Adapter for Flat Files, 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

WebSphere Adapter for Flat Files 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 local file system. The name typically reflects the operation to be performed and the structure of the business object.

Business object structures

The WebSphere Adapter for Flat Files defines and generates business objects during external service. The business object structure is based on the generic WebSphere Business Integration business object structure, which is modeled as a base XML schema.

Generic FlatFileBG object

Two types of business objects are generated during enterprise metadata discovery: content-specific and generic.

The generic FlatFileBG business object is used for generic XSD files (for example, UnstructuredContent). The FlatFileBG business object is a wrapper business object that contains the FlatFile business object as a child. The following graphic illustrates this relationship:

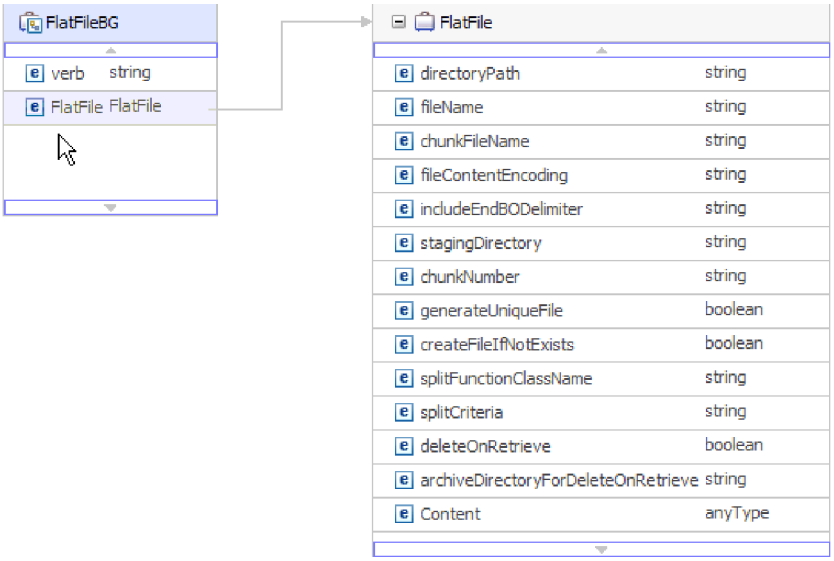


Figure 61. The generic FlatFileBG business object structure

CustomerWrapperBG object

In this example, CustomerWrapperBG represents a content-specific XSD file. The CustomerWrapperBG is a wrapper business object that contains the CustomerWrapper business object as a child. The following graphic illustrates this relationship:

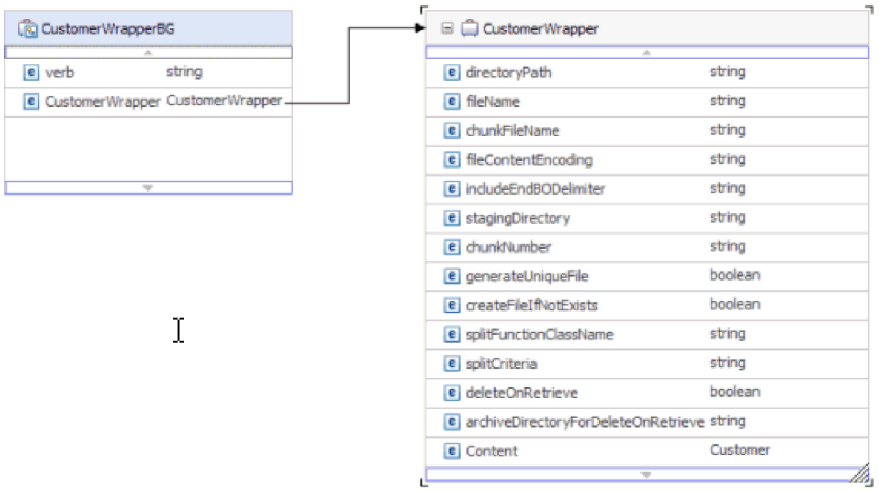


Figure 62. The CustomerWrapperBG business object structure

Append operation with business graph response business object

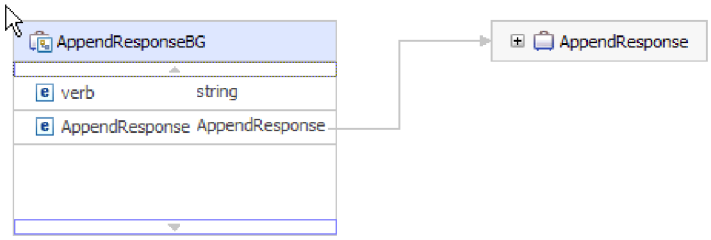


Figure 63. Structure of the Append operation response business object

Create operation with business graph response business object

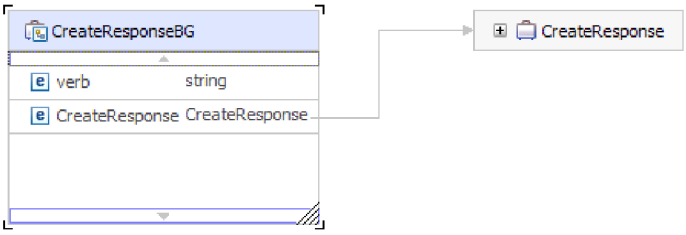


Figure 64. Structure of the Create operation response business object

Delete operation with business graph response business object

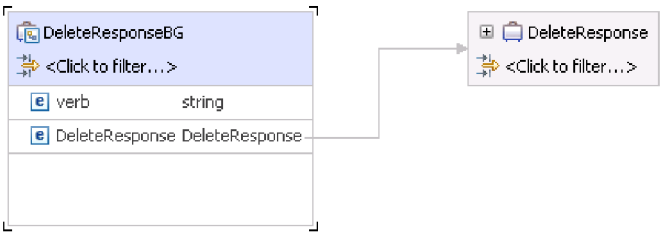


Figure 65. Structure of the Delete operation response business object

Exists operation with business graph response business object

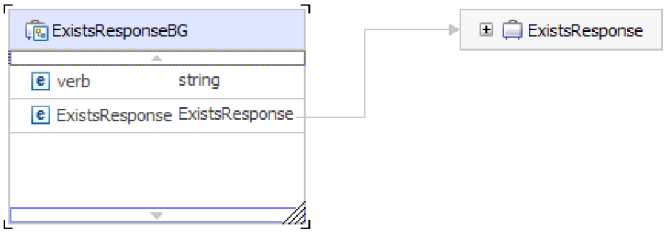


Figure 66. Structure of the Exists operation response business object

List operation with business graph response business object

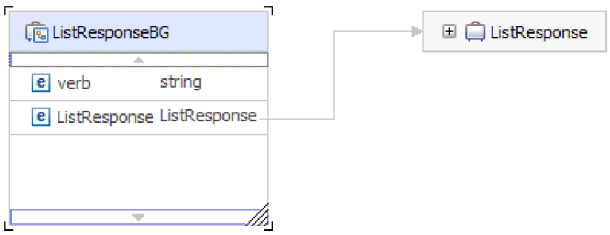


Figure 67. Structure of the List operation response business object

Overwrite operation with business graph response business object

Figure 68. Structure of the Overwrite operation response business object

Retrieve operation with business graph response business object

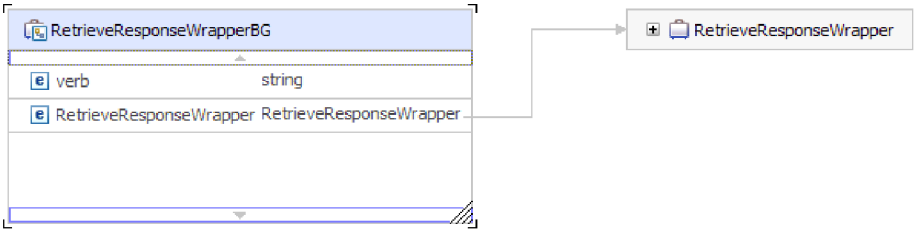


Figure 69. Structure of the Retrieve operation response business object

Global elements in a structured business object

The WebSphere Adapter for Flat Files supports global elements in structured business objects. Global elements with null namespace are also supported.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ff="http://www.ibm.com/xmlns/prod/websphere/j2ca/ff/customer"
            targetNamespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/ff/customer">

<xsd:element name="CustomerType1">
    <xsd:complexType>
        <xsd:sequence>
               <xsd:element name="name" type="xsd:string"></xsd:element>
                 <xsd:element name="address" type="xsd:string"></xsd:element>
                 <xsd:element name="city" type="xsd:string"></xsd:element>
             </xsd:sequence>
        </xsd:complexType>
</xsd:element>
</xsd:schema>
```

Figure 70. Structure of the global elements in a structured Business Object

The CustomerType1 is the global element in the above business object.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema elementFormDefault="qualified"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ff="http://www.ibm.com/xmlns/prod/websphere/j2ca/ff/customer"
            targetNamespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/ff/customer">

<xsd:element name="CustomerInventory" type="ff:CustomerInventoryType3"/>

<xsd:complexType name="CustomerInventoryType3">
    <xsd:sequence>
      <xsd:element name="shipTo" type="xsd:string"/>
      <xsd:element name="billTo" type="xsd:string"/>
      <xsd:element name="items"  type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

The CustomerInventory is the global element in the above 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.

Table 12. Attribute properties

Attribute property	Description
Cardinality	Each business object attribute that represents a child or an array of child business objects has the value of single (1) or multiple (n) cardinality. Only single cardinality flat business objects are supported.
Key and foreign key	These attributes are not used by the adapter.
Name	Represents the unique name of the attribute.
Required	This attribute is not used by the adapter.
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 another business object type.

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 local file system that it uses to build the business object.

When the external service wizard provides a name for the business object, it 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 local file system 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, that is, if you change the business object name, the behavior of the business object remains the same.

Important: If you want to rename a business object, use the refactoring functionality in IBM Integration Designer to ensure that you update all 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/v7r5mx/topic/com.ibm.wbpm.wid.bpel.doc/selector/topics/trefacts.html.

The following table describes the naming conventions that the external service wizard uses when it generates business objects for the WebSphere Adapter for Flat Files.

Table 13. Naming conventions

Element	Naming convention	Example
Name of the business graph	The business graph that contains the parent business object is named for the contained business object, followed by the string `BG`. There can be a business graph only if there is a wrapper. CustomerWrapperBG is a wrapper business object that contains the CustomerWrapper business object as a child.	CustomerWrapperBG

Note: Business graph generation is optional and is supported for IBM Business Process Manager or WebSphere Enterprise Service Bus only.

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 IBM Business Process Manager and WebSphere Enterprise Service Bus support other types of faults, the adapter generates only business faults, which are called *faults* in this documentation. Not all exceptions become faults. 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.

Fault business objects

The external service wizard creates a business object for each fault that the adapter can generate. Also, 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 71.

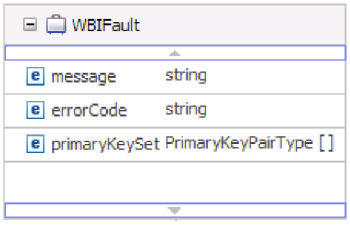


Figure 71. The structure of the WBIFault business object

Some faults contain the matchCount attribute, to provide additional information about the error. For others, WBIFault contains all the information needed to handle the fault.

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

- DuplicateRecordFault

  This fault is generated during the outbound Create operation when the file exists in the specified directory.
- RecordNotFoundFault

  This fault is generated during Append, Delete, Overwrite, and Retrieve operations when the file does not exist in the specified directory.
- MissingDataFault

  If the business object that is passed to the outbound operation does not have all the required attributes, the adapter returns this fault. This fault can occur for the Create, Delete, Update, Retrieve, ApplyChanges and Exists operations.

  For example, the adapter throws this fault if the content of the specified file is null, or the file name or directory path is empty.

Custom file splitting

You can implement a custom class containing the splitting logic. The adapter provides a Java interface for the class. The version 7.5 of the adapter provides interfaces for both outbound and inbound processes.

Interface for outbound operations

Use the `com.ibm.j2ca.utils.filesplit.SplittingFunctionalityInterface` interface for the outbound operations.

Following are the details of the interface:

```
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 objects present in the event file given by `filename`.
- `public void setSplitCriteria(String splitCriteria)`

  This method takes the `splitCriteria`, that is based on the number of business objects 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 three methods: hasNext(), next and remove(), that also must 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, and only clean up the resources that are being used.
- public boolean isSplitBySize()

  This method indicates whether the event file is parsed based on size or based on delimiter.

Interface for inbound operations

Use the com.ibm.j2ca.utils.filesplit.InboundSplittingFunctionalityInterface interface for the inbound operations.

Note: The custom splitting class for an inbound operation created in the earlier version of the adapter does not work with the version 7.5 of the adapter.

Following are the details of the interface:

```
public interface InboundSplittingFunctionalityInterface{
 public Hashtable getBOs(String filename,int quantity, long lastBO,long lastBOPos,boolean withDelim)
 public void setBODetails(String filename, long currentBO, long startPos, long endPos) throws Splitt
 public Object getBOContent();
 public boolean hasMoreBO();
 public void remove();
 public void setSplitCriteria(String splitCriteria);
 public void setEncoding(String encoding);
 public void setLogUtils(LogUtils logUtils);
 public boolean isSplitBySize();
}
```

- public Hashtable getBOs(String filename, int quantity, long lastBOCount, long lastBOPos, boolean includeEndBODelimiter) throws SplittingException, MissingDataException

  This method returns the business objects retrieved from the file specified in the filename to a hashtable. The quantity parameter specifies the number of business objects to be retrieved. The lastBOCount parameter specifies the number of business objects retrieved when the file was previously read. The lastBOPos parameter specifies the end position of the business object when the file was previously read. The includeEndBODelimiter parameter specifies the split criteria to retrieve the business objects. The hashtable that is returned contains the business object count (key) and the start/end positions of that business object (a long array of 2 elements).
- public void setBODetails(String filename, long currentBO, long startPosition, long endPosition)throws SplittingException

  This method is used to set the details for the current business object. Therefore, the getBOContent() method retrieves the content of the business object specified in the currentBO. The startPosition and endPosition parameters specify the start and end position for the business object in the file.
- public Object getBOContent()

The `getBOContent()` method returns the business object content (as a byte[]) for the details set in `setBODetails()` method. If the start and end position of the business object is not set in the `setBODetails()` method, then the `getBOContent()` method fails.

- `public boolean hasMoreBO()`

  This method returns the value `True` if there are unread business objects existing in the file after the last call to the `getBOs()` method.

- `public void remove()`

  The `remove()` method is called for each of the business object entry being deleted from the Event persistence table. Ensure that you do not delete the event file and only clean up the resources that are being used.

- `public void setSplitCriteria(String splitCriteria)`

  This method returns the `splitCriteria` set based on the number of business objects 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 used to write trace and log messages to the files.

- `public void setEncoding(String encoding)`

  This method is used to set the encoding for the content of the event file. This encoding is used while reading the file content and for the `SplitCriteria`.

- `public boolean isSplitBySize()`

  This method returns the value `True` when the event file is parsed based on the size, and returns the value `False` if a delimiter is used.

Related concepts

"Inbound processing" on page 13
WebSphere Adapter for Flat Files supports inbound event processing. It polls the local file system at specified intervals for events, such as the creation of a file. When it detects an event, it converts the event data into a business object and sends it to the module for processing.

Related tasks

"Configuring the module for inbound processing" on page 97
To configure a module to use the adapter for inbound processing, use the external service wizard in IBM Integration Designer to build business services, specify data transformation processing, and generate business object definitions and related artifacts.

Configuration properties

IBM WebSphere Adapter for Flat Files has several categories of configuration properties, which you set with the external service wizard while generating or creating objects and services. You can change the resource adapter, managed connection factory, and activation specification properties after you deploy the application to IBM Business Process Manager or WebSphere Enterprise Service Bus.

Outbound configuration properties

IBM WebSphere Adapter for Flat Files 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 IBM Business Process Manager or WebSphere Enterprise Service Bus using IBM

Integration Designer 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 Flat Files 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	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. Removing a default value from a required field on the external service wizard *will not change that default value*. When a required field contains no value at all, the external service wizard processes the field using its assigned default value, and that default value is displayed on the administrative console. Possible values are **Yes** and **No**. 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, • 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	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. The word `None` is an acceptable default value, and does not mean that there is no default value.
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	Describes the property type. Valid property types include: • Boolean • String • Integer

Row	Explanation
Usage	Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented: For Rational® Application Developer for WebSphere Software version 6.40 or earlier, the password: • Must be uppercase • Must be 8 characters in length For versions of Rational Application Developer for WebSphere Software later than 6.40, the password: • Is not case sensitive • Can be up to 40 characters in length. 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.
Example	Provides sample property values, for example: "If Language is set to JA (Japanese), code page number is set to `8000`".
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 refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file. Valid values are **Yes** and **No**.

Connection properties for the wizard

Connection properties are used to build a service description and save the built-in artifacts. These properties are configured in the external service wizard.

The following table lists the connection properties for the external service wizard. These properties can only be configured using the external service wizard and cannot be changed after deployment. A complete description of each property is provided in the sections that follow the table. For information about how to read the property detail tables in the sections that follow, see “Guide to information about properties” on page 168.

Table 14. Connection properties for the external service wizard

Property name in the wizard	Description
“Bidi format string” on page 170	The bidi format string of the content data.
“Data binding” on page 170	Specifies the data binding that is to be used for all operations or specifies that a data binding is to be selected for each operation.
“Log file output location” on page 170	The full path name of the log file generated by the external service wizard.
“Logging level” on page 170	Specifies the level of logging to be used by the adapter.
“NameSpace” on page 171	The namespace of the business object that is generated.
“Operation name” on page 171	The operation defined in the external service wizard.

Table 14. Connection properties for the external service wizard (continued)

Property name in the wizard	Description
"Processing Direction" on page 171	The processing direction, `Inbound` or `Outbound`.

Bidi format string

The bidi format string of the content data.

Table 15. Bidi format string

Required	No
Default	None
Property type	String

Data binding

Specifies the data binding that is to be used for all operations or specifies that a data binding is to be selected for each operation.

Table 16. Data binding details

Required	No
Default	Use default data binding 'FlatFileBaseDataBinding' for all operations
Usage	The value of this property can be: • Use default data binding 'FlatFileBaseDataBinding' for all operations • Use a data binding configuration for all operations • Specify a data binding for each operation
Globalized	No
Bidi supported	No

Log file output location

The full path name of the log file generated by the external service wizard.

Table 17. Log file output location details

Required	No
Default	`\.metadata \FlatFileMetadataDiscoveryImpl.log`
Property type	String
Usage	
Globalized	No
Bidi supported	No

Logging level

Specifies the level of logging to be used by the adapter.

Table 18. Logging level details

Required	No

Table 18. Logging level details (continued)

Possible values	`Severe`
	`Warning`
	`Audit`
	`Info`
	`Config`
	`Detail`

Default	`Severe`
Property type	List of values
Globalized	No
Bidi supported	No

NameSpace

The namespace of the business object that is generated.

Table 19. NameSpace details

Required	Yes
Default	http://www.ibm.com/xmlns/prod/websphere/j2ca/flatfile
Property type	String
Globalized	Yes
Bidi supported	No

Operation name

The name you give to the operation defined for this module.

Table 20. Operation name details

Required	No
Default	When the ServiceType property is set to `Outbound`, the operations listed are Create, Append, Retrieve, Delete, List, Overwrite, and Exists.
Property type	String
Globalized	No
Bidi supported	No

Processing Direction

The processing direction, `inbound` or `outbound`.

Table 21. Processing Direction details

Required	Yes
Possible values	`Outbound` `Inbound`
Default	`Outbound`
Property type	String
Globalized	No
Bidi supported	No

Related concepts

"Globalization and bidirectional data transformation" on page 214
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 script data transformation, that refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file.

Related tasks

"Configuring logging properties" on page 146
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.

Managed connection factory properties

Managed connection factory properties specify information the adapter needs at run time for outbound communication with the local file system.

The following table lists the managed connection factory properties for outbound communication. You set the managed connection factory properties using the external service wizard and can change them using the IBM Integration Designer Assembly Editor, or after deployment through the IBM Business Process Manager or WebSphere Enterprise Service Bus administrative console.

A more detailed 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 168.

Note: The external service wizard refers to these properties as managed connection factory properties and the IBM Business Process Manager or WebSphere Enterprise Service Bus administrative console refers to them as (J2C) connection factory properties.

Table 22. Managed connection factory properties

Property name		
In the wizard	**In the administrative console**	**Description**
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.
"Default target file name" on page 174	OutputFileName	Specifies the name of the file that is created in the output directory, or a WebSphere Application Server environment variable that represents this file.
Disguise user data as "XXX" in log and trace files	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
"Generate a unique file" on page 174	GenerateUniqueFile	Specifies that the adapter creates a unique file during Create, Append, and Overwrite operations.
"Prefix for the unique file name" on page 175	uniqueFilePrefix	Specifies the predefined prefix that is added to the generated unique file name during outbound operations.
"Output directory" on page 175	OutputDirectory	Specifies the full path name of the directory where the adapter creates files during outbound operations, or a WebSphere Application Server environment variable that represents this directory.

Table 22. Managed connection factory properties (continued)

Property name		
In the wizard	**In the administrative console**	**Description**
"Sequence file" on page 175	FileSequenceLog	Specifies the full path name of the file where sequences are stored during outbound Create operations, or a WebSphere Application Server environment variable that represents this file.
"Staging directory" on page 176	StagingDirectory	The full path name of the temporary directory where the adapter writes the initial output files for Create and Overwrite operations during outbound processing, or a WebSphere Application Server environment variable that represents this directory.
"Suffix for the unique file name" on page 177	uniqueFileSuffix	Specifies the predefined suffix that is added to the generated unique file name during outbound operations.

Adapter ID (AdapterID)

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

Table 23. Adapter ID details

Required	Yes
Default	`001`
Property type	String
Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `FFRA`, 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 `FFRA001`. If you run multiple instances of the same adapter, ensure that the first nine characters of the adapter ID property are unique for each instance so that you can correlate the 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 the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for Flat Files to `001` and `002`. The component IDs for those instances, `FFRA001` and `FFRA002`, 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. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `FFRAInstance0`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use 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 IBM Integration Designer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	Yes
Bidi supported	No

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

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 24. 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, the value can be set both at the resource adapter level and the managed connection factory level. After you use 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 IBM Integration Designer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	No
Bidi supported	No

Default target file name

The name of the file that is created in the output directory, or a WebSphere Application Server environment variable that represents this file.

Table 25. Default target file name details

Required	No
Default	None
Property type	String
Usage	If a value for OutputFileName is specified in the record object, this value is overridden. You can use a WebSphere Application Server environment variable to represent the default target file name. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${OUTPUT_FILENAME}`. See the topic on creating an environment variable in this documentation.
Globalized	Yes
Bidi supported	Yes

Generate a unique file

Specifies that the adapter creates a unique file during the Create operation.

Table 26. Generate unique file property details

Required	No
Possible values	`True` `False`

Table 26. Generate unique file property details (continued)

Default	`False`
Property type	Boolean
Usage	During Create operations, if this property is set to `True`, the adapter creates a unique file and ignores any value set for the Filename property. **Note:** If a value is not specified for this property on the wrapper, then the value specified here is used.
Globalized	Yes
Bidi supported	No

Prefix for the unique file name

The prefix for the unique file name that you predefine for outbound operations.

Table 27. Prefix for the unique file name details

Required	No
Default	None
Property type	String
Usage	You can define the prefix to be added when the adapter generates the unique file name during the outbound operations. The length of the prefix must be of minimum three letters. For example, `abc` can be used as a predefined prefix for the unique file name. **Note:** If you do not specify the prefix, the adapter automatically adds the prefix `ffa` to the file name.
Globalized	No
Bidi supported	No

Output directory

The full path name of the directory where the adapter creates files during outbound operations, or a WebSphere Application Server environment variable that represents this directory.

Table 28. Output directory details

Required	No
Default	None
Property type	String
Usage	The output directory is used by the adapter to write the final output files. 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: `${OUTPUT_DIRECTORY}`. See the topic on creating an environment variable in this documentation.
Globalized	Yes
Bidi supported	Yes

Sequence file

This property specifies the full path name of the file where sequences are stored during outbound Create operations, or a WebSphere Application Server environment variable that represents this file.

Table 29. Sequence file details

Required	No
Default	None
Property type	String
Usage	When the adapter receives a Create request, it checks the file sequence log to see whether a file with the specified name exists. If there is a file with that name, the adapter uses the file sequence number to create a file name. For example, if the output file name in the request is Customer.txt, the adapter creates a file with the name Customer.*n*.txt, where *n* is the sequence number. If the output file name does not have an extension, the sequence is appended at the end of the file name; for example, Customer*n*. All sequences begin with 1. If this property is not specified, and the adapter receives a request to create a file with a name that exists, the adapter generates a `DuplicateRecordException` error. The sequence number continues to increment after an adapter restart. You can reset the file sequence by changing the sequence value in the sequence file. **Notes®:** 1. To generate file sequencing for a particular request type, set the output directory and the file name at the managed connection level. 2. When the adapter is working in a clustered environment, make sure that the sequence file is on a mapped drive that is accessible by all the clusters. The adapter must have write permission for the sequence log file, or an IOException error is returned. **Note:** In the Windows operating systems, such as, Windows 7, Windows Vista, and Windows Server 2008, there are issues faced in the mapped drive connection. Due to this issue, in a clustered environment, where the nodes are running on different machines, the files in the mapped event directory might not be processed correctly during outbound operations. For more information about working with mapped drives, refer to articles on mapped drive connection to network sharing, for your operating system. 3. If the FileSequenceLog property is specified and the GenerateUniqueFile property is enabled, the GenerateUniqueFile property takes precedence over the FileSequenceLog property. 4. The directory path and file name, if specified in the business object, take precedence over the values specified at the managed connection level. You can use a WebSphere Application Server environment variable to represent the sequence file. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${SEQUENCE_FILE}`. See the topic on creating an environment variable in this documentation. **Important:** Unless they are part of a cluster, two adapter instances must not use the same sequence file, because it might result in delayed processing of batch requests.
Globalized	Yes
Bidi supported	Yes

Staging directory

The full path name of the temporary directory where the adapter writes the initial output files for Create and Overwrite operations during outbound processing, or a WebSphere Application Server environment variable that represents this directory.

Table 30. Staging directory details

Required	No
Default	None
Property type	String

Table 30. Staging directory details (continued)

Usage	If this property is specified, the output file is first written to a staging directory and then renamed to the output directory. The adapter temporarily stores the initial output files for Create and Overwrite operations in the staging directory to avoid write conflicts during outbound processing. 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: `${STAGING_DIRECTORY}`. See the topic on creating an environment variable in this documentation.
Globalized	Yes
Bidi supported	Yes

Suffix for the unique file name

The prefix for the unique file name that you predefine for outbound operations.

Table 31. Suffix for the unique file name details

Required	No
Default	None
Property type	String
Usage	You can define the suffix (file extension) to be added when the adapter generates the unique file name during the outbound operations. **Note:** If you do not specify the file extension, the adapter automatically adds the `.tmp` extension to the file name.
Globalized	No
Bidi supported	No

Related concepts

"WebSphere Application Server environment variables" on page 31
WebSphere Application Server environment variables can be used in the external service wizard to specify directory values.

"Creating the required local folders" on page 60
Before you create inbound or outbound modules, you must create folders on the local file system for events and output. You can optionally create folders for staging and archiving.

Related tasks

"Defining WebSphere Application Server environment variables" on page 62
Use the administrative console of IBM Business Process Manager or WebSphere Enterprise Service Bus to define WebSphere Application Server environment variables.

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 6.1.0. They are visible from the administrative console 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 detail tables in the sections that follow, see "Guide to information about properties" on page 168.

Table 32. Resource adapter properties for the WebSphere Adapter for Flat Files

Name		
In the wizard	**In the administrative console**	**Description**
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.
Disguise user data as "XXX" in log and trace files	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
(Not available)	Enable HA support	Do not change this property.
(Not available)	LogFileSize	Deprecated
(Not available)	LogFilename	Deprecated
(Not available)	LogNumberOfFiles	Deprecated
(Not available)	TraceFileSize	Deprecated
(Not available)	TraceFileName	Deprecated
(Not available)	TraceNumberOfFiles	Deprecated

Adapter ID (AdapterID)

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

Table 33. Adapter ID details

Required	Yes
Default	`001`
Property type	String

Table 33. Adapter ID details (continued)

Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `FFRA`, 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 `FFRA001`. If you run multiple instances of the same adapter, ensure that the first nine characters of the adapter ID property are unique for each instance so that you can correlate the 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 the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for Flat Files to `001` and `002`. The component IDs for those instances, `FFRA001` and `FFRA002`, 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. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `FFRAInstance0`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use 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 IBM Integration Designer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	Yes
Bidi supported	No

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

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 34. 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, the value can be set both at the resource adapter level and the managed connection factory level. After you use 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 IBM Integration Designer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	No

Table 34. Disguise user data as "XXX" in log and trace files details (continued)

Bidi supported	No

Enable high availability support (enableHASupport)

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

Interaction specification properties

Interaction specification properties contain the outbound connection properties the adapter uses to interface with the file system. You configure these properties using the external service wizard. To change the interaction specification properties after the application has been deployed, use the assembly editor in IBM Integration Designer.

Interaction specification properties control the interaction for an operation. The external service wizard sets the interaction specification properties when you configure the adapter. Typically, you do not need to change these properties. However, some properties for outbound operations can be changed by the user. To change these properties after the application is deployed, use the assembly editor in IBM Integration Designer. The properties reside in the method binding of the import.

The following table lists the 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 detail tables in the sections that follow, see "Guide to information about properties" on page 168.

Table 35. Interaction specification properties

Property name		
In the wizard	**In the administrative console**	**Description**
"Archive directory for retrieve operation" on page 181	ArchiveDirectoryFor DeleteOnRetrieve	The directory where retrieved files are stored before they are deleted, if the DeleteOnRetrieve property is set to `true`.
"Create a new file if the file does not exist" on page 181	CreateFileIfNotExists	When this property is set to `true`, the adapter creates a file during Append and Overwrite operations if the file does not exist.
"Default target file name" on page 182	OutputFileName	The name of the output file that is created or modified.
"Delete the file after retrieve operation" on page 182	DeleteOnRetrieve	During Retrieve operations, when this property is set to `true`, the file is deleted from the file system after the file content has been retrieved.
"Delimiter between business objects in the file" on page 182	IncludeEndBODelimiter	The file content is appended with this value.
"File content encoding" on page 183	FileContentEncoding	Specifies the encoding set used in writing to or reading from the event file.
"Generate a unique file" on page 183	GenerateUniqueFile	Specifies that the adapter creates a unique file during Create, Append, and Overwrite operations.
"Output directory" on page 183	OutputDirectory	The full path name of the directory on the local file system where the adapter writes the output files.

Table 35. Interaction specification properties (continued)

Property name		
In the wizard	**In the administrative console**	**Description**
"Prefix for the unique file name" on page 184	uniqueFilePrefix	The predefined prefix that is added to the generated unique file name during outbound operations.
"Specify criteria to split file content" on page 184	SplitCriteria	Specifies either the delimiter that separates the business objects in the retrieved file or the size of the chunks into which the retrieved file is split.
"Split function class name" on page 185	SplittingFunctionClassName	Specifies how the retrieved file is to be split, by delimiter or by size, during an outbound Retrieve operation.
"Staging directory" on page 185	StagingDirectory	A temporary directory where the adapter stores the initial output files during Create and Overwrite operations.
"Suffix for the unique file name" on page 186	uniqueFileSuffix	The predefined suffix that is added to the generated unique file name during outbound operations.

Archive directory for retrieve operation

The directory where retrieved files are stored before they are deleted, if the DeleteOnRetrieve property is set to `true`.

Table 36. Archive directory for retrieve operation details

Required	No
Default	None
Property type	String
Globalized	Yes
Bidi supported	Yes

Create a new file if the file does not exist

When this property is set to `true`, the adapter creates a file during Append and Overwrite operations if the file does not exist.

Table 37. Create a new file if the file does not exist details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	When this property is set to `false` and the file does not exist, the adapter generates the RecordNotFoundException error. **Note:** If a value is not specified for this property on the wrapper, then the value specified here is used.
Globalized	No
Bidi supported	No

Default target file name

The name of the output file that is created or modified.

Table 38. Default target file name details

Required	Required for all outbound operations except List
Default	None
Property type	String
Globalized	Yes
Bidi supported	Yes

Delete the file after retrieve operation

During Retrieve operations, if this property is set to `true`, the file is deleted from the file system after the file content has been retrieved.

Table 39. Delete the file after retrieve operation details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	To archive the file before it is deleted, specify a directory in the ArchiveDirectoryForDeleteOnRetrieve property. **Note:** If a value is not specified for this property on the wrapper, then the value specified here is used.
Globalized	No
Bidi supported	No

Delimiter between business objects in the file

The file content is appended with this value.

Table 40. Delimiter between business objects in the file details

Required	No
Default	`<EndBO>` for Append operation None for Create and Overwrite operations
Property type	String
Usage	This property is used during outbound Create, Append, and Overwrite operations. Any value specified in this property is appended to the file. If the value specified have escape sequence characters and Unicode escape characters, they are parsed and the corresponding control characters are inserted in the file. The escape sequence characters include the following: carriage return (\r), new line (\n), carriage return and new line (\r\n), tab space (\t), backspace (\b), form feed (\f), and so on. An example of a Unicode escape character that represents the character **?** is **\u2297**.
Globalized	Yes
Bidi supported	No

File content encoding

The encoding set used when writing to or reading from the event file.

Note: During the Create operation, the adapter creates the file with the specified encoding.

Table 41. File content encoding details

Required	No
Possible values	Any Java supported encoded character sets.
Default	`UTF-8`
Property type	String
Usage	You can specify any Java supported encoding set, such as UTF-8. If the adapter is working with binary event data, set this property to `BINARY`. If the adapter is working with non-binary event data, such as text or XML, set this property to a valid file encoding value, such as UTF-8 or UTF-16. **Note:** The value set in the interaction specification property is used only if no value is set on the wrapper.
Globalized	No
Bidi supported	No

Generate a unique file

Specifies that the adapter creates a unique file during the Create operation.

Table 42. Generate unique file property details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	During Create operations, if this property is set to `True`, the adapter creates a unique file and ignores any value set for the Filename property. **Note:** If a value is not specified for this property on the wrapper, then the value specified here is used.
Globalized	Yes
Bidi supported	No

Output directory

The full path name of the directory on the local file system where the adapter writes the output files.

Table 43. Output directory details

Required	No
Default	None
Property type	String
Usage	If this property is not specified, the adapter writes the output files to the directory specified by the OutputFileName property on the request.
Globalized	Yes

Table 43. Output directory details (continued)

Bidi supported	Yes

Prefix for the unique file name

The prefix for the unique file name that you predefine for outbound operations.

Table 44. Prefix for the unique file name details

Required	No
Default	None
Property type	String
Usage	You can define the prefix to be added when the adapter generates the unique file name during the outbound operations. The length of the prefix must be of minimum three letters. For example, `abc` can be used as a predefined prefix for the unique file name. **Note:** If you do not specify the prefix, the adapter automatically adds a prefix `ffa` for the file name.
Globalized	No
Bidi supported	No

Specify criteria to split file content

This property specifies either the delimiter that separates the business objects in the retrieved file, or the size of the chunks into which the retrieved file is split.

Table 45. Specify criteria to split file content details

Required	No
Possible values	A delimiter or a valid number
Default	`0`
Property type	String

Table 45. Specify criteria to split file content details (continued)

Usage	This property specifies either the delimiter that separates the business objects in the retrieved file, or the size of the chunks into which the retrieved file is split. The value of this property is determined by the value that is set in the SplittingFunctionClassName property: • If the SplittingFunctionClassName property is set to `com.ibm.j2ca.utils.filesplit.SplitByDelimiter`, the SplitCriteria property must contain the delimiter that separates the business objects in the retrieved file. • If the SplittingFunctionClassName property is set to `com.ibm.j2ca.utils.filesplit.SplitBySize`, the SplitCriteria property must contain a valid number that represents the size in bytes. If the retrieved file size is greater than this value, it is split into chunks of this value and that number of chunks are posted. If the file size is less than this value, the entire event file is posted. If the SplitCriteria property is set to 0, chunking is disabled. The SplitCriteria property must contain the same value for the newline character as the event file. For example, if the event file was created on a Macintosh system, the newline character is `\r`, and the SplitCriteria property must contain `\r`. The platform-specific newline characters are as follows: Macintosh - **\r** Microsoft Windows - **\r\n** UNIX - **\n** If there is more than one delimiter in the SplitCriteria property, each delimiter must be separated by a semicolon (:). If a semicolon (;) itself is part of the delimiter, the semicolon (;) must be escaped, as in `\;`. For example, if the delimiter given is `##\;##`. it is evaluated to `##;##`.
Globalized	Yes
Bidi supported	Yes

Split function class name

This property specifies how the retrieved file is to be split, by delimiter or by size, during an outbound Retrieve operation.

Table 46. Split function class name details

Required	No

Possible values	`com.ibm.j2ca.utils.filesplit.SplitByDelimiter`
	– Files are split based on a delimiter that separates business objects in the event file
	`com.ibm.j2ca.utils.filesplit.SplitBySize`
	– Files are split based on the size of the event file

Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Usage	The delimiter or file size is set in the SplitCriteria property.
Globalized	No
Bidi supported	No

Staging directory

A temporary directory where the adapter stores the initial output files during Create and Overwrite operations to avoid write conflicts.

Table 47. Staging directory details

Required	No

Table 47. Staging directory details (continued)

Default	None
Property type	String
Usage	If a staging directory is specified, the file to be operated is copied from the output directory to the staging directory. The operation is performed on the file in the staging directory, and the file is then renamed and copied to the output directory.
Globalized	Yes
Bidi supported	Yes

Suffix for the unique file name

The suffix for the unique file name that you predefine for outbound operations.

Table 48. Suffix for the unique file name details

Required	No
Default	None
Property type	String
Usage	You can define the suffix (file extension) to be added when the adapter generates the unique file name during the outbound operations. **Note:** If you do not specify the file extension, the adapter automatically adds the `.tmp` extension to the file name.
Globalized	No
Bidi supported	No

Related tasks

Chapter 5, “Changing interaction specification properties,” on page 117
To change interaction specification properties for your adapter module after generating the service, use the assembly editor in IBM Integration Designer.

Inbound configuration properties

WebSphere Adapter for Flat Files 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 IBM Integration Designer 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 Flat Files 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	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. Removing a default value from a required field on the external service wizard *will not change that default value*. When a required field contains no value at all, the external service wizard processes the field using its assigned default value, and that default value is displayed on the administrative console. Possible values are **Yes** and **No**. 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, • 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	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. The word `None` is an acceptable default value, and does not mean that there is no default value.
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	Describes the property type. Valid property types include: • Boolean • String • Integer
Usage	Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented: For Rational Application Developer for WebSphere Software version 6.40 or earlier, the password: • Must be uppercase • Must be 8 characters in length For versions of Rational Application Developer for WebSphere Software later than 6.40, the password: • Is not case sensitive • Can be up to 40 characters in length. 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.
Example	Provides sample property values, for example: "If Language is set to JA (Japanese), code page number is set to `8000`".
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**.

Row	Explanation
Bidi supported	Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file. Valid values are **Yes** and **No**.

Connection properties for the wizard

Connection properties are used to build a service description and save the built-in artifacts. These properties are configured in the external service wizard.

The following table lists the connection properties for the external service wizard. These properties can only be configured using the external service wizard and cannot be changed after deployment. A complete description of each property is provided in the sections that follow the table. For information about how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 168.

Table 49. Connection properties for the external service wizard

Property name in the wizard	Description
"Bidi format string"	The bidi format string of the content data
"Data binding"	Specifies the data binding that is to be used for all operations or specifies that a data binding is to be selected for each operation.
"Function selector" on page 189	During inbound processing, the name of the function selector configuration to be used.
"Log file output location" on page 189	The full path name of the log file generated by the external service wizard
"Logging level" on page 190	The level of logging to be used by the adapter
"NameSpace" on page 190	The namespace of the business object that is generated
"Operation name" on page 190	The operation defined in the external service wizard
"Processing Direction" on page 191	The processing direction, `Inbound` or `Outbound`

Bidi format string

The bidi format string of the content data.

Table 50. Bidi format string

Required	No
Default	None
Property type	String

Data binding

Specifies the data binding that is to be used for all operations or specifies that a data binding is to be selected for each operation.

Table 51. Data binding details

Required	No

Table 51. Data binding details (continued)

Default	Use default data binding 'FlatFileBaseDataBinding' for all operations
Usage	The value of this property can be: • Use default data binding 'FlatFileBaseDataBinding' for all operations • Use a data binding configuration for all operations • Specify a data binding for each operation
Globalized	No
Bidi supported	No

Function selector

During inbound processing, the name of the function selector configuration to be used.

Table 52. Function selector details

Required	Yes
Default	`FilenameFunctionSelector`
Property type	String
Usage	The function selector returns the appropriate operation to be called on the service. The adapter provides two function selectors, `FilenameFunctionSelector` and `EmbeddedNameFunctionSelector`. • `FilenameFunctionSelector` is a rule-based function selector that matches a regular expression on a file name to an object name. Use `FilenameFunctionSelector` for generic FlatFile business objects, where the object name cannot be determined from the event file. `FilenameFunctionSelector` is represented in properties as a two-column table with *N* rows. For any event file with a `.txt` extension, the corresponding object name is FlatFile, and the endpoint method name generated by the function selector is emitFlatFile. You must set this same name in the EISFunctionName property after you add the operation. 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. • Use `EmbeddedNameFunctionSelector` for content-specific business objects, where the object name is embedded in the event file. `EmbeddedNameFunctionSelector` returns the function name based on the content data, and not the wrapper. For example, if the content-specific business object is CustomerWrapperBG, the function returned by the function selector is emitCustomer. You must configure `EmbeddedNameFunctionSelector` 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.
Globalized	Yes
Bidi supported	No

Log file output location

The full path name of the log file generated by the external service wizard.

Table 53. Log file output location details

Required	No
Default	`\.metadata \FlatFileMetadataDiscoveryImpl.log`
Property type	String
Usage	

Table 53. Log file output location details (continued)

Globalized	No
Bidi supported	No

Logging level

The level of logging to be used by the adapter.

Table 54. Logging level details

Required	No

Possible values	`Severe`
	`Warning`
	`Audit`
	`Info`
	`Config`
	`Detail`

Default	`Severe`
Property type	List of values
Globalized	No
Bidi supported	No

NameSpace

The namespace of the business object that is generated.

Table 55. NameSpace details

Required	Yes
Default	http://www.ibm.com/xmlns/prod/websphere/j2ca/flatfile
Property type	String
Globalized	Yes
Bidi supported	No

Operation name

The name you give to the operation defined for this module.

Table 56. Operation name details

Required	No
Default	When the ServiceType property is set to `Outbound`, the operations listed are Create, Append, Retrieve, Delete, List, Overwrite, and Exists.
Property type	String
Globalized	No
Bidi supported	No

Processing Direction

The processing direction, `inbound` or `outbound`.

Table 57. Processing Direction details

Required	Yes
Possible values	`Outbound` `Inbound`
Default	`Outbound`
Property type	String
Globalized	No
Bidi supported	No

Related concepts

"Function selectors" on page 22
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 three function selectors, `FilenameFunctionSelector`, `EmbeddedNameFunctionSelector`, and `RootNameFunctionSelector`.

Activation specification properties

Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

The following activation specification properties are no longer required from version 6.1.0, but are supported for compatibility with previous versions.

- ArchivingProcessed
- DefaultObjectName
- EventContentType

The following table lists the activation specification properties for inbound communication. You set the activation specification properties using the external service wizard and can change them before deployment by using the IBM Integration Designer Assembly Editor or after deployment through the IBM Business Process Manager administrative console.

A detailed description of each property is provided in the sections that follow the table. For information about how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 168.

Table 58. Activation specification properties

Property name		
In the wizard	**In the administrative console**	**Description**
"Archive directory" on page 194	ArchiveDirectory	The directory where the adapter archives processed event files.
(Not available)	ArchivingProcessed	Deprecated
"Auto create event table" on page 195	EP_Create Table	Determines whether the event persistence table is created automatically or manually.

Table 58. Activation specification properties (continued)

Property name		
In the wizard	**In the administrative console**	**Description**
"Bidirectional transformation of event persistence properties" on page 195	EP_BiDiFormat	Determines whether the adapter transforms any of the event persistence properties.
(Not available)	DefaultObjectName	Deprecated
Delivery type	DeliveryType	Determines the order in which events are delivered by the adapter to the export.
Ensure once only event delivery	AssuredOnceDelivery	Specifies whether the adapter provides assured once delivery of events.
"Database schema name" on page 195	EP_SchemaName	The schema name of the database used by event persistence processing.
(Not available)	EventContentType	Deprecated
"Event directory" on page 197	EventDirectory	The directory where the event files are stored.
"Event recovery data source (JNDI) name" on page 197	EP_DataSource_JNDIName	The JNDI name of the data source used by event persistence processing to obtain the JDBC database connection. The data source must be created in IBM Business Process Manager.
"Event recovery table name" on page 197	EP_TableName	The name of the table used by the adapter for event persistence processing.
Event types to process	EventTypeFilter	A delimited list of event types that indicates to the adapter which events it should deliver.
Retry limit for failed events	FailedEventRetryLimit	The number of times the adapter attempts to redeliver an event before marking the event as failed.
"Failure file extension for archive" on page 198	FailedArchiveExtension	The file extension used to archive unsuccessfully processed business objects in the input event file. This property is applicable only when the SplitByDelimiter file splitting criteria is used.
"File content encoding" on page 199	FileContentEncoding	The encoding of the files that are read by the adapter.
"File extension for archive" on page 199	OriginalArchiveExtension	The file extension used to archive the original event file.
"Include only the newly appended content" on page 199	ProcessFileAppendedContent	Specifies whether to process and deliver only the appended file contents compared to the last polled file contents.

Table 58. Activation specification properties (continued)

Property name		
In the wizard	**In the administrative console**	**Description**
"Include business object delimiter in the file content" on page 200	IncludeEndBO Delimiter	Specifies whether the delimiter value specified in the SplitCriteria property is sent with the business object content for further processing.
"Include total business object count in the ChunkInfo (includeBOCountInChunkInfo)" on page 200	includeBOCountInChunkInfo	When set to `true`, the total business object count is included in the chunk information of the dataobject sent to the endpoint.
Interval between polling periods	PollPeriod	The length of time that the adapter waits between polling periods.
Maximum number of retries in case of system connection failure	RetryLimit	The number of times the adapter tries to reestablish an inbound connection after an error.
"Pass only file name and directory, not the content" on page 202	FilePassByReference	Specifies whether the adapter delivers the file content to the export.
"Password used to connect to event data source" on page 203	EP_Password	The password used by event persistence processing to obtain the JDBC database connection from the data source.
"Poll event files for modified content" on page 202	FileChangeNotification	Specifies whether the adapter polls the files that have changed since the last recorded time stamp.
Poll quantity	PollQuantity	The number of events the adapter delivers to the export during each poll period.
"Poll subdirectories in event directory" on page 203	PollSubDirectories	Specifies whether the adapter polls the subdirectories within the event directory.
"Retrieve files in sorted order" on page 204	SortEventFiles	The sorting order of polled event files.
"Retrieve files with pattern" on page 204	EventFileMask	The file filter for the event files.
Retry connection on startup	RetryConnectionOnStartup	Controls whether the adapter retries the connection to the local file system if it cannot connect at startup.
"Retry interval if connection fails (RetryInterval)" on page 205	RetryInterval	The length of time that the adapter waits between attempts to reestablish connection after an error during inbound operations.
"Specify criteria to split file content" on page 205	SplitCriteria	The delimiter that separates the business objects in the event file or the maximum size of the event file, depending on the value that is set in the Splitting Function Class Name.

Table 58. Activation specification properties (continued)

Property name		Description
In the wizard	**In the administrative console**	
"Split file content based on size (bytes) or delimiter" on page 206	(Not available)	Specifies file splitting either by size or delimiter.
"Split function class name" on page 207	SplittingFunctionClassName	Specifies how the event file is to be split, by delimiter or by size.
"Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 207	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling.
"Success file extension for archive" on page 208	SuccessArchiveExtension	The file extension used to archive successfully processed business objects.
"Table name to store the file processing status (EP_FileTableName) " on page 208	`EP_FileTableName`	The name of the table used to store the file processing status.
"Time interval for polling unchanged files (in milliseconds)" on page 208	FileUnchangedTimeInterval	Specifies whether the adapter retrieves only those files that are not changed during the specified time interval.
"Time out period for HA Active-Active event processing change (in seconds) (EP_Timeout)" on page 209	`EP_Timeout`	Specifies the time interval for processing the events fetched.
"User name used to connect to event data source" on page 209	EP_UserName	The user name used by event persistence processing to obtain the JDBC database connection from the data source.
Rule editor to filter files	ruleTable	The collection of rules used to filter the events.

Archive directory

This property specifies the directory where the adapter archives processed event files.

Table 59. Archive directory details

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${ARCHIVE_DIRECTORY}`. See the topic on creating an environment variable in this documentation. **Note:** You must enter the location of the archive directory, if PassByReference is set to `True`.
Globalized	Yes
Bidi supported	Yes

Auto create event table

This property determines whether the event persistence table is created automatically or manually.

Table 60. Auto create event table details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If this value is set to `True`, the adapter creates the event persistence table. If the value is set to `False`, the adapter does not create the table and you must manually create it. The automatic table creation is supported only for the following databases. • IBM DB2 • Oracle • Microsoft SQL Server • Apache Derby For other databases, you must manually create the event table and the file table.
Globalized	No

Bidirectional transformation of event persistence properties

This property determines whether the adapter transforms any of the event persistence properties.

Table 61. Bidirectional transformation of event persistence properties

Required	No
Possible values	You can specify a string value, such as `VRYNN`.
Default	None
Property type	String
Usage	The value set on the event persistence bidirectional format property (EP_BiDiFormat) determines the bidirectional transformation. You can specify a string value, such as `VRYNN` to enable bidirectional transformation of event persistence properties. If the EP_BiDiFormat property is not specified, the adapter displays a null value. **Note:** You can do bidirectional transformation of only those event properties whose values are set on the bidirectional context enterprise information system (EIS) property.
Globalized	No
Bidi supported	Yes

Database schema name

This property specifies the schema name of the database used by event persistence processing.

Table 62. Database schema name details

Required	No
Default	None
Property type	String

Table 62. Database schema name details (continued)

Globalized	Yes
Bidi supported	Yes

Delivery type (DeliveryType)

This property specifies the order in which events are delivered by the adapter to the export.

Table 63. Delivery type details

Required	No
Possible values	`ORDERED` `UNORDERED`
Default	`ORDERED`
Property type	String
Usage	The following values are supported: • `ORDERED`: The adapter delivers events to the export one at a time. • `UNORDERED`: The adapter delivers all events to the export at once. **Note:** HA Active-Active configuration supports only unordered delivery type events to the export.
Globalized	No
Bidi supported	No

Ensure once-only event delivery (AssuredOnceDelivery)

This property specifies whether to provide ensure once-only event delivery for inbound events.

Table 64. Ensure once-only event delivery details

Required	Yes
Possible values	`True` `False`
Default	`True`
Property type	Boolean
Usage	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. 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. 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.
Globalized	No
Bidi supported	No

Event directory

This property specifies the directory in the local file system where the event files are stored.

Table 65. Event directory details

Required	Yes
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the event directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${EVENT_DIRECTORY}`. See the topic on creating an environment variable in this documentation.
Globalized	Yes
Bidi supported	Yes

Event recovery data source (JNDI) name

This property specifies the JNDI name of the data source used by event persistence processing to obtain the JDBC database connection.

Table 66. Event recovery data source (JNDI) name details

Required	No
Default	None
Property type	String
Usage	The data source must be created in IBM Business Process Manager. Leave this value empty to enable event polling without using the database.
Globalized	Yes
Bidi supported	Yes

Event recovery table name

This property specifies the name of the table to be used by the adapter for event persistence processing.

Table 67. Event recovery table name details

Required	No
Default	No default value
Property type	String
Usage	When multiple activation specification instances are used, this value must be unique for each activation specification instance.
Globalized	Yes
Bidi supported	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 68. 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 events related to the Customer and Order business objects only, specify this value: `Customer,Order`
Globalized	No
Bidi supported	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 69. Retry limit for failed events details

Required	No
Possible values	Integers
Default	`5`
Property type	Integer
Usage	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: **Default** If this property is not set, the adapter tries five additional times before marking the event as failed. **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. **> 0** For integers greater than zero, the adapter retries the specified number of times before marking the event as failed. **< 0** For negative integers, the adapter does not retry failed events.
Globalized	No
Bidi supported	No

Failure file extension for archive

This property specifies the file extension used to archive unsuccessfully processed business objects in the input event file, and applicable only when an event file has failed business objects and file splitting by delimiter is enabled.

Table 70. Failure file extension for archive details

Required	No
Default	`fail`
Property type	String

Table 70. Failure file extension for archive details (continued)

Usage	The event file is archived with the `.fail` extension only when you have specified SplitByDelimiter as the file splitting criteria. When you specify SplitBySize as the file splitting criteria, the file is not archived with the `.fail` extension.
Globalized	Yes
Bidi supported	Yes

File content encoding

This property specifies the encoding of the files read by the adapter.

Table 71. File content encoding details

Required	No
Default	UTF-8
Property type	String
Usage	You can specify any Java-supported encoding set, such as UTF-8. If the FileContentEncoding property is not specified, the adapter uses the default system encoding. If the adapter is working with binary event data, set this property to `BINARY`. If the adapter is working with non-binary event data, such as text or XML, set this property to a valid file encoding value, such as `UTF-8`.
Globalized	No
Bidi supported	No

File extension for archive

This property specifies the file extension used to archive the original event file.

Table 72. File extension for archive details

Required	No
Default	`original`
Property type	String
Usage	This property preserves the entire event file for reference if any of the business objects fail processing.
Globalized	Yes
Bidi supported	Yes

Include only the newly appended content

This property specifies whether to process and deliver only the appended file contents at the end of the file when compared to the last polled file contents.

Table 73. Notification for appended file contents

Required	No
Default	False
Property type	Boolean

Table 73. Notification for appended file contents (continued)

Usage	When you select this property, the adapter processes and delivers only the appended business objects (data) at the end of the file when compared to previous poll contents. If the event file has same or less number of business objects than the last poll, then the file is not processed for delivery to the endpoint. **Note:** When you enable this property, the adapter does not archive or delete any files.
Globalized	No
Bidi supported	No

Include business object delimiter in the file content

This property specifies whether the delimiter value specified in the SplitCriteria property is sent with the business object content for further processing.

Table 74. Include business object delimiter in the file content details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	When this property is set to `true`, the delimiter value specified in the SplitCriteria property is sent with the business object content for further processing. This property is valid only if event file splitting is based on a delimiter; that is, if the SplittingFunctionClassName property is set to `com.ibm.j2ca.utils.filesplit.SplitByDelimiter`. **Note:** • This property has no effect on the inbound processing of the event files. • If the configured delimiter does not exist in the input file, the adapter processes the event file as a single business object. • This property must be used with a custom data binding that can handle end business object delimiter in the contents. Using it with XMLDataHandler results in failure at the data binding level.
Globalized	No
Bidi supported	No

Include total business object count in the ChunkInfo (includeBOCountInChunkInfo)

This property, when set to `true`, specifies that the total business object count is included in the chunk information of the dataobject, which is sent to endpoint.

Table 75. Include total business object count in the ChunkInfo property characteristics

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean

Table 75. Include total business object count in the ChunkInfo property characteristics (continued)

Usage	This property is used for specifying whether the total business object count is included in the chunk information of the dataobject sent to the endpoint.
	
	Format of the chunk information:
	
	When the property is enabled
	`AbsolutePathOfEventFileNameInLocalEventDirectory_`
	`/_YYYY_MM_DD_HH_mm_ss_SSS_`
	`/_currentBONumberofTotalBOCount`
	
	When the property is disabled
	`AbsolutePathOfEventFileNameInLocalEventDirectory_`
	`/_YYYY_MM_DD_HH_mm_ss_SSS.currentBONumber_`
	`/_currentBONumber`

Globalized	No
Bidi supported	No

Interval between polling periods (PollPeriod)

This property specifies the length of time that the adapter waits between polling periods.

Table 76. 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 77. 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 might improve slightly.
Globalized	No

Table 77. Maximum events in polling period details (continued)

Bidi supported	No

Poll event files for modified content

This property specifies whether the adapter polls files that have changed since the last recorded time stamp.

Table 78. Notification for file changes

Required	No
Default	False
Property type	Boolean
Usage	This property is used to retrieve files from the event directory, when a file is changed from the last recorded time stamp. When this property is selected, the adapter polls the new and changed files during each subsequent poll cycle after the previous event poll. Also, the adapter does not delete any event file from the event directory. **Note:** The adapter does not archive any files in the specified archive directory. Also, this property is disabled if you select the FilePassByReference property.
Globalized	No
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 79. 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 local file system 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 local file system when it is first started, use the RetryConnectionOnStartup property.
Globalized	No
Bidi supported	No

Pass only file name and directory, not the content

This property specifies whether to send the directory name and file name to the endpoint.

Table 80. Pass only file name and directory, not the content details

Required	No
Possible values	`True` `False`
Default	`False`

Table 80. Pass only file name and directory, not the content details (continued)

Property type	Boolean
Usage	If this property is set to `True`, the adapter always archives the file and sends the directory name and file name to the endpoint. However, the adapter does not load the content of the file. The event file is appended with a time stamp and archived to the archive directory. For example, if `a.txt` is the event file, it is archived as `a.txt.yyyy_MM_dd_HH_mm_ss_SSS` in the archive directory. Additionally, for COBOL or XMLDataHandler, the event file is archived to `a.txt.yyyy_MM_dd_HH_mm_ss_SSS.original` file. **Note:** • If this property is set to `True` and archive directory is not specified, the adapter throws an exception. This property can be used with a custom data binding that does not fail at run time if no content is set; or it can be used in a pass-through scenario. Using this property with XMLDataHandler results in failure at the data binding level, because XMLDataHandler expects content in addition to the file name and directory path. • In the external service wizard, you can use this property if both the SplitCriteria property and FileChangeNotification properties are not selected. • The administrative console allows you to use this property along with the FileChangeNotification, SplittingFunctionClassName, and SplitCriteria properties, but the event files are not split into chunks. The settings specified in the FileChangeNotification property takes precedence, when used with this property.
Globalized	No

Password used to connect to event data source

This property specifies the password used by event persistence processing to obtain the JDBC database connection from the data source.

Table 81. Password used to connect to event data source details

Required	No
Default	None
Property type	String
Globalized	Yes
Bidi supported	Yes

Poll subdirectories in event directory

This property specifies whether the adapter polls the subdirectories within the event directory.

Table 82. Poll subdirectories in event directory details

Required	No
Default	`False`
Property type	Boolean

Table 82. Poll subdirectories in event directory details (continued)

Usage	When this property is set to `True`, the adapter polls the files in the event directory and also the files in its subdirectories. When this property is set to `False`, the adapter polls only the files in the root directory and ignores any subdirectories. During a poll cycle, the adapter first polls the files in the root directory and then polls the files in the subdirectories. It sorts them according to the value set for the SortEventFiles property and processes them according to the value set for the PollQuantity property. It then sends the business objects to the downstream components. When the PollSubDirectories property is set to `True` and archiving is enabled, all the polled files, including the files that are polled from the subdirectories, are archived to the archive directory.
Globalized	No
Bidi supported	No

Retrieve files in sorted order

This property specifies the sorting order of polled event files.

Table 83. Retrieve files in sorted order details

Required	No
Possible values	`File name` - sort in ascending order on file name `Time stamp`- sort in ascending order on last modified time stamp `No sort`- not sorted
Default	`No sort`
Property type	String
Usage	To support globalization, the sorting of file names is provided according to the system locale. The ICU4J package is used to track the locales and the rules corresponding to the locales. **Note:** The sorting of event files by file name or timestamp is not supported in the HA Active-Active configuration.
Globalized	No
Bidi supported	No

Retrieve files with pattern

This property specifies the file filter for the event files.

Table 84. Retrieve files with pattern details

Required	Yes
Default	*.*
Property type	String
Usage	The file filter is a well-qualified valid regular expression that can consist of alphanumeric characters and the wildcard character "*". *. For example, if you specify `event*`, only file names beginning with `event` are processed.
Globalized	Yes
Bidi supported	Yes

Retry EIS connection on startup (RetryConnectionOnStartup)

This property controls whether the adapter attempts to connect again to the local file system if it cannot connect at startup.

Table 85. Retry EIS connection on startup details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	This property indicates whether the adapter should retry the connection to the local file system if the connection cannot be made when the adapter is started: • Set the property to `False` when you want immediate feedback about whether the adapter can establish a connection to the local file system, 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 `Stopped`. After you resolve the connection problem, start the adapter manually. • Set the property to `True` 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 RetryInterval property to determine how frequently to retry and the value of the RetryLimit property to retry multiple times until that value is reached. The administrative console shows the application status as `Started`.
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 reestablish a connection.

Table 86. Retry interval details

Required	Yes
Default	`2000`
Unit of measure	Milliseconds
Property type	Integer
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	No
Bidi supported	No

Specify criteria to split file content

This property specifies either the delimiter that separates the business objects in the event file or the maximum size of the event file.

Table 87. Specify criteria to split file content details

Required	No
Default	`0`
Property type	String

Usage	This property specifies the delimiter that separates the business objects in the event file or the maximum size of the event file. The value of this property is determined by the value that is set in the SplittingFunctionClassName property:
	• If the SplittingFunctionClassName property is set to `com.ibm.j2ca.utils.filesplit.SplitByDelimiter`, the SplitCriteria property must contain the delimiter that separates the business objects in the event file.
	• If the SplittingFunctionClassName property is set to `com.ibm.j2ca.utils.filesplit.SplitBySize`, the SplitCriteria property must contain a valid number that represents the maximum file size in bytes. If the event file size is greater than this value, it is split into chunks of this value and that number of chunks are posted. If the event file size is less than this value, the entire event file is posted.
	If the SplitCriteria property value is set to `0`, file splitting is disabled.
	Note: During inbound processing, if file splitting is based on size or delimiter and the FilePassByReference property is enabled, the event files are not split into chunks.
	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 the following method:
	1. Open the Business Object in a text editor.
	2. 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.
	3. Locate a namespace-appended tag called `aggregateInstanceTD` and use the value for the attribute `contentSize`. In this example, the value is 117. This value 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">`

Globalized	Yes
Bidi supported	Yes

Split file content based on size (bytes) or delimiter

This property specifies file splitting either by size or delimiter. The splitting function class name and the split criteria are used to split the file content.

Table 88. Split file content based on size (bytes) or delimiter

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean

Table 88. Split file content based on size (bytes) or delimiter (continued)

Usage	This property specifies whether the adapter splits the contents of the files present in the event directory based on either by size or delimiter. You can specify the following settings to split either by size or delimiter. • Specify the split function class name to `com.ibm.j2ca.utils.filesplit.SplitBySize` and specify the split size (in bytes) criteria to enable the adapter to split the file contents. • Specify the split function class name to `com.ibm.j2ca.utils.filesplit.SplitByDelimiter` and specify the delimiter criteria to enable the adapter to split the file contents.
Globalized	No
Bidi supported	No

Split function class name

This property determines how the event file is to be split.

Table 89. Split function class name details

Required	No

Possible values	`com.ibm.j2ca.utils.filesplit.SplitByDelimiter`
	– Files are split based on a delimiter that separates business objects in the event file
	`com.ibm.j2ca.utils.filesplit.SplitBySize`
	– Files are split based on the size of the event file

Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Usage	The delimiter or file size is set in the SplitCriteria property. **Note:** If the SplittingFunctionClassName property is null, this property is automatically set to `com.ibm.j2ca.utils.filesplit.SplitBySize`.
Globalized	No
Bidi supported	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 90. 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 archive

This property specifies the file extension used to archive successfully processed business objects.

Table 91. Success file extension for archive details

Required	No
Default	`success`
Property type	String
Globalized	Yes
Bidi supported	Yes

Table name to store the file processing status (EP_FileTableName)

This property specifies the table name to store the file processing status. The adapter continues to process the file, from its last stored status during the event recovery.

Table 92. Table name to store the file processing status (EP_FileTableName) details

Required	No
Default	`FILETABLE`
Property type	String
Usage	This property supports WebSphere Adapter for Flat Files to read only the partial contents of the file required by the polling quantity and tracks the last file position reached after a partial read of the file. The file status stored in the table is used during the event recovery. **Note:** During the event recovery, the adapter continues to process the file from its last stored status in the table.
Globalized	Yes
Bidi supported	Yes

Time interval for polling unchanged files (in milliseconds)

This property specifies whether the adapter retrieves only those files that are not changed during the specified time interval.

Table 93. Time interval for polling unchanged file

Required	No
Default	0
Unit of measure	Milliseconds
Property type	Integer
Usage	This property enables the adapter to retrieve only those files that are not modified in the event directory for a specified time interval. When this property is selected, the adapter retrieves the unchanged files during the poll cycles. The adapter also polls the files that are currently being edited but retrieves the file content present during the last save of the file.
Globalized	No
Bidi supported	No

Time out period for HA Active-Active event processing change (in seconds) (EP_Timeout)

Specifies the time interval, in seconds, for processing the events fetched. The unprocessed events at the end of the time interval are reprocessed as new events.

Table 94. Time out period for HA Active-Active event processing change (in seconds) property characteristics

Required	Yes
Default	300
Unit of measure	Seconds
Property type	Integer
Usage	This property is used for specifying the time interval, in seconds, for the adapter to process the events fetched . If for any reason the adapter fails to process all the fetched events at the end of the time interval, the unprocessed events are reprocessed as new events by a different adapter. **Note:** You can use this property if the HA Active-Active configuration is enabled and the guaranteed delivery event is required.
Globalized	No
Bidi supported	No

User name used to connect to event data source

This property specifies the user name used by event persistence processing to obtain the JDBC database connection from the data source.

Table 95. User name used to connect to event data source details

Required	No
Default	None
Property type	String
Globalized	Yes
Bidi supported	Yes

Rule editor to filter files (ruleTable)

This property is used to filter event files based on a set of rules

Table 96. Rule editor to filter files

Required	Optional
Default	No default value
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

Related concepts

"WebSphere Application Server environment variables" on page 31
WebSphere Application Server environment variables can be used in the external service wizard to specify directory values.

"Creating the required local folders" on page 60
Before you create inbound or outbound modules, you must create folders on the local file system for events and output. You can optionally create folders for staging and archiving.

"Known issues in editing the Rule Table" on page 149
When configuring the adapter to filter event files based on a set of rules, some known issues can occur while editing the Rule Table in the Properties view. To correct the problem follow the solutions described here for each of these issues.

"Inbound processing" on page 13
WebSphere Adapter for Flat Files supports inbound event processing. It polls the local file system at specified intervals for events, such as the creation of a file. When it detects an event, it converts the event data into a business object and sends it to the module for processing.

"File splitting" on page 25
The adapter supports an optional file splitting feature to reduce memory loading during the event processing. When this feature is used, the adapter divides large event files into smaller chunks, which are then posted separately to the endpoint.

"File retrieval" on page 23
During inbound processing, you can manage the retrieval of the files by using the `Poll event files for modified content` or the `Time interval for polling unchanged files` property. You can also use the `Include only the newly appended content` property to retrieve only the appended file contents.

Related tasks

"Defining WebSphere Application Server environment variables" on page 62
Use the administrative console of IBM Business Process Manager or WebSphere Enterprise Service Bus to define WebSphere Application Server environment variables.

"Setting deployment and runtime properties" on page 97
After you have decided whether your module is to be used for outbound or inbound communication with the enterprise information system (local file system), you must configure the activation specification properties, which hold the inbound event processing configuration information for the export.

"Setting deployment and runtime properties" on page 97
After you have decided whether your module is to be used for outbound or inbound communication with the enterprise information system (local file system), you must configure the activation specification properties, which hold the inbound event processing configuration information for the export.

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 7.5. They are visible from the administrative console 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 detail tables in the sections that follow, see "Guide to information about properties" on page 168.

Table 97. Resource adapter properties for the WebSphere Adapter for Flat Files

Name		
In the wizard	**In the administrative console**	**Description**
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.
Disguise user data as "XXX" in log and trace files	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
"Enable high availability support (enableHASupport)" on page 213	enableHASupport	Specifies if a single or multiple adapter instances are active at a given time in a clustered environment.
(Not available)	LogFileSize	Deprecated
(Not available)	LogFilename	Deprecated
(Not available)	LogNumberOfFiles	Deprecated
(Not available)	TraceFileSize	Deprecated
(Not available)	TraceFileName	Deprecated
(Not available)	TraceNumberOfFiles	Deprecated

Adapter ID (AdapterID)

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

Table 98. Adapter ID details

Required	Yes
Default	`001`
Property type	String

Table 98. Adapter ID details (continued)

Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `FFRA`, 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 `FFRA001`. If you run multiple instances of the same adapter, ensure that the first nine characters of the adapter ID property are unique for each instance so that you can correlate the 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 the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for Flat Files to `001` and `002`. The component IDs for those instances, `FFRA001` and `FFRA002`, 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. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `FFRAInstance0`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use 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 IBM Integration Designer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	Yes
Bidi supported	No

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

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 99. 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, the value can be set both at the resource adapter level and the managed connection factory level. After you use 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 IBM Integration Designer assembly editor or the administrative console to reset these properties, ensure that you set them consistently, to prevent inconsistent marking of the log and trace entries.
Globalized	No

Table 99. Disguise user data as "XXX" in log and trace files details (continued)

Bidi supported	No

Enable high availability support (enableHASupport)

This property specifies the cluster deployment settings for the adapter.

Note: For HA Active-Active configuration, this property must be set to `false` in the administrative console.

Table 100. Enable high availability support property details

Required	No
Possible values	`True` (Active-Passive) `False` (Active-Active)
Default	`True`
Property type	Boolean
Usage	Use this property to specify if the adapter is to be run in Active-Passive or Active-Active configuration mode. **Active-Passive configuration mode** By default, the adapter is configured to run in an Active-Passive mode that provides high availability support. This configuration allows only one adapter instance to be active to poll a remote event directory for files. **Active-Active configuration mode** When you use the Active-Active mode, the adapter provides both high availability and load balancing support. This configuration allows multiple adapter instances to simultaneously poll a remote event directory for files. It also provides the advantage of faster event processing by ensuring that there is no duplication when processing the different adapter instances. The following limitations are applicable when you configure the adapter in the Active-Active mode: • The Active-Active configuration works only for the `Unordered` delivery type. • The event persistence entries must not be left empty. • The sorting of event files by file name or timestamp is not supported in the HA Active-Active configuration. **Note:** If you change the existing value of this property in the administrative console, the adapter overwrites the value specified in the external service.
Globalized	No
Bidi supported	No

Globalization

WebSphere Adapter for Flat Files 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 data 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 script data transformation, that refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file.

Globalization

Globalized software applications are designed and developed for use within multiple linguistic and cultural environments rather than a single environment. WebSphere Adapters, IBM Integration Designer, IBM Business Process Manager or WebSphere Enterprise Service Bus are written in Java. 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). Therefore, when data is transferred between these 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 script data 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 data, standards are used to display and process it. Bidirectional script data transformation applies only to string type data. IBM Business Process Manager or WebSphere Enterprise Service Bus uses the Windows standard format, but applications or file systems that exchange data with the server might use a different format. The adapter transforms bidirectional script data passed between the two systems so that it is accurately processed and displayed on both sides of a transaction. It transforms the script data by using a set of properties that defines the format of script data, as well as properties that identify content or metadata to which transformation applies.

Bidirectional script data formats

IBM Business Process Manager or WebSphere Enterprise Service Bus uses the bidirectional format of ILYNN (implicit, left-to-right, on, off, nominal). This bidirectional format is used by Windows. If an enterprise information system uses a different format, the adapter converts the format before introducing the data to IBM Business Process Manager or WebSphere Enterprise Service Bus.

The bidirectional format consists of five attributes. When you set bidirectional properties, you assign values for each of these attributes. The attributes and settings are listed in the following table.

Table 101. Bidirectional format attributes

Letter position	Purpose	Values	Description	Default setting
1	Order schema	I	Implicit (Logical)	I
		V	Visual	

Table 101. Bidirectional format attributes (continued)

Letter position	Purpose	Values	Description	Default setting
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	
5	Numeric Shaping	H	National (Hindi)	N
		C	Contextual shaping	
		N	Numbers are not shaped (Nominal)	

Bidirectional properties that identify data for transformation

To identify business data for transformation, set the BiDiContextEIS property. Do this setting by specifying values for each of the five bidirectional format attributes (listed in the preceding table) for the property. The BiDiContextEIS property can be set for the managed connection factory and for the activation specification.

To identify event persistence data subject for transformation, set the EP_BiDiFormat property. The value for the EP_BiDiFormat property is set with the value you specified in the BiDiContextEIS property. The EP_BiDiFormat property can be set for the activation specification.

To identify application-specific data for transformation, annotate the BiDiContextEIS property and the BiDiMetadata property within a business object. Do this setting by using the business object editor within IBM Integration Designer to add the properties as application-specific elements of a business object.

Related reference

"Activation specification properties" on page 191
Activation specification properties hold the inbound event processing configuration information for an export. You set activation specification properties through either the external service wizard or the administrative console.

"Managed connection factory properties" on page 172
Managed connection factory properties specify information the adapter needs at run time for outbound communication with the local file system.

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 Flat Files 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 FlatFile.xsd file.
- For the user-defined business object, change the custom 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 Flat Files business objects. The FlatFileBaseDataBinding 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 (FlatFile) 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">
<xsd:annotation>
            <xsd: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>
            </xsd:appinfo>
        </xsd: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. Once these properties are set, bidirectional script data is correctly processed and displayed in IBM Integration Designer and IBM Business Process Manager or WebSphere Enterprise Service Bus.

Managed connection factory properties

The following managed connection factory properties control bidirectional script data transformation:

- FileSequenceLog
- OutputDirectory
- OutputFilename
- StagingDirectory

Activation specification properties

The following activation specification properties control bidirectional script data transformation:

- ArchiveDirectory
- EventDirectory
- EventFileMask
- FailedArchiveExtension
- OriginalArchiveExtension
- SplitCriteria
- SuccessArchiveExtension

Deployment Descriptor configuration properties

The following deployment descriptor configuration properties control bidirectional script data transformation:

- EPDatabasePassword
- EPDatabaseSchemaName
- EPDatabaseUsername
- EPDataSourceJNDIName
- EPEventTableName

Business object wrapper properties

The following business object wrapper properties control bidirectional script data transformation:

- DirectoryPath
- FileName
- IncludeEndBODelimiter
- StagingDirectory
- ArchiveDirectoryForDeleteOnRetrieve
- ChunkFileName

Adapter messages

View the messages issued by WebSphere Adapter for Flat Files at the following location.

Link to messages: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.ref.doc/topics/welc_ref_msg_wbpm.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 CWYFF are issued by WebSphere Adapter for Flat Files
- 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 WebSphere Adapter for Flat Files.

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

- IBM Business Process Manager, version 7.5, information center, which includes IBM Business Process Manager, IBM WebSphere Enterprise Service Bus, and IBM Integration Designer information: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/index.jsp.
- IBM Business Process Manager, version 7.0, information center, which includes IBM Business Process Manager, IBM WebSphere Enterprise Service Bus, and IBM Integration Designer information: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp.
- WebSphere Adapters, version 6.2.x, information center: http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r2mx/index.jsp.

- IBM WebSphere Adapters, version 7.5 installation on WebSphere Application Server, version 8.0 information: http://www-01.ibm.com/support/docview.wss?rs=695&uid=swg27011040.

developerWorks® resources

- WebSphere Adapter Toolkit
- WebSphere business integration zone

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 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

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X

Printed in USA