WebSphere® Adapters
Version 7 Release 0 Feature Pack 1

WebSphere Adapter for FTP User Guide Version 7 Release 0 Feature Pack 1

WebSphere® Adapters
Version 7 Release 0 Feature Pack 1

WebSphere Adapter for FTP User Guide Version 7 Release 0 Feature Pack 1

Note

Before using this information and the product it supports, read the information in “Notices” on page 201.

June 2010

This edition applies to version 7, release 0, modification 1 of IBM® WebSphere Adapter for FTP and to all subsequent releases and modifications until otherwise indicated in new editions.

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

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

Contents

WebSphere Adapter for FTP documentation

WebSphere® Adapter for FTP provides the ability for applications in WebSphere Process Server, and WebSphere Enterprise Service Bus, to access data that is only accessible through an FTP server without the need for special coding. The adapter works with the FTP server by sending requests to it and by monitoring and responding to events that occur in it.

What is new in this release

This version includes several new features that enhance the business flexibility, user experience, and performance of the adapter.

Complete information about other supported features is available at the WebSphere Adapter for FTP information center, http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/stbp_ftp_welcome.html, which is periodically updated with the latest information.

WebSphere Adapter for FTP supports the following new features:

- Renaming files after you create them on the FTP server.
- Processing the event files downloaded using FTP scripts.
- Performing inbound and outbound operations in user's home directory on the FTP server.
- Resuming transfer of files after reconnection to FTP or FTPS server.
- Authentication using connection specification properties.
- Polling unchanged files in the event directory for a specific time interval.
- Using relative paths in remote directories.

Note: In WebSphere Integration Developer, ensure that you have only one version of the adapter imported into your workspace. You can either have the adapter Fix Pack version 7.0.0.1 or Feature Pack version 7.0.1.0.

In the runtime environment, the application (EAR) should contain only one version of the embedded RAR file, either the adapter Fix Pack version 7.0.0.1 or Feature Pack version 7.0.1.0. The node level deployed adapter should also have only any one version of the adapter.

Support for renaming files

A new property, 'Temporary file name' is added to the interaction specification properties. This property specifies the temporary file name for the create operation.

The file with the specified name is created in the specified directory of the FTP server. If a temporary file name is specified, a file is created with a temporary file name on the FTP server. On successful creation of the file at a remote location, the file gets renamed to the value specified in the 'Default target file name' property.

Supported operations

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

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

Table 1. Supported outbound operations

Operation	**Result**
Create	The file with the specified name is created in the given directory of the FTP server. If a temporary file name is specified, a file is created with a temporary file name on the FTP server. After the file is created successfully at a remote location, it is renamed to the target file name. The content of the file can either be sent as part of the request or it can be retrieved from the local file system. When the file content is received as part of the request, the adapter provides the option to archive the file on the adapter workstation before creating it. When the file content is received as part of the request, the adapter provides the option to archive the file on the adapter workstation before creating it. The file can be created in a staging directory and then sent to the actual directory. If a staging directory is not specified, the file is directly created in actual directory. After the file is created, the file name is sent back to the calling component to indicate that the file was created successfully. If the file to be created exists, a DuplicateRecord exception is sent, and the file is not created. The existing file is not overwritten. The adapter provides a feature to generate unique file names. See “Generating unique file names” on page 5. The adapter provides a feature to create a file sequence for the output files created. See “Generating a file sequence during Create operations” on page 4.
Append	The file with the specified name in the specified directory of the FTP server is appended with the content sent in the request. If the file to be appended exists, the content is appended, and the file name is sent back to the calling component indicating a successful response. If the staging directory is specified, the file to be appended is copied from the specified output directory to the staging directory, and the content is appended to that file in the staging directory. The appended file is then moved back to the original directory. If the file to be appended does not exist and the CreateIfFileNotExist property is set to `true`, the adapter creates a new file. If the file to be appended does not exist, a RecordNotFound exception is sent to the calling component.
Delete	The file in the specified directory is deleted on the FTP server and the adapter returns `true` to the calling component to indicate that the file was successfully deleted. If the file to be deleted does not exist, a RecordNotFound exception is sent to the calling component.

Table 1. Supported outbound operations (continued)

Operation	**Result**
Retrieve	The content of the file or files in the specified request is returned. The file content is split based on the SplittingFunctionClassName and SplitCriteria properties. The file content is transformed into a business object based on the configured data handler. After the content of the file is retrieved it is sent as the response. The file content can either be sent back to the calling component or saved to the local file system. If the file to be retrieved does not exist, a RecordNotFound exception is sent to the calling component. The adapter provides an option to delete the file from the FTP server directory after it is retrieved through the DeleteOnRetrieve property. The adapter supports an option to archive the file on the FTP server before it is deleted through the ArchiveDirectoryForDeleteOnRetrieve property. While configuring the Retrieve operation for data transformation, create custom retrieve wrappers like CustomerRetrieveWrapper or CustomerRetrieveWrapperBG, or OrderRetrieveWrapper or OrderRetrieveWrapperBG, and use the wrapper for the output type in the operation window. For a Retrieve operation without data transformation, the default wrapper RetrieveResponseWrapper is used. **Note:** The backward compatibility may use RetrieveResponseWrapper for retrieving XML data with data transformation.
Overwrite	This operation overwrites the file in the directory with the content specified in the request. After, the content is overwritten, the file name is sent back to the calling component indicating a successful response. The file to be overwritten is copied from the specified directory to the staging directory, if specified, and the content is overwritten for that file in the staging directory. The file is then moved back to the specified directory. If a staging directory is not specified, the content is overwritten on the file in the specified directory. If the file to be overwritten does not exist, and the CreateIfFileNotExist property is set to `true`, the adapter creates a new file. If the file to be overwritten does not exist, a RecordNotFound exception is sent to the calling component.
Exists	If the file name in the request exists in the specified directory or any of the sub folders, the adapter returns `true` and the full path of the file to the calling component. If a file with the same name exists in more than one directory, the adapter returns `true` and the full path of the first file found to the calling component. If the file name does not exist, or the directory does not exist, the adapter returns `false` to the calling component.
List	All the file names and directories that are specified in the request are returned to the calling component. If only the directory is specified, all the file names in the directory are retrieved and sent as a response to the calling component. If the specified directory does not exist, a RecordNotFound exception is sent to the calling component.

Table 1. Supported outbound operations (continued)

Operation	Result
ServerToServer FileTransfer	The specified file is transferred from one FTP server directory to another FTP server directory. After the file has been transferred successfully, `true` is returned to the calling component. Both the FTP servers must support the ServerToServerFileTransfer operation and a connection must be established between the FTP servers and the workstation where the adapter is running. If the request does not contain all necessary information about the two servers, the adapter sends an FTPFileServerToServerFileTransfer exception to the calling component. **Note:** The ServerToServerFileTransfer operation does not support FTPS (FTP over SSL and FTP over TLS) or SFTP protocol.
ExecuteFTPScript	The commands contained in a FTP script file are run in the adapter workstation. The operation runs only the commands that are supported by the FTP server. If the operation fails, the adapter sends an FTPFileExecuteFTPScript exception to the calling component. The script file should not contain connection-related commands such as `open` because the adapter uses an established connection to run the commands. The directory should be specified in DirectoryPath and file name in FileName property. If the commands in the script file has to be run in a particular directory on the FTP server, then the script file must first contain the command to change to that directory. A list of commands runs and their reply strings are returned to the calling component. The adapter also supports parameter substitution in the FTP script file (replacing parameters %1, %2 with actual values). The values are sent as part of the request. **Note:** The script file should contain commands that are supported by the selected protocol.

Generating a file sequence during Create operations

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

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

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

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

When a value is specified for the FileSequenceLog property, the adapter generates file sequence numbers, and appends to the file name of the files that it creates. The sequence number accepts the following format:

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

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

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

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

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

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

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

1. The adapter reads the sequence file and checks for an entry of the form path = sequenceNumber.
2. If such an entry exists in the file, the sequence file contains the data in the form supported by Adapter for FTP version 6.1
3. The adapter gets the highest sequence number available from all the entries.
4. This number is used to create a new file.
5. The adapter increments the number and overwrites the entire file with the new number.

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

Generating unique file names

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

- The Create operation supports the generation of unique file names when the GenerateUniqueFile property is set to true. When the GenerateUniqueFile property is enabled or the FileSequenceLog property is set and if a temporary file name is provided, the file will be directly created with the target file name.

Note: For Append and Overwrite operations, the GenerateUniqueFile property is deprecated from v6.2 onwards. Even if the value is set for this property, the adapter considers the value as 'False'.

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

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

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

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

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

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

Related reference

"Wrapper and interaction specification properties" on page 10
Wrapper properties are attributes of the wrapper business object that enable an application programmer to control an operation for the business objects in a wrapper. Interaction specification properties control the interaction for an operation for the entire adapter.

Outbound data transformation

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

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

Data bindings

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

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

- DirectoryPath
- Filename
- TemporaryFilename
- DataConnectionMode
- FileTransferType
- DataProtectionLevel
- SecondServerDirectory
- SecondServerUsername
- SecondServerPassword
- IncludeEndBODelimiter
- ResumeFailedTransfer
- FileInLocalDirectory
- LocalDirectoryPath
- LocalArchivingEnabledForCreate
- LocalArchiveDirForCreate
- StagingDirectory
- GenerateUniqueFile
- SplittingFunctionClassName
- SplitCriteria
- DeleteOnRetrieve
- ArchiveDirectoryForRetrieve
- FileContentEncoding

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

Data handlers

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

- Delimited
- Fixed width
- XML

Setting interaction specification properties and generating the service

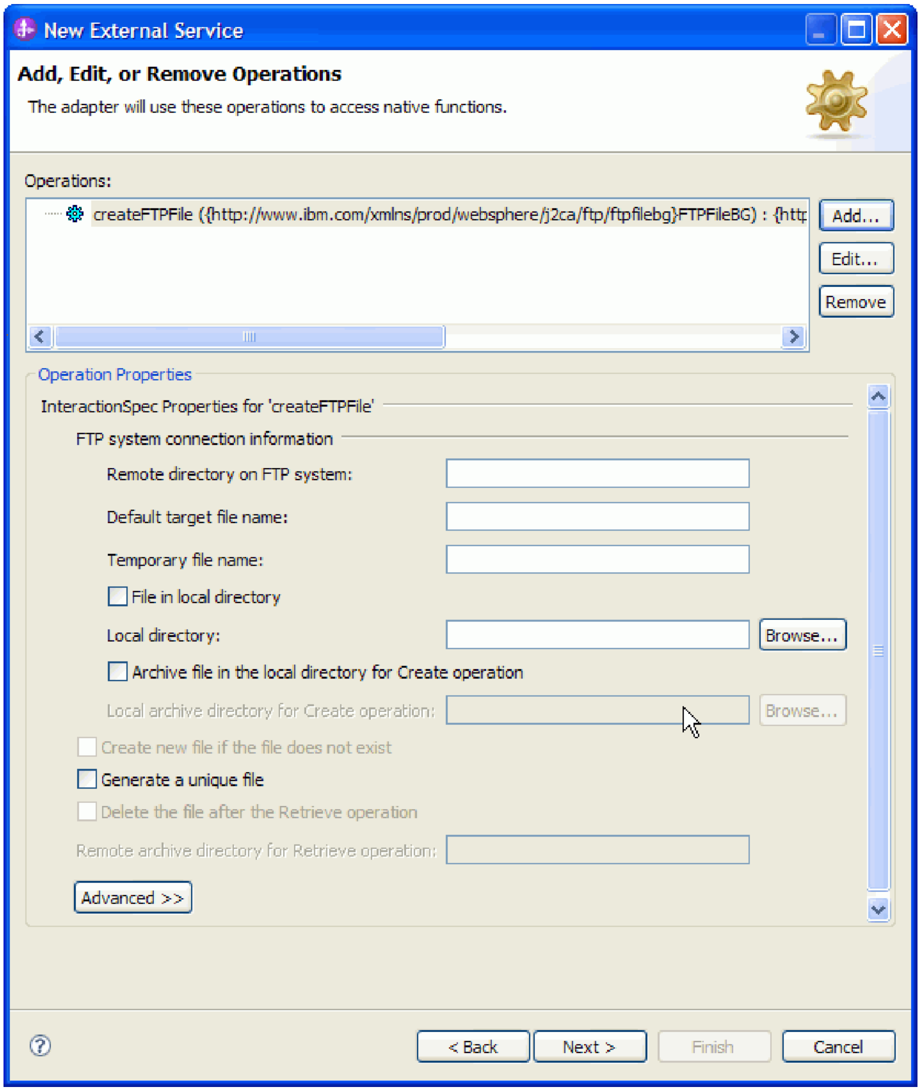
Interaction specification properties are optional. If you choose to set them, the values you specify is displayed as defaults in all parent FTP business objects generated by the external service wizard. Interaction specification properties control the interaction for an operation. While creating artifacts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

About this task

To set interaction specification properties and generate artifacts, follow this procedure. For more information, refer the topic wrapper and interaction specification properties.

Procedure

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



New External Service

Add, Edit, or Remove Operations

The adapter will use these operations to access native functions.

Operations:

createFTPFile ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfilebg}FTPFileBG) : {http

Operation Properties

InteractionSpec Properties for 'createFTPFile'

FTP system connection information

Remote directory on FTP system:

Default target file name:

Temporary file name:

□ File in local directory

Local directory:

□ Archive file in the local directory for Create operation

Local archive directory for Create operation:

□ Create new file if the file does not exist

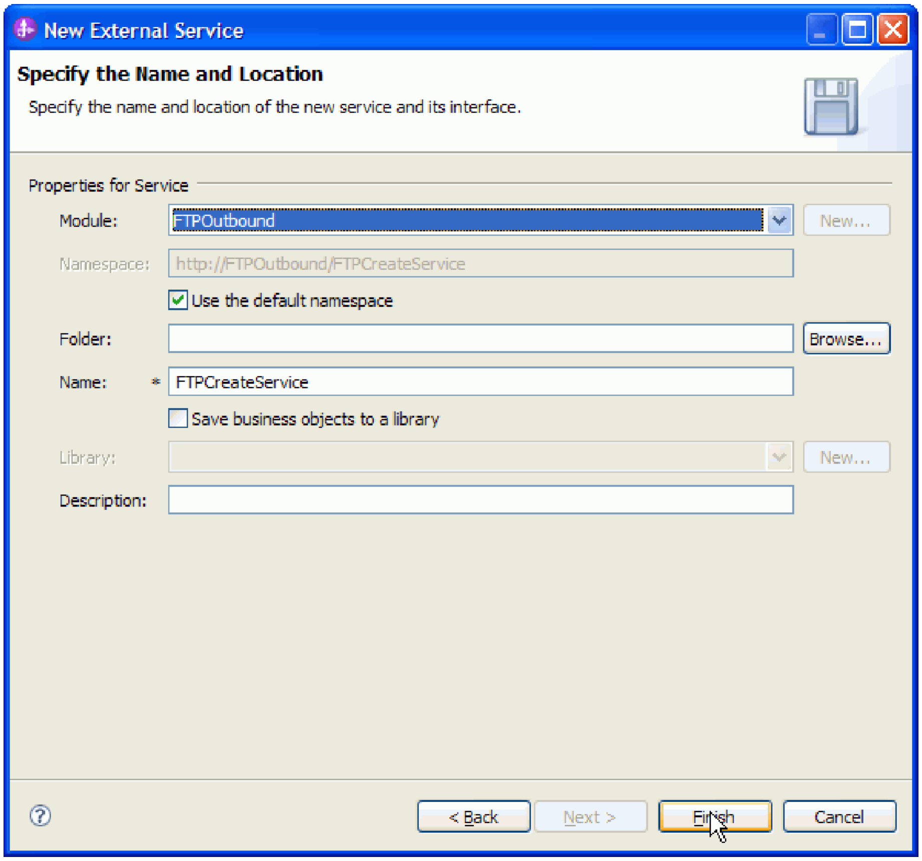
□ Generate a unique file

□ Delete the file after the Retrieve operation

Remote archive directory for Retrieve operation:

Figure 1. Interaction specification properties

2. In the Generate Service window, specify a name for the interface. This is the name that is displayed in the WebSphere Integration Developer assembly diagram.



New External Service

Specify the Name and Location

Specify the name and location of the new service and its interface.

Properties for Service

Module: FTPOutbound

Namespace: http://FTPOutbound/FTPCreateService

☑ Use the default namespace

Folder:

Name: * FTPCreateService

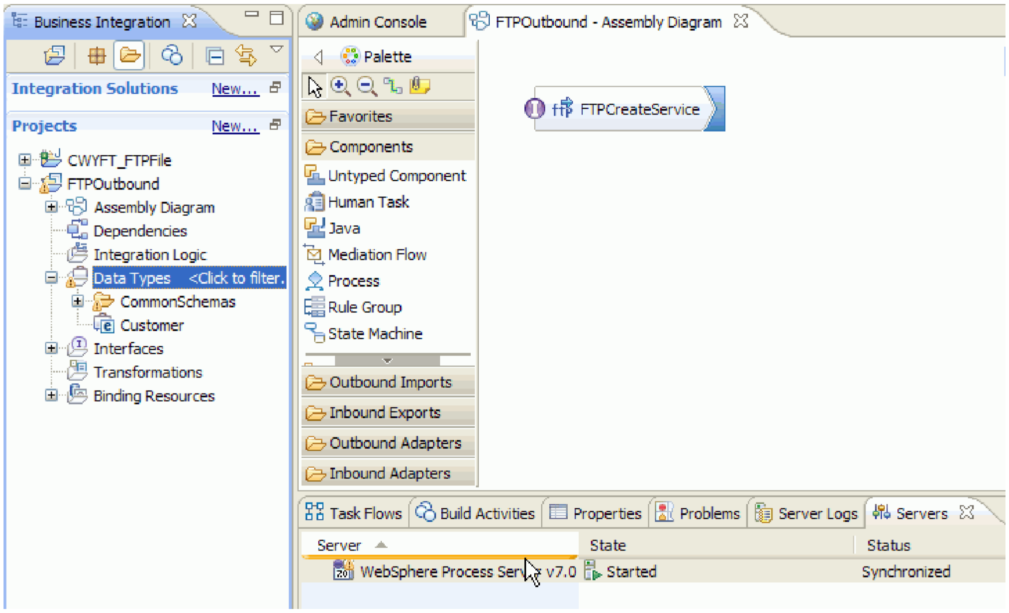
□ Save business objects to a library

Library:

Description:

Figure 2. Specify the Name and Location window

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



Business Integration

Integration Solutions New...

Projects New...

- CWYFT_FTPFile
- FTPOutbound
  - Assembly Diagram
  - Dependencies
  - Integration Logic
  - Data Types <Click to filter.
    - CommonSchemas
    - Customer
  - Interfaces
  - Transformations
  - Binding Resources

Admin Console

FTPOutbound - Assembly Diagram

Palette

Server	State	Status
WebSphere Process Server v7.0	Started	Synchronized

Figure 3. Interface in WebSphere Integration Developer

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

Results

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

What to do next

Deploy the module to the server.

Wrapper and interaction specification properties

Wrapper properties are attributes of the wrapper business object that enable an application programmer to control an operation for the business objects in a wrapper. Interaction specification properties control the interaction for an operation for the entire adapter.

The external service wizard sets the interaction specification properties when you configure the adapter. You can change some, but not all, of these properties. However, you can change some properties for outbound operations. Use the assembly editor to change these properties, which reside in the method binding of the import. You set the wrapper properties using the WebSphere Integration Developer test client or programmatically at run time.

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

Table 2. Interaction specification properties

Property name		Description
In the wizard	In the wrapper business object	
Remote archive directory for retrieve operation	ArchiveDirectoryForRetrieve	The adapter optionally archives the file to this folder before it is deleted during a Retrieve operation.
Create new file if the file does not exist	CreateFileIfNotExists	If the file does not exist on the FTP server, the adapter creates the file when this property is set to `true` during Append and Overwrite operations.
FTP server connection mode	DataConnectionMode	Data connection mode used by the FTP server during file transfers.
Delete the file after retrieve operation	DeleteOnRetrieve	The adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.
Remote directory on FTP system	DirectoryPath	Absolute path of the directory on the FTP server where the outbound operation must be performed.
“Data channel protection level (dataProtectionLevel)” on page 12	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
File content encoding	FileContentEncoding	Encoding used while writing to the file.

Table 2. Interaction specification properties (continued)

File in local directory	FileInLocalDirectory	If set to `true` during a create operation, the file content is picked from the local directory path of the adapter workstation.
Default target file name	Filename	Name of the file in the directory provided by the DirectoryPath property.
File transfer type	FileTransferType	File transfer type used during outbound operations.
Generate a unique file	GenerateUniqueFile	The adapter creates a unique file name when this property is set to `true`.
Host name property	SecondServerHostName	Host name of the second FTP server.
Delimiter between business objects in the file property	IncludeEndBODelimiter	File content is appended with this value.
Local archive directory for create operation	LocalArchiveDirForCreate	When `LocalArchivingEnabledForCreate` is set to `true` during a create operation, the file is saved to the local workstation in this directory.
Archive file in the local directory for create operation	LocalArchivingEnabledForCreate	When set to `true`, the file is saved to the local workstation during a create operation.
Local directory	LocalDirectoryPath	The file is picked from this directory.
(Not available)	ResumeFailedTransfer	When this property is set to `true` during a create operation, the adapter resumes the transfer of files from the point at which the transfer of file was interrupted due to connection error.
Port number	SecondServerPortNumber	Port number of the second FTP server.
Protocol	SecondServerProtocol	Specifies the protocol used to connect to the second server.
Script File Parameters	ScriptFileParameters	The parameters required by the FTP script file.
Directory	SecondServerDirectory	Directory path of the second FTP server during a ServerToServerFileTransfer operation.
Password	SecondServerPassword	Password of the second FTP server during a ServerToServerFileTransfer operation.
User name	SecondServerUsername	User name of the second FTP server during a ServerToServerFileTransfer operation.
Specify criteria to split file content	SplitCriteria	The delimiter that separates the business objects in the event file.
Split function class name	SplittingFunctionClassName	The fully qualified class name of the class file to be used to enable file splitting.
Staging directory	StagingDirectory	The file is first created into this directory.
Temporary file name	TemporaryFilename	Specifies the temporary file name for the create operation.

Archive file in the local directory for create operation property (LocalArchivingEnabledForCreate)

During outbound create operations, when the file content is coming as part of the business object from a J2EE application and this property is set to `true`, the file is saved to the local workstation in the LocalArchiveDirForCreate directory before performing the outbound operation.

Table 3. Archive file in the local directory for create operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Create new file if the file does not exist property (CreateFileIfNotExists)

During outbound Append and Overwrite operations, if the file does not exist on the FTP server, the adapter creates the file when this property is set to `true`. If this property is `false` and file does not exist, the adapter sends an error.

Table 4. Create new file if the file does not exist property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the “PBSZ 0” command before issuing the PROT command.

Table 5. Data channel protection level property characteristics

Required	No

Possible values	`Private - Data is transferred in encrypted form`
	`Clear - Data is transferred as clear text`

Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Delete the file after retrieve operation (DeleteOnRetrieve)

During an outbound Retrieve operation, the adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.

Table 6. Delete the file after retrieve operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Default target file name property (Filename)

Name of the file to be used during outbound operations.

Table 7. Default target file name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Delimiter between business objects in the file property (IncludeEndBODelimiter)

File content is appended with this value. Used during the outbound create, append, and overwrite operations.

Table 8. Include business object delimiter in the file content property characteristics

Required	No
Default	For the create and overwrite operations, no default value is set. For the append operation, the default value is `<EndBO>`. For the append operation, the following rules apply: • If the delimiter is set to `null` in the business object wrapper, no delimiter is used to separate the business objects. • If the IncludeEndBODelimiter property is not set in the business object wrapper, and the value in the interaction specification is also `null`, the default is `<EndBO>`. • If a specific delimiter value is specified in the business object wrapper, the specified value will be appended. • If both the business object wrapper and the interaction specification have specified values, the business object wrapper value takes precedence.
Property type	String
Globalized	Yes

Directory property (SecondServerDirectory)

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

Table 9. Directory property characteristics

Required	No
Default	None

Table 9. Directory property characteristics (continued)

Property type	String
Usage	For interaction specification properties, the directory located on the FTP server and used in outbound operation represents the absolute path of the FTP directory. For example: `/home/usr/output`. It does not contain any host name or URL information. For wrapper business object properties, the URL of the second server to which the ServerToServerFileTransfer outbound operation is performed. For example: The syntax for specifying the FTP URL is: `ftp://[UserId:password@]FTPserver[:port]/DirectoryForSecondServer`.
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used while writing to the file. If this property is not specified, the adapter tries to read without using any specific encoding. You can specify any Java™ supported encoding set.

Table 10. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File in local directory property (FileInLocalDirectory)

During outbound create operations, if this property is set to `true`, the file content is not available in the business object. The file is retrieved from the local directory on the adapter workstation. During outbound retrieve operations, if this property is set to `true`, the file content is not sent to the J2EE application as part of the business object. The file is saved to the local directory of the adapter workstation.

Table 11. File in local directory property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during outbound operations. Takes either ASCII or binary.

Table 12. File transfer type property characteristics

Required	No
Default	`binary`
Property type	String
Globalized	No

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Takes either `active` or `passive`. This value is used only when a file transfer is taking place. This property is not used when performing a server to server file transfer outbound operation.

Table 13. FTP server connection mode property characteristics

Required	No
Default	`active`
Property type	String
Possible values	`active` or `passive`
Globalized	No

Generate a unique file (GenerateUniqueFile)

During outbound Create operation, the adapter creates a unique file name when this property is `true`. The adapter ignores any value that is set for the Filename property when this property is set to `true`.

Note: The adapter does not support both GenerateUniqueFile and StagingDirectory options at the same time.

Table 14. Generate unique file property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No
Restrictions	The FTP server must support RFC1123 to use this feature.

Host name property (SecondServerHostName)

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

Table 15. Host name property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Local archive directory for create operation property (LocalArchiveDirForCreate)

During outbound create operations, when the file content is coming as part of the business object and LocalArchivingEnabledForCreate is set to `true`, the file is saved to the local workstation in this directory.

Table 16. Local archive directory for create property characteristics

Required	No

Table 16. Local archive directory for create property characteristics (continued)

Default	None
Property type	String
Usage	The LocalArchiveDirForCreate directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Local directory property (LocalDirectoryPath)

During outbound create operations, when FileInLocalDirectory property is set to `true`, the file content is not available in the business object. Instead the file is picked from this directory. During outbound retrieve operations, when FileInLocalDirectory property is set to `true`, the file content is not sent as part of business object. The file is saved to this directory.

Table 17. Local directory property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalDirectoryPath directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Port number property (SecondServerPortNumber)

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

Table 18. Port number property characteristics

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

Protocol property (SecondServerProtocol)

Protocol that is used to establish a connection to the second server. The FTP protocol is used in establishing the connection.

Table 19. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Password property (SecondServerPassword)

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

Table 20. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Remote archive directory for retrieve operation property (ArchiveDirectory ForRetrieve)

During an outbound Retrieve operation, the adapter optionally archives the file to this folder before it is deleted. The archive directory must exist.

Table 21. Remote archive directory for retrieve operation property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Remote directory on FTP system property (DirectoryPath)

Absolute path of the directory on the FTP server where the outbound operation must be performed for all operations except ExecuteFTPScript, or the directory path on the local adapter workstation for the ExecuteFTPScript operation only. The directory must exist.

Note: If the value <HOME_DIR> is specified as the DirectoryPath, the outbound operations will be performed in the user's home directory.

Table 22. Remote directory on FTP system property characteristics

Required	No
Default	None
Property type	String
Usage	The DirectoryPath directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

ResumeFailedTransfer

This property supports resuming the transfer of files, which were interrupted due to an error in connection to the FTP server.

Note: This property is applicable only to outbound processing.

Table 23. ResumeFailedTransfer property characteristics

Required	No
Default	`false`
Property type	Boolean
Usage	During a create operation, when this property is set to `true`, the adapter resumes the transfer of files from the point at which the transfer of file was interrupted due to an error in connection.
Globalized	No

Script File Parameters property (ScriptFileParameters)

During an outbound ExecuteFTPScript operation, the parameters required by the FTP script file are set in this property. During run time, the adapter replaces the parameters with these values.

Table 24. Script File Parameters property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Specify criteria to split file content property (SplitCriteria)

This property accepts different values based on the value of the SplittingFunctionClassName property.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
  - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
  - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=`0`, chunking is disabled.

Table 25. Specify criteria to split file content property characteristics

Required	No
Default	`0`
Property type	String
Globalized	Yes

Split function class name property (SplittingFunction ClassName)

Takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The com.ibm.j2ca.utils.filesplit.SplitByDelimiter class that splits the event file based on delimiter.

- The com.ibm.j2ca.utils.filesplit.SplitBySize class that splits the event file based on the event file size.

The delimiter or file size is provided in the SplitCriteria property.

Table 26. Split function class name property characteristics

Required	No
Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Globalized	No

Staging directory property (StagingDirectory)

During outbound create operations, the file will be created in this directory first. When the file creation is complete, the file is copied to the directory specified in the DirectoryPath property. This staging directory is also used for Append and Overwrite operations where the specified file is copied to the StagingDirectory, if present. The appended or overwritten content is then moved back to the original specified directory. If StagingDirectory is not specified, the operation is run in the actual required directory.

Note: The adapter does not support both StagingDirectory and GenerateUniqueFile options at the same time.

Table 27. Staging directory property characteristics

Required	No
Default	None
Property type	String
Usage	The StagingDirectory directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Temporary file name property (TemporaryFilename)

This property specifies the temporary file name for the create operation. After successful creation of the file, the file gets renamed to the value specified in the 'Default target file name' property.

Table 28. Temporary file name property characteristics

Required	No
Possible values	All valid file names
Default	None
Property type	String
Usage	This property is used in the create operation. If the temporary file name is specified, the file is created with the temporary file name. After the file is successfully created, the file is renamed to the value that is specified in the 'Default target file name' property.
Example	xyz.tmp
Globalized	No

User name property (SecondServerUsername)

User name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 29. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Related concepts

"Supported operations" on page 2
An operation is an action that the adapter can perform on remote file systems accessible through an FTP server during outbound processing. The name of the operation typically indicates the type of action that the adapter takes, such as *Create* or *Append*.

Processing the event files downloaded using FTP script

WebSphere Adapter for FTP can be used to process the files downloaded using the FTP scripts, in addition to processing the files downloaded from the event directory during polling.

You can specify the scripts to be run before or after polling the event directory using the properties, 'Run FTP script file before downloading files' and 'Run FTP script file after downloading files'.

Inbound processing

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

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

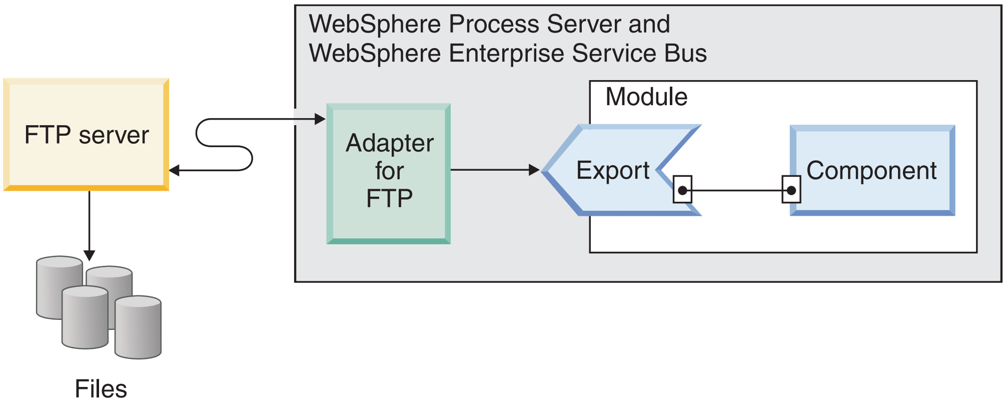

WebSphere Process Server and
WebSphere Enterprise Service Bus
Module
FTP server
Adapter for FTP
Export
Component
Files

Figure 4. Inbound processing flow

The adapter polls files from the event directory of the FTP server at regular intervals based on the FTPPollFrequency property. When a file arrives in the event directory, the adapter reads the entire file and downloads the file to a local event directory on the adapter workstation. The adapter downloads the files from the FTP server sequentially, that is one file at a time, and cannot download all the files simultaneously. After the file is downloaded, the adapter either archives the file in the FTP server in an archive directory given by the FTPArchiveDirectory property or deletes it based on your configuration. The event directory, archive directory, poll frequency, and poll quantity (the number of files to poll in a single poll cycle) are all configurable properties.

Note: If the Remote directory is set to <HOME_DIR>, the adapter polls for event files in `user's` home directory.

Note: The value of an event directory property accepts both the absolute and relative paths of the directory. If the value does not begin with a forward slash (/), the adapter considers the path to be relative to the home directory of the user.

For example, if the value in the remote directory property is set to “ftpuser/event”, the adapter considers this to be the path relative to the home directory. If the home directory is set to “/usr/ftp", then the adapter will poll the directory “/usr/ftp/ftpuser/event” for event files.

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

Inbound event processing consists of the following steps:

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

Processing of files using FTP scripts

In addition to processing the files downloaded from the event directory during polling, WebSphere Adapter for FTP can also be used to process the files downloaded using the FTP scripts.

You can specify the scripts to be run before or after polling the event directory using the properties, “Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)" on page 67 and "Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)" on page 66. The script files can contain FTP commands, such as mget and get, to download the files from the remote directories on the FTP server to the local event directory of the machine where the adapter is installed. The WebSphere Adapter for FTP processes the files that are downloaded to the local event directory configured in the activation specification properties and delivers the processed business objects to the consuming service.

Following is an example of a script:

```
lcd C:\FTPAdapter\localevent
cd /ftpDir1
mget *.txt
cd /ftpDir2
get abc.xml
```

Where, C:\FTPAdapter\localevent is the local event directory of WebSphere Adapter for FTP, and ftpDir1 and ftpDir2 are directories that exist on the FTP server. The adapter executes the script and downloads the files to the local event directory. The adapter then processes the files and delivers it to the consuming service.

Note:

1. You must place the files that have been downloaded using the script to the configured local event directory for the adapter to process it. Use the FTP command lcd to change the local working directory to the localEventDirectory before you download any files using the script.
2. The files downloaded to the local event directory using the commands, mget or get will be deleted from the FTP server by the FTP adapter after you download the files. This is to ensure that the files are not downloaded again during the next poll cycle.
3. Use the script file to download the files only from remote directories and not from the event directory of the FTP adapter.

Supported inbound operation

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

Event file locking

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

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

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

Rule-Based filtering of events

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

- FileName
- File Size
- 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.

Though using the rule is optional and specifying an event file mask is mandatory, the rule takes a higher precedence over the event file mask, when both a rule and an event file mask are specified. Event file mask is effective only when there is no rule specified. By default, an event file mask has "*.*" as the 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 30. 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
LastModified	Greater than, Less than, Greater than or equal to, Less than or equal to, Equal to, Not equal to. **Note:** Select 'Equal to' operator when you choose the days of 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

Inbound and outbound operations in the user's home directory

The home directory can be configured to perform the inbound and outbound operations. The properties, Output Directory, and Event Directory contains a default value, <Home_Dir>. When this value is specified, the adapter performs the operation in the user's home directory. You can also provide your own directory path which will be the absolute path.

Inbound processing

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

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

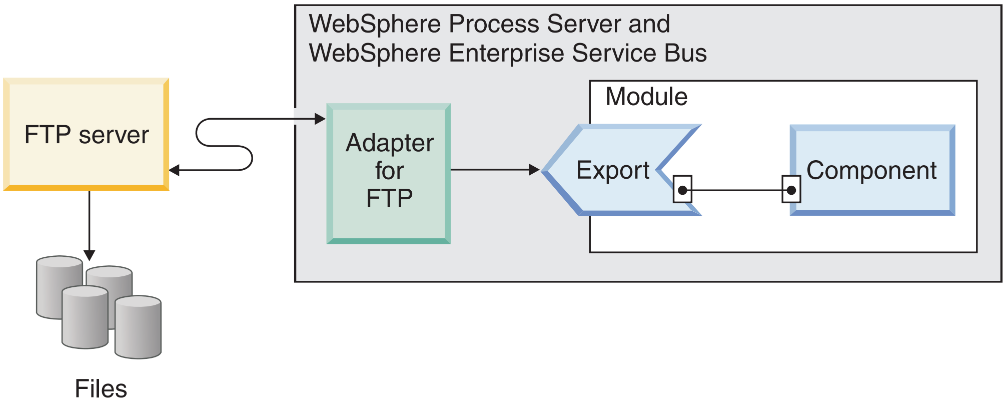

WebSphere Process Server and
WebSphere Enterprise Service Bus
Module
FTP server
Adapter for FTP
Export
Component
Files

Figure 5. Inbound processing flow

The adapter polls files from the event directory of the FTP server at regular intervals based on the FTPPollFrequency property. When a file arrives in the event directory, the adapter reads the entire file and downloads the file to a local event directory on the adapter workstation. The adapter downloads the files from the FTP server sequentially, that is one file at a time, and cannot download all the files simultaneously. After the file is downloaded, the adapter either archives the file in the FTP server in an archive directory given by the FTPArchiveDirectory property or deletes it based on your configuration. The event directory, archive directory, poll frequency, and poll quantity (the number of files to poll in a single poll cycle) are all configurable properties.

Note: If the Remote directory is set to <HOME_DIR>, the adapter polls for event files in `user's` home directory.

Note: The value of an event directory property accepts both the absolute and relative paths of the directory. If the value does not begin with a forward slash (/), the adapter considers the path to be relative to the home directory of the user.

For example, if the value in the remote directory property is set to "ftpuser/event", the adapter considers this to be the path relative to the home directory. If the home directory is set to "/usr/ftp", then the adapter will poll the directory "/usr/ftp/ftpuser/event" for event files.

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

Inbound event processing consists of the following steps:

1. FTP server generates events in the form of files.
2. The Adapter for FTP polls the event directory.
3. The files are downloaded to the adapter.

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

Processing of files using FTP scripts

In addition to processing the files downloaded from the event directory during polling, WebSphere Adapter for FTP can also be used to process the files downloaded using the FTP scripts.

You can specify the scripts to be run before or after polling the event directory using the properties, "Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)" on page 67 and "Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)" on page 66. The script files can contain FTP commands, such as `mget` and `get`, to download the files from the remote directories on the FTP server to the local event directory of the machine where the adapter is installed. The WebSphere Adapter for FTP processes the files that are downloaded to the local event directory configured in the activation specification properties and delivers the processed business objects to the consuming service.

Following is an example of a script:

```
lcd C:\FTPAdapter\localevent
cd /ftpDir1
mget *.txt
cd /ftpDir2
get abc.xml
```

Where, `C:\FTPAdapter\localevent` is the local event directory of WebSphere Adapter for FTP, and ftpDir1 and ftpDir2 are directories that exist on the FTP server. The adapter executes the script and downloads the files to the local event directory. The adapter then processes the files and delivers it to the consuming service.

Note:

1. You must place the files that have been downloaded using the script to the configured local event directory for the adapter to process it. Use the FTP command `lcd` to change the local working directory to the localEventDirectory before you download any files using the script.
2. The files downloaded to the local event directory using the commands, `mget` or `get` will be deleted from the FTP server by the FTP adapter after you download the files. This is to ensure that the files are not downloaded again during the next poll cycle.
3. Use the script file to download the files only from remote directories and not from the event directory of the FTP adapter.

Supported inbound operation

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

Event file locking

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

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

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

Rule-Based filtering of events

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

- FileName
- File Size
- 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.

Though using the rule is optional and specifying an event file mask is mandatory, the rule takes a higher precedence over the event file mask, when both a rule and an event file mask are specified. Event file mask is effective only when there is no rule specified. By default, an event file mask has "*.*" as the 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 31. Metadata filtering properties

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

Table 31. Metadata filtering properties (continued)

Property	Valid operators	Value	Prerequisites
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
LastModified	Greater than, Less than, Greater than or equal to, Less than or equal to, Equal to, Not equal to. **Note:** Select 'Equal to' operator when you choose the days of 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

Setting deployment and runtime properties

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

Before you begin

Before you can set the properties in this section, you must have created your adapter module. It should be displayed in WebSphere Integration Developer below the adapter project.

About this task

To set deployment and runtime properties, follow this procedure. For more information about the properties in this topic, refer to “Managed (J2C) connection factory properties” on page 145 topic.

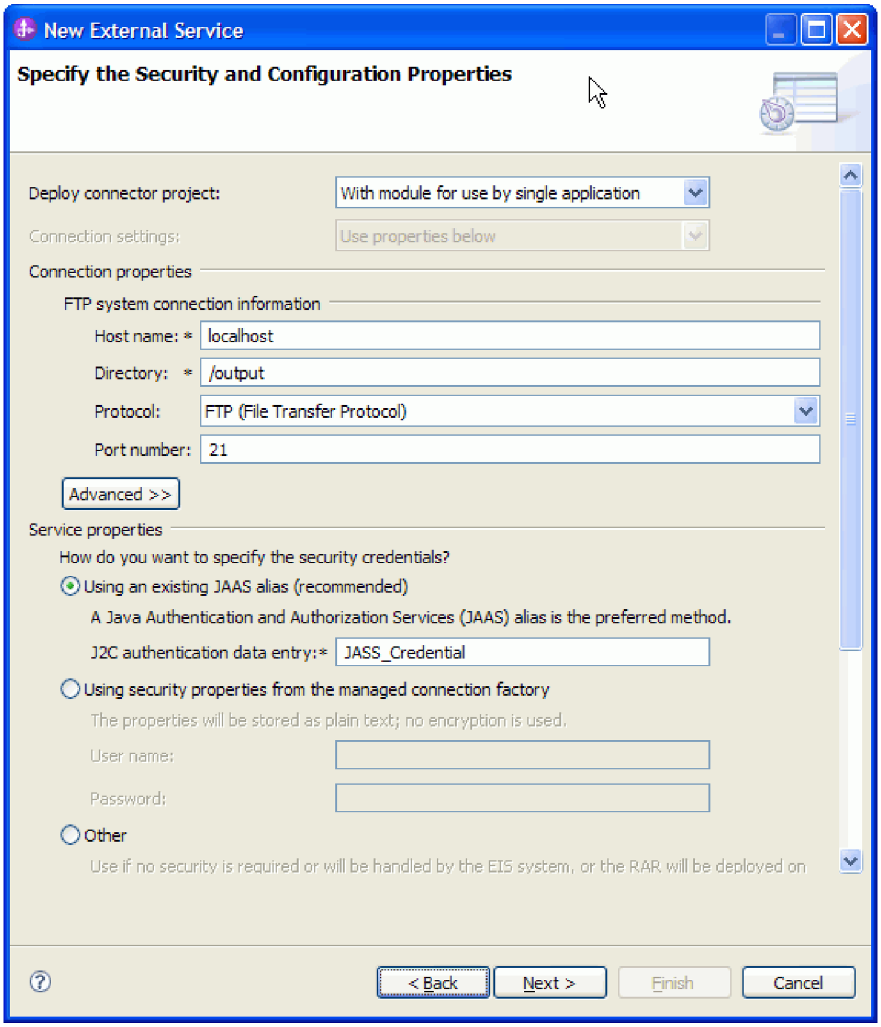
Procedure

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

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

     If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
3. Define the following FTP system connection information for your module. For more information, refer to “Managed (J2C) connection factory properties” on page 145 topic.
   - **Host name** - Specifies the host name of the FTP server.

- **Directory** - Specifies the output directory on the FTP server. If the value of the **Directory** field is set to <HOME_DIR>, the adapter performs the outbound operations in the user's home directory.
- **Protocol** - Specifies the protocol used to connect to the FTP server. Following are the protocols that can be specified:
  - FTP - File Transfer Protocol
  - FTP over SSL - File Transfer Protocol over Secure Socket Layer
  - FTP over TLS - File Transfer Protocol over Transport Layer Security
  - SFTP - Secure shell File Transfer Protocol
- **Port number** - Specifies the port number of the FTP server.



New External Service

Specify the Security and Configuration Properties

Deploy connector project: With module for use by single application

Connection settings: Use properties below

Connection properties

FTP system connection information

Host name: * localhost

Directory: * /output

Protocol: FTP (File Transfer Protocol)

Port number: 21

Service properties

How do you want to specify the security credentials?

● Using an existing JAAS alias (recommended)

A Java Authentication and Authorization Services (JAAS) alias is the preferred method.

J2C authentication data entry:* JASS_Credential

○ Using security properties from the managed connection factory

The properties will be stored as plain text; no encryption is used.

User name:

Password:

○ Other

Use if no security is required or will be handled by the EIS system, or the RAR will be deployed on

Figure 6. Specify the Security and Configuration Properties window

4. Click **Advanced** to specify additional properties, service properties, data format options, those that control working with a second FTP server, bidi formatting, a staging directory, logging and tracing, secure connection and sequence file selection. For more information, refer to “Managed (J2C) connection factory properties” on page 145 topic.
5. Specify the required security credentials in the **Service Properties** area:

- To use a J2C authentication alias, select the **Using an existing JAAS alias (recommended)** field, and specify the name of the alias in the **J2C Authentication Data Entry** field. You can specify an existing authentication alias or create one at any time before deploying the module. The name is case-sensitive and includes the node name.
- To use managed connection properties, select the **Using security properties from managed connection factory** field, and type the values in the **User name** and **Password** fields.
- **User name** - Specifies the name of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see "User name property (UserName)" on page 80.
- **Password** - Specifies the password of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see "Password property (Password)" on page 69
- To administer the user name and password from other mechanism, select **Other**.

6. If you have multiple instances of the adapter, expand **Logging and tracing** and set **Adapter ID** to a value that is unique for this instance. For more information about this property, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/rbp_ftp_resource_adapter_props.html.
7. Optional: In the Service properties section of the window, specify a Java Authentication and Authorization Services (JAAS) alias for the adapter to use at run time. This is the authentication alias that you set up on the FTP server. The name is case-sensitive. For more details, see "Creating the authentication alias".
8. In the **Data format options** field, select one of the following:
   - **Use default data binding 'FTPFileBaseDataBinding' for all operations**

     A non-configured data binding for all the operations used in the service.
   - **Use a data binding configuration for all operations**

     A configured data binding that will be used for all the operations used in the service.
   - **Specify a data binding for each operation**

     No default binding is specified. You can select a specific data binding for each operation used in the service.
9. Optional: Select the **Change the logging properties for the wizard** check box if you want to specify the log file output location or define the level of logging for this module. For information about logging levels, see the section on configuring logging properties in the Troubleshooting and support topic.

Results

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

What to do next

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

If you have selected the **Data format options** as Use a data binding configuration for all operations, proceed to Configuring the data binding and data handler.

Setting interaction specification properties and generating the service

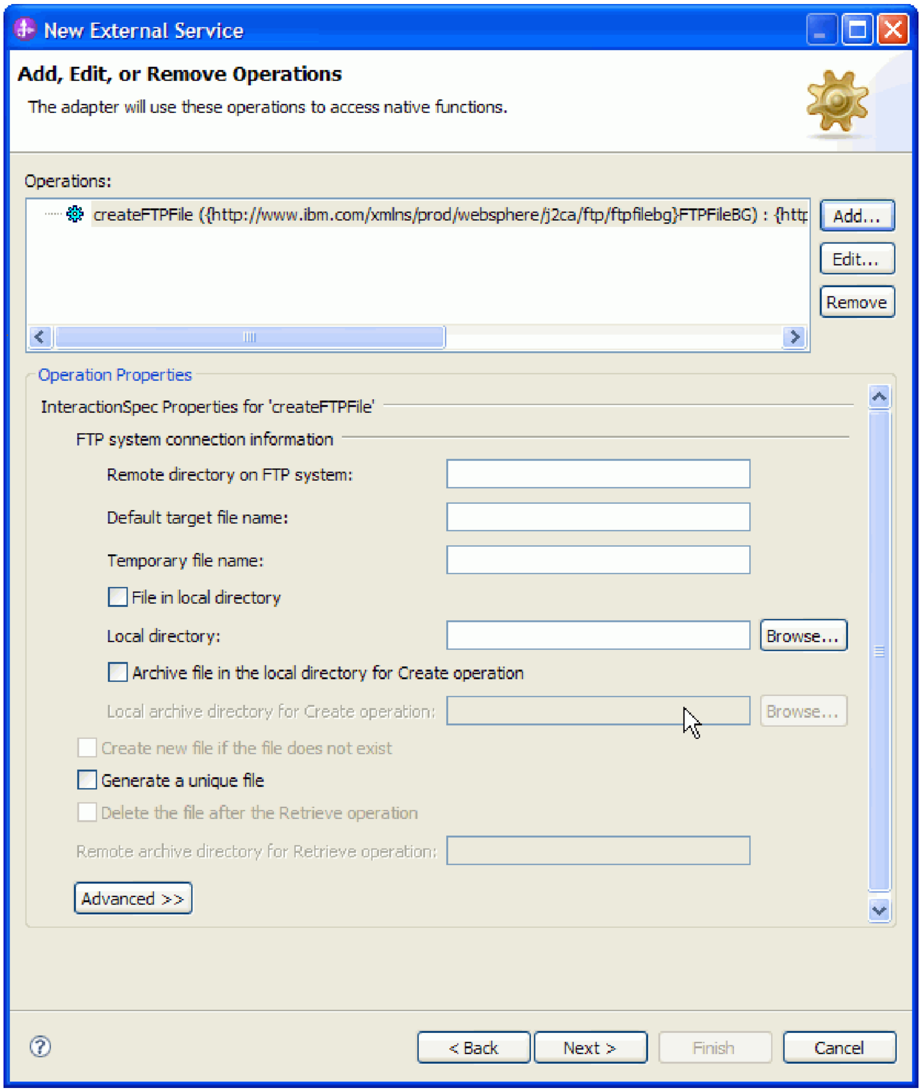
Interaction specification properties are optional. If you choose to set them, the values you specify is displayed as defaults in all parent FTP business objects generated by the external service wizard. Interaction specification properties control the interaction for an operation. While creating artifacts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

About this task

To set interaction specification properties and generate artifacts, follow this procedure. For more information, refer the topic wrapper and interaction specification properties.

Procedure

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



Add, Edit, or Remove Operations

The adapter will use these operations to access native functions.

Operations:

createFTPFile ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfilebg}FTPFileBG) : {http

Operation Properties

InteractionSpec Properties for 'createFTPFile'

FTP system connection information

Remote directory on FTP system:

Default target file name:

Temporary file name:

□ File in local directory

Local directory:

□ Archive file in the local directory for Create operation

Local archive directory for Create operation:

□ Create new file if the file does not exist

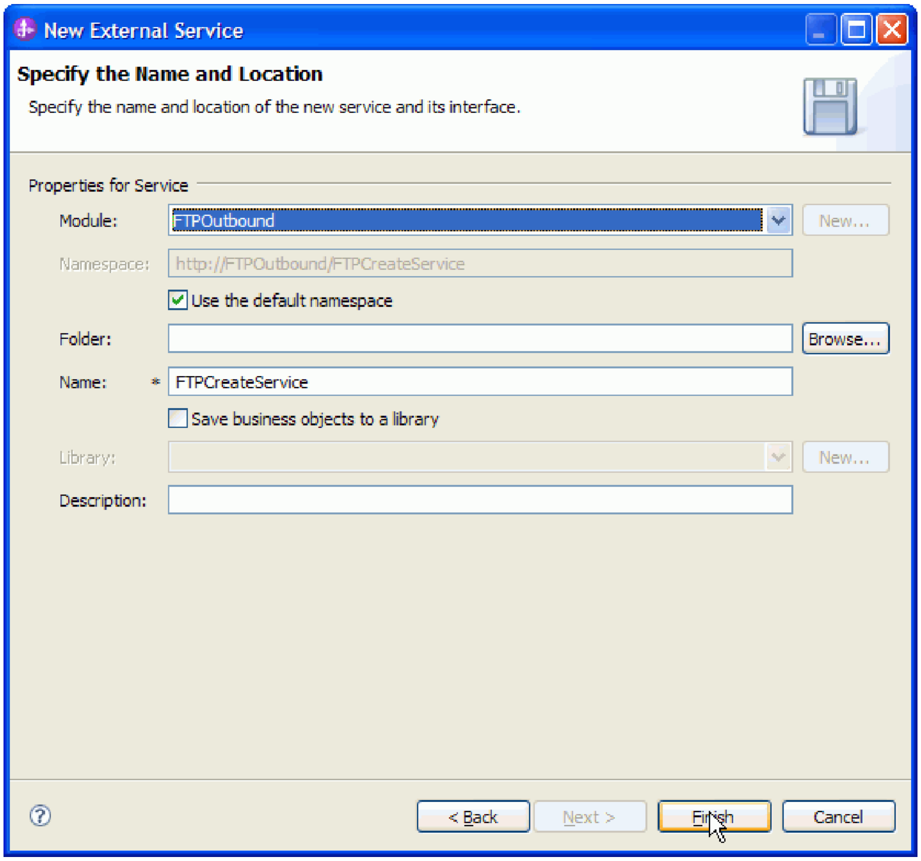
□ Generate a unique file

□ Delete the file after the Retrieve operation

Remote archive directory for Retrieve operation:

Figure 7. Interaction specification properties

2. In the Generate Service window, specify a name for the interface. This is the name that is displayed in the WebSphere Integration Developer assembly diagram.



New External Service

Specify the Name and Location

Specify the name and location of the new service and its interface.

Properties for Service

Module: FTPOutbound

Namespace: http://FTPOutbound/FTPCreateService

☑ Use the default namespace

Folder:

Name: * FTPCreateService

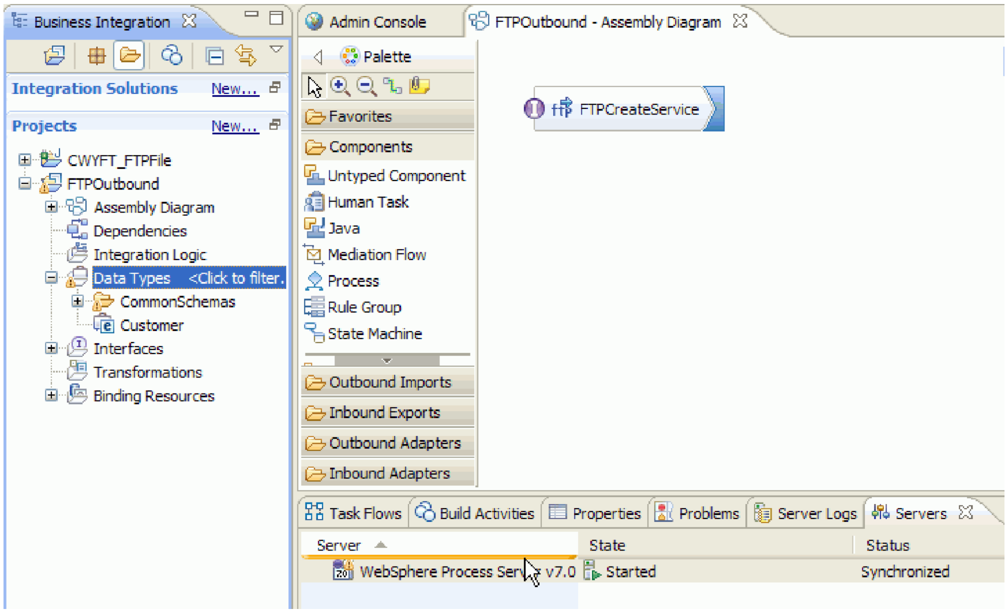
☐ Save business objects to a library

Library:

Description:

Figure 8. Specify the Name and Location window

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



Server	State	Status
WebSphere Process Server v7.0	Started	Synchronized

Figure 9. Interface in WebSphere Integration Developer

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

Results

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

What to do next

Deploy the module to the server.

Setting deployment and runtime properties

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

Before you begin

Before you can set the properties in this section, you must create your adapter module. It should be displayed in WebSphere Integration Developer below the adapter project.

About this task

To set deployment and runtime properties, follow this procedure. For more information about the properties in this topic, refer to “Activation specification properties” on page 53 topic.

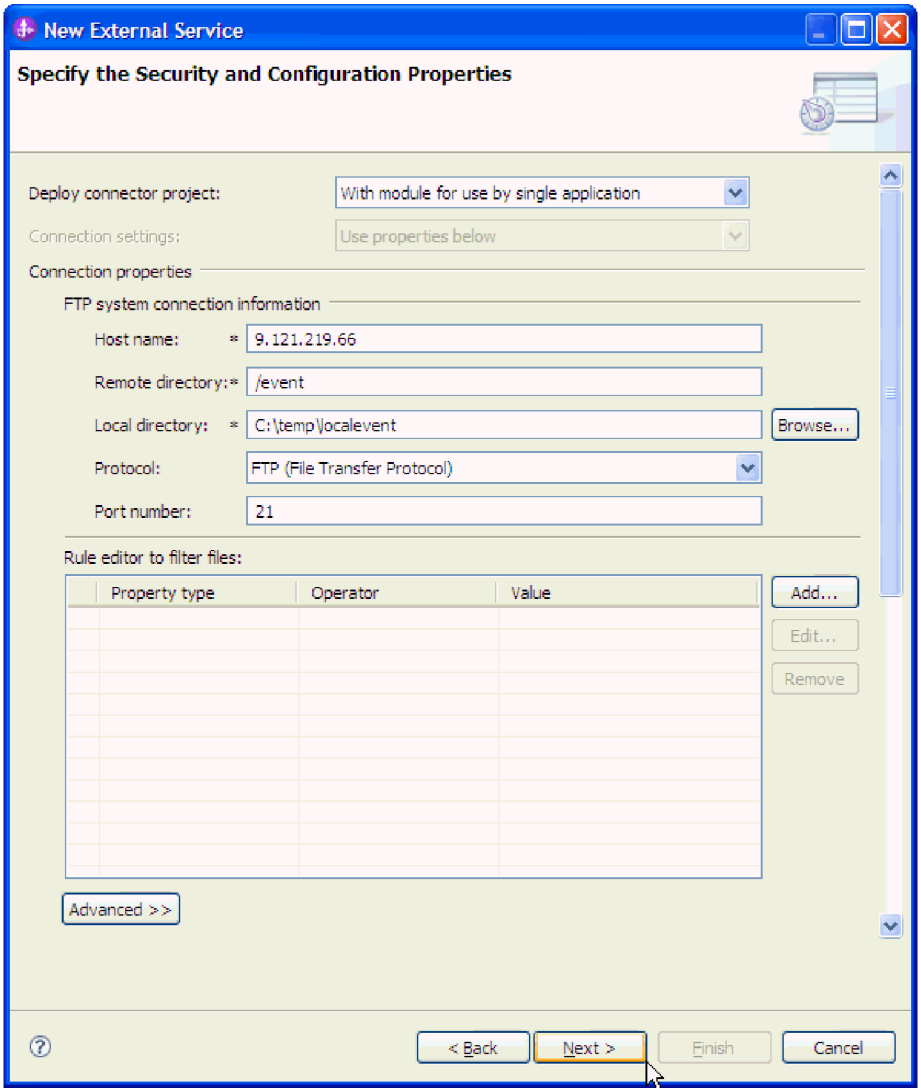
Procedure

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

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

     If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
3. Define the following FTP system connection information for your module. For more information, refer to “Activation specification properties” on page 53 topic.
   - **Host name** - Specifies the host name of the FTP server.

- **Remote directory** - Specifies the directory on the FTP server, where the adapter polls and picks up files. If the Remote directory is set to <HOME_DIR>, the adapter polls for event files in the home directory.
- **Local directory** - Specifies the directory on the adapter workstation where the event files are downloaded from the FTP server.
- **Protocol** - Specifies the protocol used to connect to the FTP server. Following are the protocols that can be specified:
  - FTP - File Transfer Protocol
  - FTP over SSL - File Transfer Protocol over Secure Socket Layer
  - FTP over TLS - File Transfer Protocol over Transport Layer Security
  - SFTP - Secure shell File Transfer Protocol
- **Port number** - Specifies the port number of the FTP server.



New External Service

Specify the Security and Configuration Properties

Deploy connector project: With module for use by single application

Connection settings: Use properties below

Connection properties

FTP system connection information

Host name: * 9.121.219.66

Remote directory: * /event

Local directory: * C:\temp\localevent

Protocol: FTP (File Transfer Protocol)

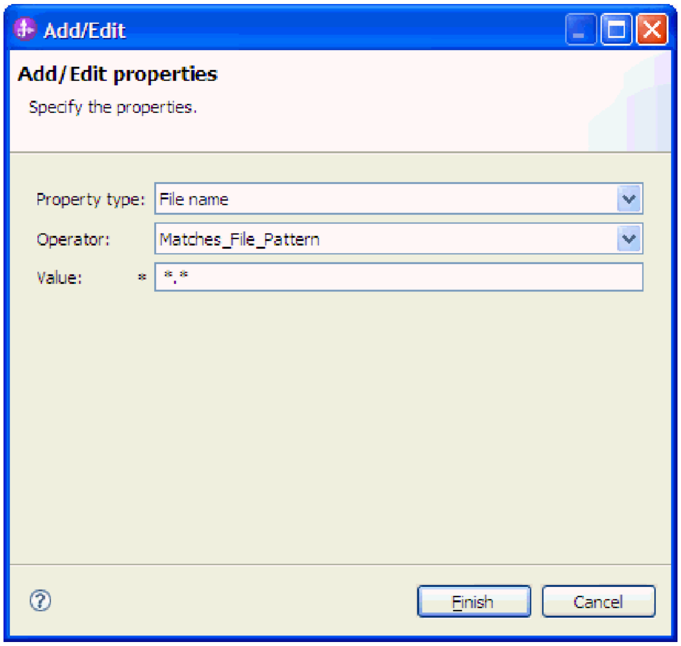
Port number: 21

Rule editor to filter files:

Property type	Operator	Value

Figure 10. Specify the Security and Configuration Properties window

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



Property type: File name

Operator: Matches_File_Pattern

Value: * *.*

Figure 11. Adding or editing a rule

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

b. Select the operator for the property type from the **Operator** list. Each of the property type metadata has its own operators.
   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) 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" on page 80.

5. Optional: Specify advanced properties by clicking **Advanced**. Expand each of the advanced sections to review the properties.
   - Event polling configuration
   - Event delivery configuration
   - Event persistence configuration
   - Additional configuration
   - FTP archiving configuration
   - Socks proxy server connection information
   - Secure configuration
   - Bidi properties
   - Logging and tracing properties

   The following sections describe the options that are available in the advanced property groups.
   - **Event polling configuration**
     a. In the **Interval between polling periods** field, specify the number of milliseconds that the adapter should wait between polling periods. For more information, see "Interval between polling periods (PollPeriod)" on page 70.
     b. In the **Maximum events in polling period** field, specify the number of events that the adapter should deliver in each polling period. For more information, see "Maximum events in polling period (PollQuantity)" on page 70.
     c. In the **Retry interval if connection fails** field, specify the number of milliseconds for the adapter to wait before trying to connect after a connection failure during polling. For more information, see "Retry interval if connection fails (RetryInterval)" on page 75.
     d. In the **Number of times to retry the system connection** field, specify the number of times to retry the connection before reporting a polling error. For more information, see "Number of times to retry the system connection (RetryLimit)" on page 75.
     e. If you want the adapter to stop if polling errors occur, select **Stop the adapter when an error is encountered while polling**. If you do not select this option, the adapter logs an exception but continues to run. For more information, see "Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 79.
     f. Select **Retry EIS connection on startup** if you want the adapter to retry a failed connection when starting. For more information, see "Retry EIS connection on startup (RetryConnectionOnStartup)" on page 74.

g. In the **Time interval for polling unchanged files** field, specify the time interval for which the adapter needs to monitor the files for any updates in the content before polling. The adapter polls those files that are not changed during the specified time interval. For more information, see "Time interval for polling unchanged files (fileUnchangedTimeInterval)" on page 64.

- **Event delivery configuration**
  a. In the **Type of delivery** field, select the delivery method. The methods are described in "Delivery type (DeliveryType)" on page 59.
  b. If you want to ensure that events are delivered only once and to only one export, select **Ensure once-only delivery**. This option might reduce performance but does not result in duplicate or missing an event delivery. For more information, see "Ensure once-only event delivery (AssuredOnceDelivery)" on page 57.
  c. In the **Retry limit for failed events** field, specify the number of times that the adapter will attempt to redeliver an event before marking it as failed. For more information, see "Retry limit for failed events (FailedEventRetryLimit)" on page 66.
- **Event persistence configuration**
  a. Select **Auto create event table** if you want the adapter to create the Event Persistence table. For more information, see "Auto create event table property (EP_CreateTable)" on page 57.
  b. In the **Event recovery table name** field, specify the name of the table that the adapter uses for event persistence. For more information, see "Event recovery table name property (EP_EventTableName)" on page 60
  c. In the **Event recovery data source (JNDI) name** field, specify the JNDI name of the data source that event persistence will use to connect to the JDBC database. For more information, see "Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)" on page 60
  d. In the **User name used to connect to event data source** field, specify the user name that the event persistence will use to connect to the database from the data source. For more information, see "User name used to connect to event data source property (EP_UserName)" on page 80
  e. In the **Password used to connect to event data source** field, specify the password that the event persistence will use to connect to the database from the data source. For more information, see "Password used to connect to event data source property (EP_Password)" on page 69.
  f. In the **Database schema name** field, specify the schema name of the database that the event persistence will use. For more information, see "Database schema name property (EP_SchemaName)" on page 59.
- **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 this pattern property (EventFileMask)" on page 73.
  b. In the **Sort event files** field, specify the sorting order of the event files being polled. For more information, see "Sort event files property (SortEventFiles)" on page 77.
  c. Select the **Enable remote verification** check box to enable remote verification. This property checks if the control and data connections are established with the same host (typically, the machine from which you establish a connection to the FTP server). The connection fails if the

control and data connections are not established. By default, the **Enable remote verification** check box is selected.

Note: This property is applicable only to FTP and FTPS protocols.
For more information, see “Enable remote verification property (enableRemoteVerification)” on page 73

d. In the **Encoding used by FTP server** field, specify the encoding of the FTP server. For more information, see “Encoding used by FTP server property (EISEncoding)” on page 60.
e. In the **File content encoding** field, specify the encoding used to read the event files. For more information, see “File content encoding property (FileContentEncoding)” on page 62.
f. In the **FTP server connection mode** field, specify the data connection mode used by the FTP server during file transfers. For more information, see “FTP server connection mode property (DataConnectionMode)” on page 61.
g. In the **File transfer type** field, specify the file transfer type used during inbound processing. For more information, see “File transfer type property (FileTransferType)” on page 65.
h. In the **Number of files to get at a time** field, specify the number of files retrieved from the remote FTP URL. For more information, see “Number of files to get at a time property (ftpGetQuantity)” on page 65.
i. In the **Number of poll periods between downloads** field, specify how frequently the adapter polls the FTP server. For more information, see “Number of poll periods between downloads property (ftpPollFrequency)” on page 65.
j. In the **Custom parser class name** field, specify the fully qualified class name of the custom parser that is used to parse the ls output. For more information, see “Custom parser class name property (CustomParserClassName)” on page 58.
k. Select **Pass only file name and directory, not the content** to specify that the file content of the event file is not sent to the export. For more information, see “Pass only file name and directory, not the content property (FilePassByReference)” on page 65.
l. Select **Include business object delimiter in the file content** to specify that the delimiter will be sent with the business object content for further processing. For more information, see “Include business object delimiter in the file content property (IncludeEndBODelimiter)” on page 67
m. Select **Split file content based on the size (bytes) or delimiter** to use the size in bytes or the delimiter to split the file content. For more information, see “Splitting function class name property” on page 79
n. In the **Specify criteria to split file content** field, specify that different values will be taken, based on the value of the SplittingFunctionClassName property. For more information, see “Specify criteria to split file content property (SplitCriteria)” on page 77.
o. In the **Split function class name** field, specify the fully qualified class name of the class file to be used to enable file splitting. For more information, see “Splitting function class name property” on page 79.
p. In the **Run FTP script file before downloading files** field, specifies the path of the script file that will be executed before downloading the files from the FTP server. For more information, see “Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)” on page 67.

q. In the **Run FTP script file after downloading files** field, specifies the path of the script file that will be executed after downloading the files from the FTP server. For more information, see "Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)" on page 66.

- **FTP archiving configuration**
  a. In the **Local archive directory** field, specify the absolute path of the local Archive directory. For more information, see "Local archive directory property (LocalArchiveDirectory)" on page 67.
  b. In the **File extension for local archive** field, specify the file extension used to archive the original event file. For more information, see "File extension for local archive property (originalArchiveExt)" on page 69.
  c. In the **Success file extension for local archive** field, specify the file extension used to archive all the successfully processed business objects. For more information, see "Success file extension for local archive property (SuccessArchiveExt)" on page 80.
  d. In the **Failure file extension for local archive** field, specify the file extension used to archive business objects in the event file that are not successfully processed. For more information, see "Failure file extension for local archive property (FailedArchiveExt)" on page 61.
  e. In the **Remote archive directory** field, specify the directory. For more information, see "Remote archive directory property (ftpArchiveDirectory)" on page 72.
  f. In the **File extension for remote archive** field, specify the file extension or suffix that the adapter uses to rename the remote FTP file. For more information, see "File extension for remote archive property (ftpRenameExt)" on page 62.
- **Socks proxy server connection information**
  a. In the **Host name** field, specify the host name of the machine used as a proxy server through which the adapter requests are routed to the FTP server. For more information, see "Host name property (SocksProxyHost)" on page 76.
  b. In the **Port number** field, specify the port number of the proxy server through which the adapter requests are routed to the FTP server. For more information, see "Port number property (SocksProxyPort)" on page 77.
  c. In the **User name** field, specify the user name for authenticating the proxy server. For more information, see "User name property (SocksProxyUserName)" on page 77.
  d. In the **Password** field, specify the password used to authenticate the proxy server. For more information, see "Password property (SocksProxyPassword)" on page 76.
- **Secure configuration**
  a. If you want to compare the host key of the SFTP server with the host keys known to the adapter:
     1) Select the **Enable remote server authentication for SFTP protocol** check box. The host key file has to be available with the host keys of the trusted server before the first attempt to connect to SFTP server is made. For more information, see Enable server verification property (EnableServerVerification)
     2) In the **Host key file** field, specify the absolute file path to the host key file. The host key file is created by the administrator and contains

the host keys of all the trusted servers. The Host key file property points to the file on the adapter workstation. For more information, see Host key file property (HostKeyFile)

b. If you want to enable public key authentication, specify the following properties:

   1) In the **Private key file** field, specify the private key used to authenticate to the Secure shell server. For more information, see "Private key file property (PrivateKeyFilePath)" on page 71.

   2) In the **Passphrase** field, specify the phrase used for enhanced security by encrypting the private key. For more information, see Passphrase property (Passphrase)

c. Specify the following properties for the FTPS protocol:

   1) In the **FTPS connection mode** field, specify the connection mode (Implicit or Explicit) to connect to the FTPS server, when FTPS is selected as protocol. For more information, see FTPS connection mode property (ftpsConnectionMode).

   2) In the **Data channel protection level** field, select the level of the data channel protection that you want to use:

      - Select **Private**, if the data transfer between the Adapter and the FTPS server has to be in an encrypted form.
      - Select **Clear**, if the data transfer between the Adapter and the FTPS server has to be in clear text form.

        For more information, see "Data channel protection level (dataProtectionLevel)" on page 149.

   3) In the **Keystore type** field, specify type of the keystore. For more information, see Keystore type property (keyStoreType).

   4) In the **Truststore file** field, specify the path of the truststore file that contains the certificates of the servers trusted by the adapter. For more information, see Truststore file property (trustStorePath).

   5) In the **Truststore password** field, specify the password of the truststore file. It is used to check the integrity of the truststore data. If this value is not specified, the integrity check will not be performed. For more information, see Truststore password property (trustStorePassword).

   6) In the **Keystore file** field, specify the path of the keystore file. The Keystore file contains the private key entry of the FTPS client and also contains a certificate chain for the corresponding public key. For more information, see Keystore file property (keyStorePath).

      Note: Both Keystore file and Truststore file properties share the properties of Keystore type.

   7) In the **Keystore password** field, specify the password of the keystore. It is used to check the integrity of the keystore data. If this value is not specified, integrity check will not be performed. For more information, see Keystore password property (keyStorePassword).

   8) In the **Key password** field, specify the password of the key that is used to recover the keys from the keystore. For more information, see Key password property (keyPassword).

- **Bidi properties**
- **Logging and tracing**

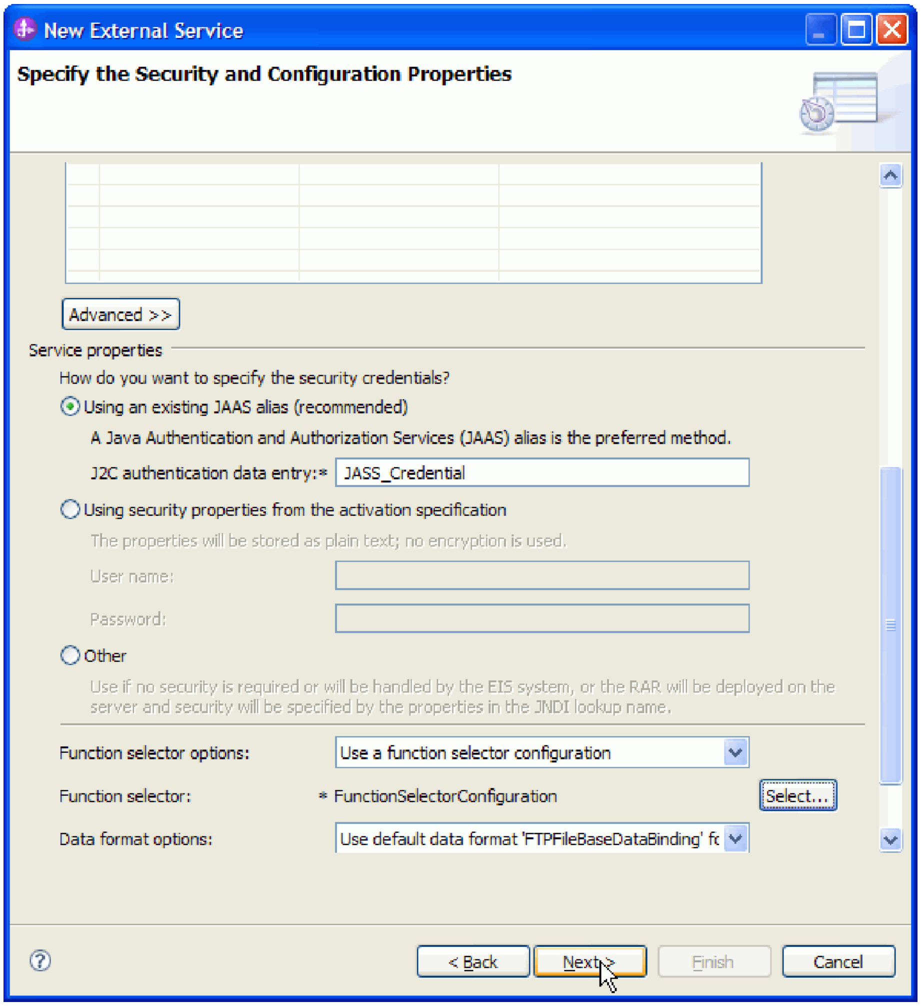
   a. If you have multiple instances of the adapter, expand and set Adapter ID to a value that is unique for this instance. For more information about this property, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/rbp_ftp_resource_adapter_props.html.
   b. Select **Disguise user data as 'XXX' in log and trace files** if you want to prevent sensitive user data from being written to log and trace files. For more information, see Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) .

6. Specify the required security credentials in the **Service Properties** area:
   - To use a J2C authentication alias, select the **Using an existing JAAS alias (recommended)** field, and specify the name of the alias in the **J2C Authentication Data Entry** field. You can specify an existing authentication alias or create one at any time before deploying the module. The name is case sensitive and includes the node name.
   - To use activation specification properties, select the **Using security properties from the activation specification** field, and type the values in the **User name** and **Password** fields.
   - **User name** - Specifies the name of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see “User name property (UserName)” on page 80.
   - **Password** - Specifies the password of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see “Password property (Password)” on page 69.
   - To administer the user name and password from other mechanism, select **Other**.
7. Select one of the options from the **Function selector** field. A function selector assigns incoming messages or requests to the correct operation on the service.
   - **Function selector options**

     For example, select **Use a Function Selector configuration**. If choosing to use this option, click **Next**.
   - **Function selector**

     If choosing this option, complete the following steps:

     a. Click **Select** next to the **Function Selector** field.


New External Service

Specify the Security and Configuration Properties

Service properties

How do you want to specify the security credentials?

● Using an existing JAAS alias (recommended)

A Java Authentication and Authorization Services (JAAS) alias is the preferred method.

J2C authentication data entry:* JASS_Credential

○ Using security properties from the activation specification

The properties will be stored as plain text; no encryption is used.

User name:

Password:

○ Other

Use if no security is required or will be handled by the EIS system, or the RAR will be deployed on the server and security will be specified by the properties in the JNDI lookup name.

Function selector options: Use a function selector configuration

Function selector: * FunctionSelectorConfiguration

Data format options: Use default data format 'FTPFileBaseDataBinding' fc

Figure 12. Specify the Security and Configuration Properties window

b. In the Select Function Selector window, select the option, **Use existing function selector from the list**. A list of available function selectors is displayed. Select the function selector (this example uses FilenameFunctionSelector properties). Click **Next**.

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

8. Click **Finish** in the New Function Selector Configuration window.
9. Click **Next** in the Service Configuration Properties window.

Results

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

What to do next

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

If you have selected the **Data format options** as Use a data binding configuration for all operations, proceed to Configuring the data binding and data handler.

Wrapper and interaction specification properties

Wrapper properties are attributes of the wrapper business object that enable an application programmer to control an operation for the business objects in a wrapper. Interaction specification properties control the interaction for an operation for the entire adapter.

The external service wizard sets the interaction specification properties when you configure the adapter. You can change some, but not all, of these properties. However, you can change some properties for outbound operations. Use the assembly editor to change these properties, which reside in the method binding of the import. You set the wrapper properties using the WebSphere Integration Developer test client or programmatically at run time.

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

Table 32. Interaction specification properties

Property name		Description
In the wizard	In the wrapper business object	
Remote archive directory for retrieve operation	ArchiveDirectoryForRetrieve	The adapter optionally archives the file to this folder before it is deleted during a Retrieve operation.
Create new file if the file does not exist	CreateFileIfNotExists	If the file does not exist on the FTP server, the adapter creates the file when this property is set to `true` during Append and Overwrite operations.
FTP server connection mode	DataConnectionMode	Data connection mode used by the FTP server during file transfers.
Delete the file after retrieve operation	DeleteOnRetrieve	The adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.
Remote directory on FTP system	DirectoryPath	Absolute path of the directory on the FTP server where the outbound operation must be performed.
“Data channel protection level (dataProtectionLevel)” on page 45	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
File content encoding	FileContentEncoding	Encoding used while writing to the file.
File in local directory	FileInLocalDirectory	If set to `true` during a create operation, the file content is picked from the local directory path of the adapter workstation.
Default target file name	Filename	Name of the file in the directory provided by the DirectoryPath property.
File transfer type	FileTransferType	File transfer type used during outbound operations.

Table 32. Interaction specification properties (continued)

Generate a unique file	GenerateUniqueFile	The adapter creates a unique file name when this property is set to `true`.
Host name property	SecondServerHostName	Host name of the second FTP server.
Delimiter between business objects in the file property	IncludeEndBODelimiter	File content is appended with this value.
Local archive directory for create operation	LocalArchiveDirForCreate	When `LocalArchivingEnabledForCreate` is set to `true` during a create operation, the file is saved to the local workstation in this directory.
Archive file in the local directory for create operation	LocalArchivingEnabledForCreate	When set to `true`, the file is saved to the local workstation during a create operation.
Local directory	LocalDirectoryPath	The file is picked from this directory.
(Not available)	ResumeFailedTransfer	When this property is set to `true` during a create operation, the adapter resumes the transfer of files from the point at which the transfer of file was interrupted due to connection error.
Port number	SecondServerPortNumber	Port number of the second FTP server.
Protocol	SecondServerProtocol	Specifies the protocol used to connect to the second server.
Script File Parameters	ScriptFileParameters	The parameters required by the FTP script file.
Directory	SecondServerDirectory	Directory path of the second FTP server during a ServerToServerFileTransfer operation.
Password	SecondServerPassword	Password of the second FTP server during a ServerToServerFileTransfer operation.
User name	SecondServerUsername	User name of the second FTP server during a ServerToServerFileTransfer operation.
Specify criteria to split file content	SplitCriteria	The delimiter that separates the business objects in the event file.
Split function class name	SplittingFunctionClassName	The fully qualified class name of the class file to be used to enable file splitting.
Staging directory	StagingDirectory	The file is first created into this directory.
Temporary file name	TemporaryFilename	Specifies the temporary file name for the create operation.

Archive file in the local directory for create operation property (LocalArchivingEnabledForCreate)

During outbound create operations, when the file content is coming as part of the business object from a J2EE application and this property is set to `true`, the file is saved to the local workstation in the LocalArchiveDirForCreate directory before performing the outbound operation.

Table 33. Archive file in the local directory for create operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Create new file if the file does not exist property (CreateFileIfNotExists)

During outbound Append and Overwrite operations, if the file does not exist on the FTP server, the adapter creates the file when this property is set to `true`. If this property is `false` and file does not exist, the adapter sends an error.

Table 34. Create new file if the file does not exist property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the "PBSZ 0" command before issuing the PROT command.

Table 35. Data channel protection level property characteristics

Required	No

Possible values	`Private - Data is transferred in encrypted form`
	`Clear - Data is transferred as clear text`

Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Delete the file after retrieve operation (DeleteOnRetrieve)

During an outbound Retrieve operation, the adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.

Table 36. Delete the file after retrieve operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Default target file name property (Filename)

Name of the file to be used during outbound operations.

Table 37. Default target file name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Delimiter between business objects in the file property (IncludeEndBODelimiter)

File content is appended with this value. Used during the outbound create, append, and overwrite operations.

Table 38. Include business object delimiter in the file content property characteristics

Required	No
Default	For the create and overwrite operations, no default value is set. For the append operation, the default value is `<EndBO>`. For the append operation, the following rules apply: • If the delimiter is set to `null` in the business object wrapper, no delimiter is used to separate the business objects. • If the IncludeEndBODelimiter property is not set in the business object wrapper, and the value in the interaction specification is also `null`, the default is `<EndBO>`. • If a specific delimiter value is specified in the business object wrapper, the specified value will be appended. • If both the business object wrapper and the interaction specification have specified values, the business object wrapper value takes precedence.
Property type	String
Globalized	Yes

Directory property (SecondServerDirectory)

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

Table 39. Directory property characteristics

Required	No
Default	None
Property type	String
Usage	For interaction specification properties, the directory located on the FTP server and used in outbound operation represents the absolute path of the FTP directory. For example: `/home/usr/output`. It does not contain any host name or URL information. For wrapper business object properties, the URL of the second server to which the ServerToServerFileTransfer outbound operation is performed. For example: The syntax for specifying the FTP URL is: `ftp://[UserId:password@]FTPserver[:port]/DirectoryForSecondServer`.

Table 39. Directory property characteristics (continued)

Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used while writing to the file. If this property is not specified, the adapter tries to read without using any specific encoding. You can specify any Java supported encoding set.

Table 40. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File in local directory property (FileInLocalDirectory)

During outbound create operations, if this property is set to `true`, the file content is not available in the business object. The file is retrieved from the local directory on the adapter workstation. During outbound retrieve operations, if this property is set to `true`, the file content is not sent to the J2EE application as part of the business object. The file is saved to the local directory of the adapter workstation.

Table 41. File in local directory property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during outbound operations. Takes either ASCII or binary.

Table 42. File transfer type property characteristics

Required	No
Default	`binary`
Property type	String
Globalized	No

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Takes either `active` or `passive`. This value is used only when a file transfer is taking place. This property is not used when performing a server to server file transfer outbound operation.

Table 43. FTP server connection mode property characteristics

Required	No
Default	`active`

Table 43. FTP server connection mode property characteristics (continued)

Property type	String
Possible values	`active` or `passive`
Globalized	No

Generate a unique file (GenerateUniqueFile)

During outbound Create operation, the adapter creates a unique file name when this property is `true`. The adapter ignores any value that is set for the Filename property when this property is set to `true`.

Note: The adapter does not support both GenerateUniqueFile and StagingDirectory options at the same time.

Table 44. Generate unique file property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No
Restrictions	The FTP server must support RFC1123 to use this feature.

Host name property (SecondServerHostName)

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

Table 45. Host name property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Local archive directory for create operation property (LocalArchiveDirForCreate)

During outbound create operations, when the file content is coming as part of the business object and LocalArchivingEnabledForCreate is set to `true`, the file is saved to the local workstation in this directory.

Table 46. Local archive directory for create property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalArchiveDirForCreate directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Local directory property (LocalDirectoryPath)

During outbound create operations, when FileInLocalDirectory property is set to `true`, the file content is not available in the business object. Instead the file is picked from this directory. During outbound retrieve operations, when FileInLocalDirectory property is set to `true`, the file content is not sent as part of business object. The file is saved to this directory.

Table 47. Local directory property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalDirectoryPath directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Port number property (SecondServerPortNumber)

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

Table 48. Port number property characteristics

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

Protocol property (SecondServerProtocol)

Protocol that is used to establish a connection to the second server. The FTP protocol is used in establishing the connection.

Table 49. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Password property (SecondServerPassword)

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

Table 50. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Remote archive directory for retrieve operation property (ArchiveDirectory
ForRetrieve)

During an outbound Retrieve operation, the adapter optionally archives the file to this folder before it is deleted. The archive directory must exist.

Table 51. Remote archive directory for retrieve operation property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Remote directory on FTP system property (DirectoryPath)

Absolute path of the directory on the FTP server where the outbound operation must be performed for all operations except ExecuteFTPScript, or the directory path on the local adapter workstation for the ExecuteFTPScript operation only. The directory must exist.

Note: If the value <HOME_DIR> is specified as the DirectoryPath, the outbound operations will be performed in the user's home directory.

Table 52. Remote directory on FTP system property characteristics

Required	No
Default	None
Property type	String
Usage	The DirectoryPath directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

ResumeFailedTransfer

This property supports resuming the transfer of files, which were interrupted due to an error in connection to the FTP server.

Note: This property is applicable only to outbound processing.

Table 53. ResumeFailedTransfer property characteristics

Required	No
Default	`false`
Property type	Boolean
Usage	During a create operation, when this property is set to `true`, the adapter resumes the transfer of files from the point at which the transfer of file was interrupted due to an error in connection.
Globalized	No

Script File Parameters property (ScriptFileParameters)

During an outbound ExecuteFTPScript operation, the parameters required by the FTP script file are set in this property. During run time, the adapter replaces the parameters with these values.

Table 54. Script File Parameters property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Specify criteria to split file content property (SplitCriteria)

This property accepts different values based on the value of the SplittingFunctionClassName property.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
  - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
  - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=`0`, chunking is disabled.

Table 55. Specify criteria to split file content property characteristics

Required	No
Default	`0`
Property type	String
Globalized	Yes

Split function class name property (SplittingFunction ClassName)

Takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The com.ibm.j2ca.utils.filesplit.SplitByDelimiter class that splits the event file based on delimiter.
- The com.ibm.j2ca.utils.filesplit.SplitBySize class that splits the event file based on the event file size.

The delimiter or file size is provided in the SplitCriteria property.

Table 56. Split function class name property characteristics

Required	No
Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Globalized	No

Staging directory property (StagingDirectory)

During outbound create operations, the file will be created in this directory first. When the file creation is complete, the file is copied to the directory specified in the DirectoryPath property. This staging directory is also used for Append and Overwrite operations where the specified file is copied to the StagingDirectory, if present. The appended or overwritten content is then moved back to the original specified directory. If StagingDirectory is not specified, the operation is run in the actual required directory.

Note: The adapter does not support both StagingDirectory and GenerateUniqueFile options at the same time.

Table 57. Staging directory property characteristics

Required	No
Default	None
Property type	String
Usage	The StagingDirectory directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Temporary file name property (TemporaryFilename)

This property specifies the temporary file name for the create operation. After successful creation of the file, the file gets renamed to the value specified in the 'Default target file name' property.

Table 58. Temporary file name property characteristics

Required	No
Possible values	All valid file names
Default	None
Property type	String
Usage	This property is used in the create operation. If the temporary file name is specified, the file is created with the temporary file name. After the file is successfully created, the file is renamed to the value that is specified in the 'Default target file name' property.
Example	xyz.tmp
Globalized	No

User name property (SecondServerUsername)

User name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 59. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Activation specification properties

Activation specification properties are properties that hold the inbound event processing configuration information for a message endpoint.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint (a message driven bean).

You set the activation specification properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the administrative console.

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

Table 60. Activation specification properties

Property name		
In the wizard	In the administrative console	Description
"Ensure once-only event delivery (AssuredOnceDelivery)" on page 57	AssuredOnceDelivery	Specifies whether the adapter provides assured once delivery of events.
"Auto create event table property (EP_CreateTable)" on page 57	`EP_CreateTable`	Tells the adapter whether to create the Event Persistence table
"Create Table property (CreateTable)" on page 57	`CreateTable`	When set to `true`, the event table and related indexes are created
"Custom parser class name property (CustomParserClassName)" on page 58	`CustomParserClassName`	Fully qualified class name of the custom parser which is used to parse the `ls –l` output
"Data channel protection level (dataProtectionLevel)" on page 58	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
"Database Password property (DatabasePassword)" on page 58	`DatabasePassword`	Password used by event persistence for retrieving the JDBC database connection from the data source
"Database schema name property (EP_SchemaName)" on page 59	`EP_SchemaName`	Schema name of the database used by event persistence
"Database Username property (DatabaseUsername)" on page 59	`DatabaseUsername`	User name used by event persistence for retrieving the JDBC database connection from the data source
"FTP server connection mode property (DataConnectionMode)" on page 61	`DataConnectionMode`	Data connection mode used by the FTP server during file transfers
"FTPS connection mode property (ftpsConnectionMode)" on page 61	`ftpsConnectionMode`	Specifies the FTPS connection mode used to set up connection to the FTPS server.
(Not available)	`DefaultObjectName`	Supported for compatibility with earlier versions

Table 60. Activation specification properties (continued)

"Delivery type (DeliveryType)" on page 59	DeliveryType	Determines the order in which events are delivered by the adapter to the export.
"Encoding used by FTP server property (EISEncoding)" on page 60	EISEncoding	Encoding of the FTP server
(Not available)	EventContentType	Supported for compatibility with earlier versions
"Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)" on page 60	EP_DataSource_JNDIName	JNDI name of the data source used by event persistence to get the JDBC database connection
"Event recovery table name property (EP_EventTableName)" on page 60	EP_TableName	Name of the table that is used by the adapter for event persistence
"Failure file extension for local archive property (FailedArchiveExt)" on page 61	FailedArchiveExt	File extension used to archive business objects in the event file that are not successfully processed
"File content encoding property (FileContentEncoding)" on page 62	FileContentEncoding	Encoding used to read the event files
"File extension for remote archive property (ftpRenameExt)" on page 62	ftpRenameExt	File extension or suffix that the adapter uses to rename the remote FTP file
"Keystore file property (keyStorePath)" on page 62	keyStorePath	Specifies the path of the keystore that contains the private key entries.
"Keystore password property (keyStorePassword)" on page 63	keyStorePassword	Specifies the password that is used to encrypt the keystore.
"Key password property (keyPassword)" on page 63	keyPassword	Specifies the password that is used to encrypt the key.
"Keystore type property (keyStoreType)" on page 63	keyStoreType	Specifies the type of the keystore.
"Pass only file name and directory, not the content property (FilePassByReference)" on page 65	FilePassByReference	Specifies that the file content of the event file is not sent to the export
"File transfer type property (FileTransferType)" on page 65	FileTransferType	File transfer type used during inbound processing
"Number of files to get at a time property (ftpGetQuantity)" on page 65	ftpGetQuantity	Determines the number of files retrieved from the remote FTP URL
"Number of poll periods between downloads property (ftpPollFrequency)" on page 65	ftpPollFrequency	Determines how frequently the adapter polls the FTP server
Retry limit for failed events	FailedEventRetryLimit	The number of times the adapter attempts to redeliver an event before marking the event as failed.
"Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)" on page 66	ftpScriptFileExecutedAfterInbound	Specifies the path of the script file that will be executed after downloading the files from the FTP server
"Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)" on page 67	ftpScriptFileExecutedBeforeInbound	Specifies the path of the script file that will be executed before downloading the files from the FTP server

Table 60. Activation specification properties (continued)

"Host name property (HostName)" on page 67	`HostName`	Host name of the FTP Server to which the connection is established
"Include business object delimiter in the file content property (IncludeEndBODelimiter)" on page 67	`IncludeEndBODelimiter`	When set to `true`, the delimiter is sent with the business object content for further processing
"Local archive directory property (LocalArchiveDirectory)" on page 67	`LocalArchiveDirectory`	Absolute path of the local Archive directory
"Local directory property (LocalEventDirectory)" on page 68	`LocalEventDirectory`	Local system directory into which the adapter downloads event files from the FTP site
"Maximum connections (MaximumConnections)" on page 68	MaximumConnections	The maximum number of connections that the adapter can use for inbound event delivery.
"Minimum connections (MinimumConnections)" on page 69	MinimumConnections	The minimum number of connections that the adapter can use for inbound event delivery.
"File extension for local archive property (originalArchiveExt)" on page 69	`OriginalArchiveExt`	File extension used to archive the original event file
Passphrase property	passPhrase	Used for enhanced security by encrypting the private key
"Password property (Password)" on page 69	`Password`	Password of the user who has privileges to connect to the FTP server and perform FTP operations
"Password used to connect to event data source property (EP_Password)" on page 69	`EP_Password`	Password used during event persistence
"Interval between polling periods (PollPeriod)" on page 70	PollPeriod	The length of time that the adapter waits between polling periods.
"Maximum events in polling period (PollQuantity)" on page 70	PollQuantity	The number of events the adapter delivers to the export during each poll period.
"Port number property (PortNumber)" on page 71	`PortNumber`	Port number of the FTP server
"Private key file property (PrivateKeyFilePath)" on page 71	`PrivateKeyFilePath`	Private key used to authenticate to the Secure shell server
"Protocol property (Protocol)" on page 71	`Protocol`	Specifies if the connection to the FTP server is normal FTP or secure FTP.
"Retrieve files with this pattern property (EventFileMask)" on page 73	`EventFileMask`	Filter for the event files
Retry EIS connection on startup	RetryConnectionOnStartup	Controls whether the adapter retries the connection to the FTP server if it cannot connect at startup.
Time between retries in case of system connection failure (milliseconds)	RetryInterval	The length of time that the adapter waits between attempts to establish a new connection after an error during inbound operations.
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.

Table 60. Activation specification properties (continued)

"Remote archive directory property (ftpArchiveDirectory)" on page 72	`ftpArchiveDirectory`	Relative path of the archive directory on the FTP server
"Remote directory property (EventDirectory)" on page 72	`EventDirectory`	Remote directory of the FTP server from where the event files are retrieved for inbound processing
Enable server verification	EnableServerVerification	Enables the remote server verification for SFTP protocol
Host key file	HostKeyFile	The absolute path of the host key file that contains the host keys of the trusted servers
"Host name property (SocksProxyHost)" on page 76	`SocksProxyHost`	Host name of the machine used as a proxy server
"Password property (SocksProxyPassword)" on page 76	`SocksProxyPassword`	Password used to authenticate the proxy server
"Port number property (SocksProxyPort)" on page 77	`SocksProxyPort`	Port number of the proxy server
"User name property (SocksProxyUserName)" on page 77	`SocksProxyUserName`	User name used to authenticate the proxy server
"Sort event files property (SortEventFiles)" on page 77	`SortEventFiles`	Determines the sorting order of event files being polled
"Specify criteria to split file content property (SplitCriteria)" on page 77	`SplitCriteria`	Takes different values based on the value of the SplittingFunctionClassName property
"Splitting function class name property" on page 79	`SplittingFunctionClassName`	Takes the fully qualified class name of the class file to be used to enable file splitting
"Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 79	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling.
"Success file extension for local archive property (SuccessArchiveExt)" on page 80	`SuccessArchiveExt`	File extension used to archive all the successfully processed business objects
"Truststore file property (trustStorePath)" on page 63	trustStorePath	Specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.
"Truststore password property (trustStorePassword)" on page 64	trustStorePassword	Specifies the password of the truststore.
"Time interval for polling unchanged files (fileUnchangedTimeInterval)" on page 64	fileUnchangedTimeInterval	Specifies the time interval for the adapter to monitor the files for any updates in the content.
"User name property (UserName)" on page 80	`UserName`	Name of the user who has privileges to connect to the FTP server and perform FTP operations
"User name used to connect to event data source property (EP_UserName)" on page 80	`EP_UserName`	User name used by event persistence for getting the database connection
Rule editor to filter files	ruleString	The collection of rules used to filter the events.

Table 60. Activation specification properties (continued)

"Enable remote verification property (enableRemoteVerification)" on page 73	`enableRemoteVerification`	Used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

Ensure once-only event delivery (AssuredOnceDelivery)

This property specifies whether to provide ensure once-only event delivery for inbound events.

Table 61. 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

Auto create event table property (EP_CreateTable)

Tells the adapter whether to create the Event Persistence table. If the value is `true` and table does not exist then the adapter creates the table. If the value is `false` the adapter does not create the table.

Table 62. Auto create event table property characteristics

Required	No
Default	`true`
Property type	Boolean
Globalized	No

Create Table property (CreateTable)

When set to `true`, the event table and related indexes are created. For troubleshooting table creation errors, set this property to `false`. The table and indexes can then be created manually.

Table 63. Create Table property characteristics

Required	No
Default	`true`
Property type	Boolean
Globalized	No

Custom parser class name property (CustomParserClassName)

Fully qualified class name of the custom parser which is used to parse the `ls –l` output. Used only when the `ls –l` output deviates from standard output.

Table 64. Custom parser class name property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the "PBSZ 0" command before issuing the PROT command.

Table 65. Data channel protection level property characteristics

Required	No

Possible values	`Private - Data is transferred in encrypted form`
	`Clear - Data is transferred as clear text`

Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Database Password property (DatabasePassword)

Password used by event persistence for retrieving the JDBC database connection from the data source.

Table 66. Database Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database schema name property (EP_SchemaName)

Schema name of the database used by event persistence.

Table 67. Database schema name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database Username property (DatabaseUsername)

User name used by event persistence for retrieving the JDBC database connection from the data source.

Table 68. Database username property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Delivery type (DeliveryType)

This property specifies the order in which events are delivered by the adapter to the export.

Table 69. 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.
Globalized	No
Bidi supported	No

Encoding used by FTP server property (EISEncoding)

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

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

Specify any Java-supported encoding set for this attribute.

Table 70. Encoding used by FTP server property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)

JNDI name of the data source used by event persistence to get the JDBC database connection. The data source must be created in WebSphere Process Server. The database name specified while creating the data source must exist.

Table 71. Event recovery data source (JNDI) name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery table name property (EP_EventTableName)

Name of the table that is used by the adapter for event persistence. When using multiple activation specifications, this value must be unique for each. The same table name must not be used by other instances of same adapter or a different adapter. If the table does not exist in the database, the adapter will create the table.

Table 72. Event recovery table name property characteristics

Required	No

Table 72. Event recovery table name property characteristics (continued)

Default	FTPTABLE
Property type	String
Globalized	Yes

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Accepts either active or passive settings.

Table 73. FTP server connection mode property characteristics

Required	No
Default	active
Property type	String
Globalized	No

FTPS connection mode property (ftpsConnectionMode)

This property is used to specify the connection mode when establishing a connection with the FTPS server. The WebSphere Adapter for FTP now supports both Implicit and Explicit connection modes. This property is used when you select either FTP over secure sockets layer (SSL) protocol or FTP over transport layer security (TLS) protocol.

Table 74. FTPS connection mode property characteristics

Required	No
Possible values	Explicit Implicit
Default	Explicit
Property type	String
Usage	This property represents the mode used to connect to the FTPS server. When this property is set to: • Explicit connection mode, initially the connection is established as a normal FTP connection. To send sensitive information, such as password the adapter switches to a secure FTP connection by issuing an AUTH command. **Note:** The default port for Explicit connection mode is 21. • Implicit connection mode, the connection is established as a secure FTP connection. All communications between the adapter and the server continues in a secure mode. There is no exchange of clear text information between the Adapter and the server. **Note:** The default port for Implicit connection mode is 990.
Globalized	No
Bidi supported	No

Failure file extension for local archive property (FailedArchiveExt)

File extension used to archive business objects in the event file that are not successfully processed. This property is used only when LocalArchiveDirectory is valid and exists.

Table 75. Failure file extension for local archive property characteristics

Required	No
Default	`fail`
Property type	String
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used to read the event files based on the EndBODelimiter property and during string to byte[] conversions. If not specified, the adapter attempts to read without any specific encoding. You can specify any Java supported encoding set.

Table 76. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File extension for remote archive property (ftpRenameExt)

File extension or suffix that the adapter uses to rename the remote FTP file after the connector has polled for it. Renaming the file prevents the connector from polling the same file in the next poll cycle. The adapter can be configured to rename the processed event file and move it to an archive directory.

Table 77. File extension for remote archive property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Keystore file property (keyStorePath)

This property specifies the path of the keystore that contains the private key entries.

Table 78. Keystore file property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the absolute path of the keystore file on the adapter machine (on which the adapter is running). The keystore file contains the private key entry of the FTPS client. It is also accompanied by a certificate chain for the corresponding public key. The keystore data is used to authenticate the clients identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Keystore password property (keyStorePassword)

This property specifies the password that is used to encrypt the keystore.

Table 79. Keystore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the keystore. It is used to check the integrity of the keystore data. If the value is not specified, integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Key password property (keyPassword)

This property specifies the password that is used to encrypt the key.

Table 80. Key password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the key that is used to recover the key from the keystore. The property is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Keystore type property (keyStoreType)

This property specifies the type of keystore.

Table 81. Keystore type property characteristics

Required	No
Possible values	JKS and PKCS12
Default	JKS
Property type	String
Usage	This property specifies the type of the keystore. It is applicable only if you select FTP over SSL or FTP over TLS as the protocol. This property is also applicable for the type of the truststore.
Globalized	No
Bidi supported	No

Truststore file property (trustStorePath)

This property specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.

Table 82. Truststore file property characteristics

Required	This property is required only if you set the protocol as FTP over SSL or FTP over TLS
Default	None
Property type	String
Usage	This property specifies the absolute path of the truststore file on the adapter machine (on which the adapter is running). The truststore file contains the certificates of FTPS servers trusted by the adapter and is used to authenticate the servers identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Truststore password property (trustStorePassword)

This property specifies the password of the truststore.

Table 83. Truststore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password for the truststore. It is used to check the integrity of the truststore data. If the value is not specified, the integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Time interval for polling unchanged files (fileUnchangedTimeInterval)

This property specifies the time interval for the adapter to monitor the files for any updates in the content. The adapter polls only those files that are not changed during the specified time interval.

Table 84. Time interval for polling unchanged file

Required	No
Default	0
Unit of measure	Milliseconds
Property type	Integer
Usage	This property enables the adapter to poll 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. If the value is set to '0' the adapter polls the files instantly and does not check if the files are being modified.
Globalized	No
Bidi supported	No

Pass only file name and directory, not the content property (FilePassByReference)

Specifies that the file content of the event file is not sent to the export.

If set to `true`, the file is appended with a timestamp and sent to the LocalArchiveDirectory. The timestamp prevents errors and overwrites to the file when another file with the same name is received. This property can be set to `true` only when the LocalArchiveDirectory property is set and the specified directory exists. This property is used only for PassThrough inbound processing. When enabled, the file is not split into chunks.

Table 85. Pass only file name and directory, not the content property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during inbound processing. Accepts either ASCII or binary.

Table 86. File transfer type property characteristics

Required	No
Default	`binary`
Property type	String
Globalized	no

Number of files to get at a time property (ftpGetQuantity)

Determines the number of files retrieved from the remote FTP URL with each remote poll.

Table 87. Number of files to get at a time property characteristic

Required	Yes
Default	`10`
Property type	Integer
Globalized	No

Number of poll periods between downloads property (ftpPollFrequency)

Determines how frequently the adapter polls the FTP server, measured in the number of standard poll cycles. For example, if PollPeriod is set to 10000, and FTPPollFrequency is set to 6, the adapter polls the LocalEventDirectory every 10 seconds and polls the remote EventDirectory every 60 seconds. The adapter performs FTP polling only if you specify a value for this property. If PollPeriod is 0, you consider it as 1 for calculation. If the calculation evaluates to 0, the adapter does not perform FTP polling.

Table 88. Number of poll periods between downloads property characteristics

Required	Yes
Default	5
Property type	Integer
Globalized	No

Retry limit for failed events (FailedEventRetryLimit)

This property specifies the number of times that the adapter attempts to redeliver an event before marking the event as failed.

Table 89. 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

Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)

Specifies the path of the script file that will be executed after downloading the files from the FTP server.

Table 90. Run FTP script file after downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)

Specifies the path of the script file that will be executed before downloading the files from the FTP server.

Table 91. Run FTP script file before downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Host name property (HostName)

Host name of the FTP Server to which the connection is established during inbound processing.

Table 92. Create Table property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Include business object delimiter in the file content property (IncludeEndBODelimiter)

When set to `true`, the delimiter is sent with the business object content for further processing. This property is valid only when splitting the event files based on a delimiter.

Table 93. Include business object delimiter in the file content property characteristics

Required	No
Default	`false`
Property type	String
Globalized	No

Local archive directory property (LocalArchiveDirectory)

Absolute path of the local Archive directory. The directory must be valid and exist.

Table 94. Local archive directory property characteristics

Required	No
Default	None
Property type	String

Table 94. Local archive directory property characteristics (continued)

Usage	You can use a WebSphere Application Server environment variable to represent the local archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${LOCALARCHIVE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **LocalArchiveDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Local directory property (LocalEventDirectory)

Local system directory into which the adapter downloads event files from the FTP site. You must specify a value for this property to enable the adapter to process events.

Table 95. Local directory property characteristics

Required	Yes
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the local event directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${LOCAL_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **LocalEventDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Maximum connections (MaximumConnections)

This property specifies the maximum number of connections that the adapter can use for inbound event delivery.

Table 96. Maximum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. The adapter considers any positive entry less than 1 to be equal to 1. Typing a negative value for this property may result in run time errors.
Globalized	No
Bidi supported	No

Minimum connections (MinimumConnections)

This property specifies the minimum number of connections that the adapter can use for inbound event delivery.

Table 97. Minimum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. Any value less than 1 is treated as 1 by the adapter. Typing a negative value or 1 for this property may result in run time errors.
Globalized	No
Bidi supported	No

File extension for local archive property (originalArchiveExt)

File extension used to archive the original event file. This preserves the entire event file for reference in case any of its business objects fail. This property is used only when LocalArchiveDirectory is valid and exists.

Table 98. File extension for local archive property characteristics

Required	No
Default	`original`
Property type	String
Globalized	Yes

Password property (Password)

Password of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the password is included in the URL specified in the EventDirectory property.

Table 99. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password used to connect to event data source property (EP_Password)

The password used during event persistence to get the database connection from the data source.

Table 100. Password used to connect to event data source property characteristics

Required	No
Default	None
Property type	String

Table 100. Password used to connect to event data source property characteristics (continued)

Globalized	Yes

Interval between polling periods (PollPeriod)

This property specifies the length of time that the adapter waits between polling periods.

Table 101. 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 102. 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
Bidi supported	No

Passphrase property (passPhrase)

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

Table 103. Passphrase property property characteristics

Required	No
Default	None
Property type	String
Usage	Used for enhanced security. It protects the private key by encrypting it in a SFTP configuration.

Table 103. Passphrase property property characteristics (continued)

Globalized	Yes
Bidi supported	No

Port number property (PortNumber)

Port number of the FTP server through which the connection is established during inbound processing.

Table 104. Port number property characteristics

Required	Yes
Default	`21` for FTP and FTPS in Explicit mode, `990` for FTPS in Implicit mode, and `22` for SFTP.
Property type	Integer
Globalized	No

Private key file property (PrivateKeyFilePath)

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

Table 105. Private key property characteristics

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

Protocol property (Protocol)

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

For example:

Normal connection: `FTP`

FTP over SSL connection: `FTPS_SSL`

FTP over TLS connection: `FTPS_TLS`

FTP over SSH connection: `SFTP`

Table 106. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String

Table 106. Protocol property characteristics (continued)

Globalized	No

Remote archive directory property (ftpArchiveDirectory)

Relative path of the archive directory on the FTP server. The directory must exist. There are several options for using this property to specify archiving:

- Specifying a value for this property, but no value for the FTPRenameExt property causes the adapter to append a timestamp to the event file name and move it to the FTP server archive directory specified in this property.
- Specifying a value for this property and the FTPRenameExt property causes the adapter to rename the processed event file name with a timestamp and the value specified in FTPRenameExt and moves it to the FTP server archive directory specified in this property.
- Specifying no value either for this property or the FTPRenameExt property causes the adapter to delete the processed event file without archiving it.
- Specifying no value for this property but specifying a value for the FTPRenameExt property causes the adapter to rename the processed event file, adding a timestamp and the value specified in FTPRenameExt.

The value of remote archive directory property accepts both the absolute and relative paths of the directory. If the value does not start with a forward slash, the adapter considers the path to be relative to your home directory.

Table 107. Remote archive directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the remote archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${REMOTEARCHIVE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. The archive directory located on the FTP server and used in inbound configuration represents the absolute path of the archive directory. It does not contain any host name or URL information. This directory is located on the same FTP server where the Event Directory is located, for example: /home/archive. **Note:** The **FTPArchiveDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Remote directory property (EventDirectory)

Remote directory of the FTP server from where the event files are retrieved for inbound processing. If the value of Remote directory is set to <HOME_DIR>, the adapter polls for event files in the user's home directory.

The value of event directory property accepts both the absolute and relative paths of the directory. If the value does not start with a forward slash, the adapter considers the path to be relative to the user's home directory.

Table 108. Remote directory property characteristics

Required	Yes
Default	<HOME_DIR>
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the remote directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${REMOTE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **EventDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Retrieve files with this pattern property (EventFileMask)

Filter for the event files. The file filter is a well-qualified expression consisting of alphanumeric characters and the `*` and `?` wild cards.

Table 109. Retrieve files with this pattern property characteristics

Required	Yes
Default	`*.*`
Property type	String
Globalized	Yes

Enable remote verification property (enableRemoteVerification)

When a client connects to the FTP server, two kinds of connections or channels are established; a command connection (also known as control connection), and a data connection. The command connection is the one through which the FTP commands are sent (and replies to these commands received) to the server and the data connection is the channel through which the data transfer takes place between the client and the server.

This property is used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

The verification is done while establishing a data connection to perform data transfer.

Note: This property is applicable only to FTP and FTPS protocols.

Table 110. Enable Remote verification property characteristics

Required	No
Possible values	`True` `False`

Table 110. Enable Remote verification property characteristics (continued)

Default	`True`
Property type	Boolean
Usage	This property verifies if the data connection and the control connection are from the same host system. By default, the remote verification property is set to TRUE by the FTP server. When this property is set to: • True, during run time, the adapter checks if the data connection is established with the same host as the control connection. If the data connection is established from a different host than the control connection, then an exception is thrown and the connection fails. • False, remote verification is not performed. **Note:** Disabling the remote verification leads to low security. Precaution must be taken before disabling the remote verification.
Globalized	No
Bidi supported	No

Retry EIS connection on startup (RetryConnectionOnStartup)

This property controls whether the adapter attempts to connect again to the FTP server if it cannot connect at startup.

Table 111. 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 FTP server 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 FTP server, for example, when you are building and testing the application that receives events from the adapter. If the adapter cannot connect, the adapter writes log and trace information and stops. The administrative console shows the application status as `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 establish a new connection.

Table 112. 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

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 113. Number of times to retry the system connection details

Required	No
Possible values	`0` and positive integers
Default	`0`
Property type	Integer
Usage	This property controls how many times the adapter retries the connection if the adapter cannot connect to the FTP server to perform inbound processing. A value of `0` indicates an infinite number of retries. To control whether the adapter retries if it cannot connect to the FTP server when it is first started, use the RetryConnectionOnStartup property.
Globalized	No
Bidi supported	No

Enable server verification property (EnableServerVerification)

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

Table 114. Enable server verification property details

Required	No
Possible values	`True` `False`
Default	False
Property type	Boolean

Table 114. Enable server verification property details (continued)

Usage	When this property is set to: • True, server authentication is enabled • False, server authentication is disabled The adapter checks for the HostKeyFile property in the path of the file that contains the host keys of the trusted servers.
Globalized	Yes
Bidi supported	No

Host key file property (HostKeyFile)

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

Table 115. Host key file property characteristics

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

Host name property (SocksProxyHost)

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

Table 116. Host name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SocksProxyPassword)

Password used to authenticate the proxy server.

Table 117. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Port number property (SocksProxyPort)

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

Table 118. Port number property characteristics

Required	No
Default	`1080`
Property type	Integer
Globalized	No

User name property (SocksProxyUserName)

User name used to authenticate the proxy server.

Table 119. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Sort event files property (SortEventFiles)

Determines the sorting order of event files being polled. Supported values are:

- by file name – sort ascending on file name
- by time stamp – sort ascending on last modified timestamp
- no sort – not sorted

Event file ordering from which events need to be delivered is valid only if the activation specification DeliveryType property is set to `ORDERED`. File name sorting is provided based on the locale of the FTP server. The ICU4J package is used to track the locales and their corresponding rules.

Table 120. Sort event files property characteristics

Required	No
Default	no sort (= not sorted)
Property type	String
Globalized	No

Specify criteria to split file content property (SplitCriteria)

This property takes different values based on the value of the SplittingFunctionClassName property. For example: To specify that a file is to be split every 5 KB, set the SplitCriteria property to `5000`.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.

- If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
- If the event file size is less than this value, the entire event file is posted. When SplitCriteria=0, chunking is disabled.

When FilePassByReference is enabled during inbound PassThrough, the event file is not split.

Note: For input files that contain multiple COBOL copybook records, in order to enable file splitting by size you must provide the correct length of each record. To determine the size of each record, use one of these methods:

1. Open the Business Object in a text editor.
   a. For example:

      ```
      <element name="CustomerNumber">
      <annotation>
      <appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
      <td:typeDescriptorElement>
      <td:initialValue kind="SPACE"/>
      <td:simpleInstanceTD accessor="readWrite" attributeInBit="false"
      contentSize="5" offset="0" size="5">
      <td:sharedType>
      <td:stringTD addrUnit="byte" alignment="byte" characterSize="1"
      lengthEncoding="fixedLength" paddingCharacter=" "
      prefixLength="0" width="5"/>
      </td:sharedType>
      </td:simpleInstanceTD>
      </td:typeDescriptorElement>
      </appinfo>
      </annotation>
      <simpleType>
         <restriction base="string">
          <maxLength value="5"/>
        </restriction>
       </simpleType>
      </element>
      ```

      Each element in the business object has a corresponding `<element>` entry.
   b. Look for a restriction tag for each element tag (the COBOL data binding requires a fixed-width data handler).
   c. Add up the `maxLength` attribute values for each of the elements. In this example, the value is 5. The sum of the `maxLength` values is the size of each record of type DFHCOMMAREA.
2. Open the Business Object in a text editor.
   a. Look for the complex type tag with the business object name value in the name attribute. In the example that follows, the business object name is DFHCOMMAREA.
   b. Locate a namespace-appended tag called `aggregateInstanceTD` and use the value for the attribute `contentSize`. In this example, the value is 117. This is the size of each record of type DFHCOMMAREA.

      ```
      <complexType name="DFHCOMMAREA">
      <annotation>
      <appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
      <td:typeDescriptorCT>
      <td:aggregateInstanceTD accessor="readWrite" attributeInBit="false"
      contentSize="117" offset="0" size="117">
      ```

Table 121. Specify criteria to split file content property characteristics

Required	No

Table 121. Specify criteria to split file content property characteristics (continued)

Default	0
Property type	String
Globalized	Yes

Splitting function class name property

This value takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The `com.ibm.j2ca.utils.filesplit.SplitByDelimiter` class that splits the event file based on delimiter.
- The `com.ibm.j2ca.utils.filesplit.SplitBySize` class that splits the event file based on the event file size.

Optionally, you can provide a custom file splitter class and use it by inputting the class name into the SplittingFunctionClassName property.

The delimiter or file size is provided in the SplitCriteria property. If the EventContentType property is set to null, it is automatically set to a class name that performs splitting based on file size.

Table 122. Splitting function class name property characteristics

Required	No
Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Globalized	No

Stop the adapter when an error is encountered while polling (StopPollingOnError)

This property specifies whether the adapter will stop polling for events when it encounters an error during polling.

Table 123. Stop the adapter when an error is encountered while polling details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If this property is set to `True`, the adapter stops polling when it encounters an error. If this property is set to `False`, the adapter logs an exception when it encounters an error during polling and continues polling.
Globalized	No
Bidi supported	No

Success file extension for local archive property (SuccessArchiveExt)

File extension used to archive all the successfully processed business objects. This property is used only when LocalArchiveDirectory is valid and exists. For example, `12345.order > 12345.order.success`

Table 124. Success file extension for local archive property characteristics

Required	No
Default	`success`
Property type	String
Globalized	Yes

User name property (UserName)

Name of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the user name is included in the URL specified in the EventDirectory property.

Table 125. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User name used to connect to event data source property (EP_UserName)

User name used by event persistence for getting the database connection from the data source.

Table 126. User name used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Rule editor to filter files

This property is used to filter event files based on a set of rules

Table 127. Rule editor to filter files

Required	Optional
Default	None
Property type	String
Usage	During an inbound processing, if the value in the rule table is specified, then the event files are fetched after filtering, based on the specified rules before polling those event files.
Globalized	Yes
Bidi supported	No

Resume transfer of files upon reconnection to FTP or FTPS server

WebSphere adapter for FTP supports resuming the transfer of files that were interrupted due to an error in connection to the FTP server. You can resume the transfer of files from the point at which it was interrupted during the previous transfer when the connection is reestablished. This feature is useful when downloading or uploading large files.

During a create operation, if the connection to the FTP server breaks, the FTPFileTransferInterruptedException is returned by the adapter. To resume transferring the file, a request must be resubmitted to the FTP adapter. Set the ResumeFailedTransfer property to `true` in the wrapper object, to indicate to the adapter to resume the transfer of the file. The adapter, upon reestablishing the connection to the FTP server, will resume the transfer of the file being created on the FTP server.

Note: The property, ResumeFailedTransfer is applicable only for outbound processing. Support for resuming transfer of a file is provided only for outbound Create operation.

Similarly, for an inbound operation, the adapter keeps track of the files downloaded partially and will resume downloading the file after the connection has been reestablished. The adapter saves the file with a ".partial" extension while downloading to the local event directory and renames the file to the original file after the file is completely retrieved to the local event directory.

The file for which the transfer was interrupted due to connection error should not be modified until the file is completely transferred to the FTP server. Also, the partially uploaded or downloaded file created by the adapter should not be modified until the file is transferred completely.

Note:

1. The FTP or FTPS server should provide support for the REST FTP command to resume the transfer of the file.
2. The feature, resuming of transfer of the file is not supported with SFTP protocol.

For more information, see the ResumeFailedTransfer property details in "Wrapper and interaction specification properties" on page 10.

Outbound data transformation

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

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

Data bindings

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

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

- DirectoryPath
- Filename
- TemporaryFilename
- DataConnectionMode
- FileTransferType
- DataProtectionLevel
- SecondServerDirectory
- SecondServerUsername
- SecondServerPassword
- IncludeEndBODelimiter
- ResumeFailedTransfer
- FileInLocalDirectory
- LocalDirectoryPath
- LocalArchivingEnabledForCreate
- LocalArchiveDirForCreate
- StagingDirectory
- GenerateUniqueFile
- SplittingFunctionClassName
- SplitCriteria
- DeleteOnRetrieve
- ArchiveDirectoryForRetrieve
- FileContentEncoding

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

Data handlers

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

- Delimited
- Fixed width
- XML

Inbound processing

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

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

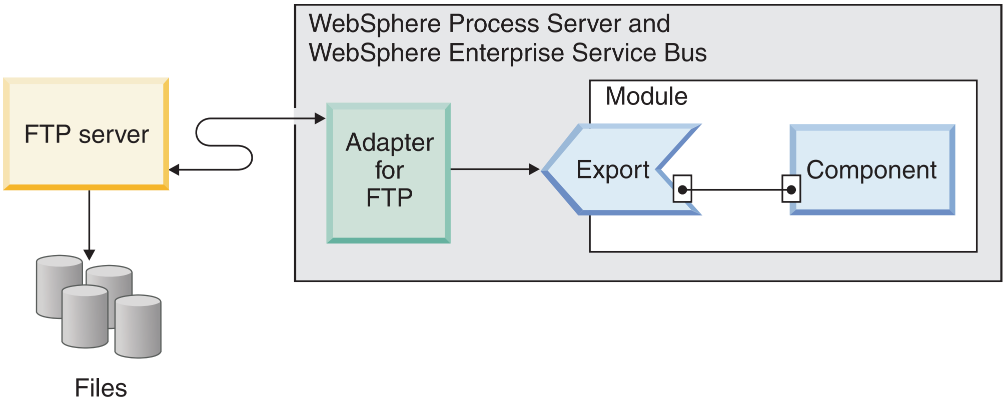

WebSphere Process Server and
WebSphere Enterprise Service Bus
Module
FTP server
Adapter for FTP
Export
Component
Files

Figure 13. Inbound processing flow

The adapter polls files from the event directory of the FTP server at regular intervals based on the FTPPollFrequency property. When a file arrives in the event directory, the adapter reads the entire file and downloads the file to a local event directory on the adapter workstation. The adapter downloads the files from the FTP server sequentially, that is one file at a time, and cannot download all the files simultaneously. After the file is downloaded, the adapter either archives the file in the FTP server in an archive directory given by the FTPArchiveDirectory property or deletes it based on your configuration. The event directory, archive directory, poll frequency, and poll quantity (the number of files to poll in a single poll cycle) are all configurable properties.

Note: If the Remote directory is set to <HOME_DIR>, the adapter polls for event files in `user's` home directory.

Note: The value of an event directory property accepts both the absolute and relative paths of the directory. If the value does not begin with a forward slash (/), the adapter considers the path to be relative to the home directory of the user.

For example, if the value in the remote directory property is set to “ftpuser/event”, the adapter considers this to be the path relative to the home directory. If the home directory is set to “/usr/ftp", then the adapter will poll the directory “/usr/ftp/ftpuser/event” for event files.

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

Inbound event processing consists of the following steps:

1. FTP server generates events in the form of files.

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

Processing of files using FTP scripts

In addition to processing the files downloaded from the event directory during polling, WebSphere Adapter for FTP can also be used to process the files downloaded using the FTP scripts.

You can specify the scripts to be run before or after polling the event directory using the properties, “Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)” on page 67 and “Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)” on page 66. The script files can contain FTP commands, such as `mget` and `get`, to download the files from the remote directories on the FTP server to the local event directory of the machine where the adapter is installed. The WebSphere Adapter for FTP processes the files that are downloaded to the local event directory configured in the activation specification properties and delivers the processed business objects to the consuming service.

Following is an example of a script:

```
lcd C:\FTPAdapter\localevent
cd /ftpDir1
mget *.txt
cd /ftpDir2
get abc.xml
```

Where, `C:\FTPAdapter\localevent` is the local event directory of WebSphere Adapter for FTP, and ftpDir1 and ftpDir2 are directories that exist on the FTP server. The adapter executes the script and downloads the files to the local event directory. The adapter then processes the files and delivers it to the consuming service.

Note:

1. You must place the files that have been downloaded using the script to the configured local event directory for the adapter to process it. Use the FTP command `lcd` to change the local working directory to the localEventDirectory before you download any files using the script.

2. The files downloaded to the local event directory using the commands, `mget` or `get` will be deleted from the FTP server by the FTP adapter after you download the files. This is to ensure that the files are not downloaded again during the next poll cycle.
3. Use the script file to download the files only from remote directories and not from the event directory of the FTP adapter.

Supported inbound operation

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

Event file locking

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

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

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

Rule-Based filtering of events

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

Though using the rule is optional and specifying an event file mask is mandatory, the rule takes a higher precedence over the event file mask, when both a rule and an event file mask are specified. Event file mask is effective only when there is no rule specified. By default, an event file mask has "*.*" as the 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 128. 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
LastModified	Greater than, Less than, Greater than or equal to, Less than or equal to, Equal to, Not equal to. **Note:** Select 'Equal to' operator when you choose the days of 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

Wrapper and interaction specification properties

Wrapper properties are attributes of the wrapper business object that enable an application programmer to control an operation for the business objects in a wrapper. Interaction specification properties control the interaction for an operation for the entire adapter.

The external service wizard sets the interaction specification properties when you configure the adapter. You can change some, but not all, of these properties. However, you can change some properties for outbound operations. Use the assembly editor to change these properties, which reside in the method binding of the import. You set the wrapper properties using the WebSphere Integration Developer test client or programmatically at run time.

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

Table 129. Interaction specification properties

Property name		Description
In the wizard	In the wrapper business object	
Remote archive directory for retrieve operation	ArchiveDirectoryForRetrieve	The adapter optionally archives the file to this folder before it is deleted during a Retrieve operation.
Create new file if the file does not exist	CreateFileIfNotExists	If the file does not exist on the FTP server, the adapter creates the file when this property is set to `true` during Append and Overwrite operations.
FTP server connection mode	DataConnectionMode	Data connection mode used by the FTP server during file transfers.
Delete the file after retrieve operation	DeleteOnRetrieve	The adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.
Remote directory on FTP system	DirectoryPath	Absolute path of the directory on the FTP server where the outbound operation must be performed.

Table 129. Interaction specification properties (continued)

"Data channel protection level (dataProtectionLevel)" on page 88	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
File content encoding	FileContentEncoding	Encoding used while writing to the file.
File in local directory	FileInLocalDirectory	If set to `true` during a create operation, the file content is picked from the local directory path of the adapter workstation.
Default target file name	Filename	Name of the file in the directory provided by the DirectoryPath property.
File transfer type	FileTransferType	File transfer type used during outbound operations.
Generate a unique file	GenerateUniqueFile	The adapter creates a unique file name when this property is set to `true`.
Host name property	SecondServerHostName	Host name of the second FTP server.
Delimiter between business objects in the file property	IncludeEndBODelimiter	File content is appended with this value.
Local archive directory for create operation	LocalArchiveDirForCreate	When `LocalArchivingEnabledForCreate` is set to `true` during a create operation, the file is saved to the local workstation in this directory.
Archive file in the local directory for create operation	LocalArchivingEnabledForCreate	When set to `true`, the file is saved to the local workstation during a create operation.
Local directory	LocalDirectoryPath	The file is picked from this directory.
(Not available)	ResumeFailedTransfer	When this property is set to `true` during a create operation, the adapter resumes the transfer of files from the point at which the transfer of file was interrupted due to connection error.
Port number	SecondServerPortNumber	Port number of the second FTP server.
Protocol	SecondServerProtocol	Specifies the protocol used to connect to the second server.
Script File Parameters	ScriptFileParameters	The parameters required by the FTP script file.
Directory	SecondServerDirectory	Directory path of the second FTP server during a ServerToServerFileTransfer operation.
Password	SecondServerPassword	Password of the second FTP server during a ServerToServerFileTransfer operation.
User name	SecondServerUsername	User name of the second FTP server during a ServerToServerFileTransfer operation.
Specify criteria to split file content	SplitCriteria	The delimiter that separates the business objects in the event file.
Split function class name	SplittingFunctionClassName	The fully qualified class name of the class file to be used to enable file splitting.
Staging directory	StagingDirectory	The file is first created into this directory.
Temporary file name	TemporaryFilename	Specifies the temporary file name for the create operation.

Archive file in the local directory for create operation property (LocalArchivingEnabledForCreate)

During outbound create operations, when the file content is coming as part of the business object from a J2EE application and this property is set to `true`, the file is saved to the local workstation in the LocalArchiveDirForCreate directory before performing the outbound operation.

Table 130. Archive file in the local directory for create operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Create new file if the file does not exist property (CreateFileIfNotExists)

During outbound Append and Overwrite operations, if the file does not exist on the FTP server, the adapter creates the file when this property is set to `true`. If this property is `false` and file does not exist, the adapter sends an error.

Table 131. Create new file if the file does not exist property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the "PBSZ 0" command before issuing the PROT command.

Table 132. Data channel protection level property characteristics

Required	No

Possible values	`Private - Data is transferred in encrypted form`
	`Clear - Data is transferred as clear text`

Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.

Table 132. Data channel protection level property characteristics (continued)

Globalized	No
Bidi supported	No

Delete the file after retrieve operation (DeleteOnRetrieve)

During an outbound Retrieve operation, the adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.

Table 133. Delete the file after retrieve operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Default target file name property (Filename)

Name of the file to be used during outbound operations.

Table 134. Default target file name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Delimiter between business objects in the file property (IncludeEndBODelimiter)

File content is appended with this value. Used during the outbound create, append, and overwrite operations.

Table 135. Include business object delimiter in the file content property characteristics

Required	No
Default	For the create and overwrite operations, no default value is set. For the append operation, the default value is `<EndBO>`. For the append operation, the following rules apply: • If the delimiter is set to `null` in the business object wrapper, no delimiter is used to separate the business objects. • If the IncludeEndBODelimiter property is not set in the business object wrapper, and the value in the interaction specification is also `null`, the default is `<EndBO>`. • If a specific delimiter value is specified in the business object wrapper, the specified value will be appended. • If both the business object wrapper and the interaction specification have specified values, the business object wrapper value takes precedence.
Property type	String
Globalized	Yes

Directory property (SecondServerDirectory)

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

Table 136. Directory property characteristics

Required	No
Default	None
Property type	String
Usage	For interaction specification properties, the directory located on the FTP server and used in outbound operation represents the absolute path of the FTP directory. For example: `/home/usr/output`. It does not contain any host name or URL information. For wrapper business object properties, the URL of the second server to which the ServerToServerFileTransfer outbound operation is performed. For example: The syntax for specifying the FTP URL is: `ftp://[UserId:password@]FTPserver[:port]/DirectoryForSecondServer`.
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used while writing to the file. If this property is not specified, the adapter tries to read without using any specific encoding. You can specify any Java supported encoding set.

Table 137. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File in local directory property (FileInLocalDirectory)

During outbound create operations, if this property is set to `true`, the file content is not available in the business object. The file is retrieved from the local directory on the adapter workstation. During outbound retrieve operations, if this property is set to `true`, the file content is not sent to the J2EE application as part of the business object. The file is saved to the local directory of the adapter workstation.

Table 138. File in local directory property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during outbound operations. Takes either ASCII or binary.

Table 139. File transfer type property characteristics

Required	No
Default	`binary`
Property type	String
Globalized	No

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Takes either `active` or `passive`. This value is used only when a file transfer is taking place. This property is not used when performing a server to server file transfer outbound operation.

Table 140. FTP server connection mode property characteristics

Required	No
Default	`active`
Property type	String
Possible values	`active` or `passive`
Globalized	No

Generate a unique file (GenerateUniqueFile)

During outbound Create operation, the adapter creates a unique file name when this property is `true`. The adapter ignores any value that is set for the Filename property when this property is set to `true`.

Note: The adapter does not support both GenerateUniqueFile and StagingDirectory options at the same time.

Table 141. Generate unique file property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No
Restrictions	The FTP server must support RFC1123 to use this feature.

Host name property (SecondServerHostName)

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

Table 142. Host name property characteristics

Required	Yes
Default	None
Property type	String

Table 142. Host name property characteristics (continued)

Globalized	Yes

Local archive directory for create operation property (LocalArchiveDirForCreate)

During outbound create operations, when the file content is coming as part of the business object and LocalArchivingEnabledForCreate is set to `true`, the file is saved to the local workstation in this directory.

Table 143. Local archive directory for create property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalArchiveDirForCreate directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Local directory property (LocalDirectoryPath)

During outbound create operations, when FileInLocalDirectory property is set to `true`, the file content is not available in the business object. Instead the file is picked from this directory. During outbound retrieve operations, when FileInLocalDirectory property is set to `true`, the file content is not sent as part of business object. The file is saved to this directory.

Table 144. Local directory property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalDirectoryPath directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Port number property (SecondServerPortNumber)

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

Table 145. Port number property characteristics

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

Protocol property (SecondServerProtocol)

Protocol that is used to establish a connection to the second server. The FTP protocol is used in establishing the connection.

Table 146. Protocol property characteristics

Required	Yes
Default	FTP
Property type	String
Globalized	No

Password property (SecondServerPassword)

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

Table 147. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Remote archive directory for retrieve operation property (ArchiveDirectory ForRetrieve)

During an outbound Retrieve operation, the adapter optionally archives the file to this folder before it is deleted. The archive directory must exist.

Table 148. Remote archive directory for retrieve operation property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Remote directory on FTP system property (DirectoryPath)

Absolute path of the directory on the FTP server where the outbound operation must be performed for all operations except ExecuteFTPScript, or the directory path on the local adapter workstation for the ExecuteFTPScript operation only. The directory must exist.

Note: If the value <HOME_DIR> is specified as the DirectoryPath, the outbound operations will be performed in the user's home directory.

Table 149. Remote directory on FTP system property characteristics

Required	No
Default	None
Property type	String

Table 149. Remote directory on FTP system property characteristics (continued)

Usage	The DirectoryPath directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

ResumeFailedTransfer

This property supports resuming the transfer of files, which were interrupted due to an error in connection to the FTP server.

Note: This property is applicable only to outbound processing.

Table 150. ResumeFailedTransfer property characteristics

Required	No
Default	`false`
Property type	Boolean
Usage	During a create operation, when this property is set to `true`, the adapter resumes the transfer of files from the point at which the transfer of file was interrupted due to an error in connection.
Globalized	No

Script File Parameters property (ScriptFileParameters)

During an outbound ExecuteFTPScript operation, the parameters required by the FTP script file are set in this property. During run time, the adapter replaces the parameters with these values.

Table 151. Script File Parameters property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Specify criteria to split file content property (SplitCriteria)

This property accepts different values based on the value of the SplittingFunctionClassName property.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
  - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
  - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=`0`, chunking is disabled.

Table 152. Specify criteria to split file content property characteristics

Required	No
Default	`0`
Property type	String
Globalized	Yes

Split function class name property (SplittingFunction ClassName)

Takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The com.ibm.j2ca.utils.filesplit.SplitByDelimiter class that splits the event file based on delimiter.
- The com.ibm.j2ca.utils.filesplit.SplitBySize class that splits the event file based on the event file size.

The delimiter or file size is provided in the SplitCriteria property.

Table 153. Split function class name property characteristics

Required	No
Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Globalized	No

Staging directory property (StagingDirectory)

During outbound create operations, the file will be created in this directory first. When the file creation is complete, the file is copied to the directory specified in the DirectoryPath property. This staging directory is also used for Append and Overwrite operations where the specified file is copied to the StagingDirectory, if present. The appended or overwritten content is then moved back to the original specified directory. If StagingDirectory is not specified, the operation is run in the actual required directory.

Note: The adapter does not support both StagingDirectory and GenerateUniqueFile options at the same time.

Table 154. Staging directory property characteristics

Required	No
Default	None
Property type	String
Usage	The StagingDirectory directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Temporary file name property (TemporaryFilename)

This property specifies the temporary file name for the create operation. After successful creation of the file, the file gets renamed to the value specified in the 'Default target file name' property.

Table 155. Temporary file name property characteristics

Required	No
Possible values	All valid file names
Default	None
Property type	String
Usage	This property is used in the create operation. If the temporary file name is specified, the file is created with the temporary file name. After the file is successfully created, the file is renamed to the value that is specified in the 'Default target file name' property.
Example	xyz.tmp
Globalized	No

User name property (SecondServerUsername)

User name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 156. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Troubleshooting and support

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

Related concepts

"Resume file transfer"

Resume file transfer

Problem:

If the connection to the FTP server breaks during the transfer of a file, the transfer can be resumed from the point at which it was interrupted. In case of network-related issues when a file transfer is in progress for an outbound create operation, it is observed that some of the FTP servers retain reference to the connection at the server end and does not close the reference when the connection is broken. This causes an error when the outbound request is resent to resume the transfer of the file. The FTP server returns a reply code "550 Can't access file" when the outbound create request is resent. This is due to the file lock in the target file maintained by the connection reference that is created at the FTP server during the previously failed file transfer request.

Resolution:

The invalid connection handle must be cleared from the FTP server manually for the outbound request to resume transfer of the file. If 'connection timeout' or 'No-transfer timeout' related properties are set at the FTP server, the invalid connection handle will be cleared automatically upon exceeding the timeout interval and any subsequent outbound request to resume the file transfer will be successful.

Related tasks

“Troubleshooting and support” on page 96
Common troubleshooting techniques and self-help information help you identify and solve problems quickly.

Authentication to FTP server by passing connection parameters dynamically

WebSphere® Adapter for FTP uses connection properties either through Managed Connection Factory properties or Java Authentication and Authorization Services (JAAS) alias. You can change the connection properties through the WebSphere Process Server admin console and restart the J2EE application or change the JAAS security settings.

The ConnectionSpec properties are used by an application component to pass connection request-specific properties.

Authentication using connection specification properties

WebSphere Adapter for FTP uses connection properties either through Managed Connection Factory properties or Java Authentication and Authorization Services (JAAS) alias. If you want to change the connection properties used for authentication with either one of these authentication methods, you can change the connection properties through the WebSphere Process Server admin console and restart the J2EE application or change the JAAS security settings.

In addition to the above methods, the connection parameters can also be specified through the ConnectionSpec properties. The ConnectionSpec properties are used by an application component to pass connection-related properties.

Based on the protocol used in the Managed Connection Factory, you can specify the relevant ConnectionSpec properties for the outbound request. If you specify both ConnectionSpec properties and Managed Connection Factory properties, during runtime, the adapter uses the values specified in the ConnectionSpec properties to create a connection and ignores the values in the Managed Connection Factory properties.

The ConnectionSpec properties include:

For FTP:

- userName
- password

For FTPS:

- userName
- password
- trustStorePath
- trustStorePassword

- keyStorePath
- keyStorePassword
- keyPassword
- keyStoreType

For SFTP:

- userName
- password
- privateKeyFilePath
- passphrase
- hostKeyFile

To configure the adapter to create a connection to the FTP server by passing the connection parameters dynamically, see “Passing the connection parameters dynamically during outbound processing.”

Related tasks

“Passing the connection parameters dynamically during outbound processing”
To pass the connection-related properties dynamically as part of the outbound request you must configure the connection specification class name and set the connection properties on the business graph.

Creating a new interface
After passing and configuring the connection parameters, during the outbound processing, create an application component to send the outbound request along with the connection properties to test the functionality.

Creating a Java component
After creating a new interface and testing it, create a Java component to set the values for the properties element.

Passing the connection parameters dynamically during outbound processing

To pass the connection-related properties dynamically as part of the outbound request you must configure the connection specification class name and set the connection properties on the business graph.

Before you begin

1. The FTP adapter import interface, for example, FTPImport, must be created for the required outbound operations by executing the external service wizard.
2. The input data type for each of the outbound operation must be configured to use the business graph of the business object. For example, the input data type of the operations can be FTPFileBG or CustomerWrapperBG.

   The business graph implementation has a child business object, ‘properties' defined as an element in the business graph schema definition. The connection properties should be set in the dataobject ‘properties' of the business graph.

About this task

To pass the connection-related properties dynamically as part of the outbound request, follow this procedure.

Procedure

1. Configure the ConnectionSpec class name in the FTP Import created.

   a. Right-click the FTP adapter import in the assembly diagram and select **Show in → Properties view**.
   b. In the Properties tab, select **Binding → End-point Configuration**.
   c. In the Connection Spec properties tab, select ConnectionSpec class name as `com.ibm.j2ca.ftp.FTPFileConnectionSpec`
2. Set the **Resource authentication** field in Security Attributes to `Application`.
   a. Select **Security Attributes** from Binding properties.
   b. Set the **Resource authentication** property to Application from Advanced properties. The default value is Container.

   When the Resource Authentication property is set to Application, the J2EE component executes a programmatic sign-on to the FTP server. The application component passes security information, such as user name and password, through the ConnectionSpec instance.
3. Set the **Connection properties** in the BusinessGraph within the properties child business object.

   For the adapter to accept the connection parameters dynamically during an outbound request, the application component should set the connection parameters on the business graph data object of the business object.

   The connection properties set on the business graph is prefixed as "CS" to identify them as ConnectionSpec properties. For example, you can set the user name and password to 'CSuserName' and 'CSpassword' in the properties element of the BusinessGraph to set the values of connection properties.

   Note: The host name, protocol, or port number values are not accepted through the ConnectionSpec properties. The adapter accepts only authentication-related properties of the user, such as user name, password, and truststore, to be passed dynamically during an outbound request.

Results

The connection parameters are configured.

What to do next

Create a new interface and a Java component, and then deploy the application onto the WebSphere Process Server.

Related concepts

"Authentication using connection specification properties" on page 97
WebSphere Adapter for FTP uses connection properties either through Managed Connection Factory properties or Java Authentication and Authorization Services (JAAS) alias. If you want to change the connection properties used for authentication with either one of these authentication methods, you can change the connection properties through the WebSphere Process Server admin console and restart the J2EE application or change the JAAS security settings.

Creating a new interface:

After passing and configuring the connection parameters, during the outbound processing, create an application component to send the outbound request along with the connection properties to test the functionality.

Before you begin

You have run the external service wizard to create the outbound interface. The new FTPImport interface has multiple input properties to pass the connection properties.

About this task

You create an application component to send the outbound request using the connection properties. The new FTPImport interface has multiple input properties to pass the connection properties. To create a new interface to test the functionality, use the following procedure.

Procedure

1. From the Business Integration view, click **File** → **New** → **Interface**. The New Interface Wizard is displayed.
2. Type a name, for example, FTPDynamicConnectionInterface, for the new interface, and click **Finish**.
3. Add a "Request Response" operation. It matches the operation in the FTPOutboundInterface with additional input parameters for the connection properties. The input parameters of the outbound operation contains the BusinessGraph object and a set of connection properties for which the value is set in the BusinessGraph.

Results

A new interface is created.

What to do next

Create a Java component. For more information, see "Creating a Java component"

Related concepts

"Authentication using connection specification properties" on page 97
WebSphere Adapter for FTP uses connection properties either through Managed Connection Factory properties or Java Authentication and Authorization Services (JAAS) alias. If you want to change the connection properties used for authentication with either one of these authentication methods, you can change the connection properties through the WebSphere Process Server admin console and restart the J2EE application or change the JAAS security settings.

Creating a Java component:

After creating a new interface and testing it, create a Java component to set the values for the properties element.

Before you begin

Make sure that you have created a new interface that has multiple input properties to pass the connection properties.

About this task

You must create a Java component and set the connection-related properties to pass it as input to the interface on the business graph object. To create a Java component, use the following procedure.

Procedure

1. Create a Java component in the assembly diagram.
2. Wire the Java component to the FTPOutboundInterface import. The Java component interface, that is, FTPDynamicConnectionInterface is created. To create the Java component, click the **Java component**. Click the 'add an interface' icon and select the interface, FTPDynamicConnectionInterface.
3. Set the connection-related properties, which are sent as input to the interface on the BusinessGraph object, for the implementation of the Java component.

   The sample code below is the J2EE component implementation that sets the connection parameters on the properties business object of the BusinessGraph:

```
public DataObject createFTPFile(DataObject createFTPFileWrapperBG,String userName,
   String password, String privateKeyFilePath,
   String passphrase) {

 DataObject prop = createFTPFileWrapperBG.getDataObject("properties");

 // check if they already created this child object or not
 if(prop == null) {
  // Create the "properties" business object
  prop = createFTPFileWrapperBG.createDataObject("properties");
 }

 // Setting the property 'userName' to connectionSpec
 // Note that the username property is prefixed by CS
 prop.setString("CSuserName", userName);

 // Setting the property 'password' to connectionSpec
 // Note that the password property is prefixed by CS
 prop.setString("CSpassword", password);

 //Setting the property 'privateKeyFilePath' which is used for SFTP protocol          to conne
 prop.setString("CSprivateKeyFilePath", privateKeyFilePath);

 //Setting the property 'passphrase' which is used for SFTP protocol to         connection spe
 prop.setString("CSpassphrase", passphrase);


 // invoke the Adapter
 Service serv= locateService_SFTPImportPartner();
 Object boReturn= serv.invoke(
   "create",
   createFTPFileWrapperBG);

 // return the result BO that we got back from the FTP Adapter
 return ((DataObject)boReturn).get(0);

  }
```

   During runtime, the connection properties values are set on the input parameters of the Java component. This will in turn be set on the dataobject 'properties' of the BusinessGraph by the above displayed code. The EIS binding will then pass the connection properties to the adapter which is set on the dataobject 'properties' by populating it in the ConnectionSpec bean. The adapter uses the ConnectionSpec properties to get a connection to the EIS.

   For more information about EIS binding, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.websphere.wesb.doc/doc/cadm_dynamicheader.html. For information about how to configure dynamic authentication, see http://www.ibm.com/developerworks/websphere/library/techarticles/0608_martinez/0608_martinez.html.

Results

A Java component is created.

What to do next

Deploy the application onto the WebSphere Process Server and send an outbound request, which includes the connection parameters using the WebSphere Integration Developer test client. As a result, the adapter uses these connection parameters from the request to create the connection. Any value specified in the Managed Connection Factory properties is ignored by the adapter.

Related concepts

"Authentication using connection specification properties" on page 97
WebSphere Adapter for FTP uses connection properties either through Managed Connection Factory properties or Java Authentication and Authorization Services (JAAS) alias. If you want to change the connection properties used for authentication with either one of these authentication methods, you can change the connection properties through the WebSphere Process Server admin console and restart the J2EE application or change the JAAS security settings.

Support for polling unchanged files

You can retrieve files using the property, Time interval for polling unchanged files, during inbound processing. The property specifies the time interval for which the adapter needs to monitor the files for any updates in the content before polling. The adapter polls those files that are not changed during the specified time interval.

File retrieval

During inbound processing, you can manage the retrieval of the files by using the `Time interval for polling unchanged files` property. This property helps you to retrieve only those files which are not changed during the specified time interval. If the time difference between last modified timestamp to the current system time is greater than the value set in FileUnchangedTimeInterval, then such files will be polled.

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 that have not undergone any change during the time interval. Although, the adapter also polls the files that are currently being edited but any unsaved content is not processed during the event processing. This configuration prevents occurrence of any erroneous results.

When the adapter polls the event directory, it uses this property to check if a file has been modified during the specified time interval. The adapter uses the lastModifiedtimestamp value 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.

Setting deployment and runtime properties

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

Before you begin

Before you can set the properties in this section, you must create your adapter module. It should be displayed in WebSphere Integration Developer below the adapter project.

About this task

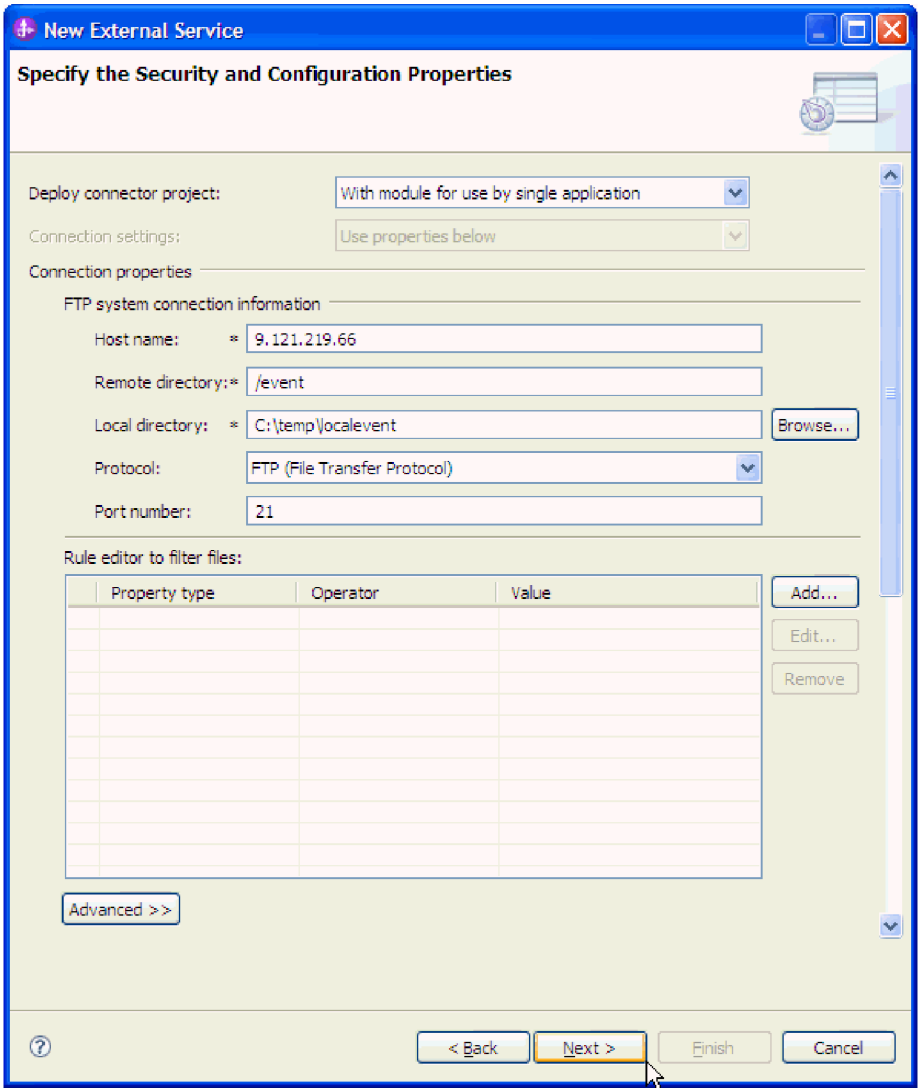
To set deployment and runtime properties, follow this procedure. For more information about the properties in this topic, refer to “Activation specification properties” on page 53 topic.

Procedure

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

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

     If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.
3. Define the following FTP system connection information for your module. For more information, refer to “Activation specification properties” on page 53 topic.
   - **Host name** - Specifies the host name of the FTP server.
   - **Remote directory** - Specifies the directory on the FTP server, where the adapter polls and picks up files. If the Remote directory is set to <HOME_DIR>, the adapter polls for event files in the home directory.
   - **Local directory** - Specifies the directory on the adapter workstation where the event files are downloaded from the FTP server.
   - **Protocol** - Specifies the protocol used to connect to the FTP server. Following are the protocols that can be specified:
     - FTP - File Transfer Protocol
     - FTP over SSL - File Transfer Protocol over Secure Socket Layer
     - FTP over TLS - File Transfer Protocol over Transport Layer Security
     - SFTP - Secure shell File Transfer Protocol
   - **Port number** - Specifies the port number of the FTP server.



Deploy connector project: With module for use by single application

Connection settings: Use properties below

Connection properties

FTP system connection information

Host name: * 9.121.219.66

Remote directory:* /event

Local directory: * C:\temp\localevent

Protocol: FTP (File Transfer Protocol)

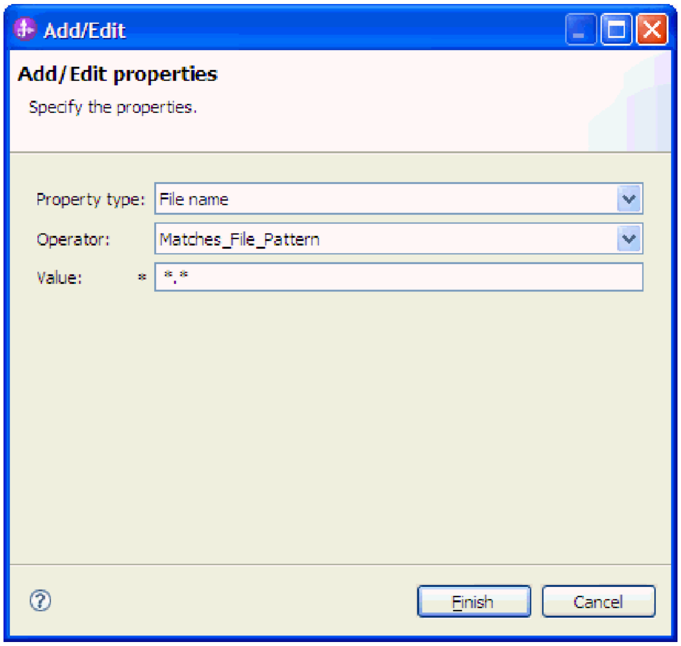
Port number: 21

Rule editor to filter files:

Property type	Operator	Value

Figure 14. Specify the Security and Configuration Properties window

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



Add/Edit properties

Specify the properties.

Property type: File name

Operator: Matches_File_Pattern

Value: * *.*

Figure 15. Adding or editing a rule

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

b. Select the operator for the property type from the **Operator** list. Each of the property type metadata has its own operators.
   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) 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" on page 80.

5. Optional: Specify advanced properties by clicking **Advanced**. Expand each of the advanced sections to review the properties.
   - Event polling configuration
   - Event delivery configuration
   - Event persistence configuration
   - Additional configuration
   - FTP archiving configuration
   - Socks proxy server connection information
   - Secure configuration
   - Bidi properties
   - Logging and tracing properties

   The following sections describe the options that are available in the advanced property groups.
   - **Event polling configuration**
     a. In the **Interval between polling periods** field, specify the number of milliseconds that the adapter should wait between polling periods. For more information, see "Interval between polling periods (PollPeriod)" on page 70.
     b. In the **Maximum events in polling period** field, specify the number of events that the adapter should deliver in each polling period. For more information, see "Maximum events in polling period (PollQuantity)" on page 70.
     c. In the **Retry interval if connection fails** field, specify the number of milliseconds for the adapter to wait before trying to connect after a connection failure during polling. For more information, see "Retry interval if connection fails (RetryInterval)" on page 75.
     d. In the **Number of times to retry the system connection** field, specify the number of times to retry the connection before reporting a polling error. For more information, see "Number of times to retry the system connection (RetryLimit)" on page 75.
     e. If you want the adapter to stop if polling errors occur, select **Stop the adapter when an error is encountered while polling**. If you do not select this option, the adapter logs an exception but continues to run. For more information, see "Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 79.
     f. Select **Retry EIS connection on startup** if you want the adapter to retry a failed connection when starting. For more information, see "Retry EIS connection on startup (RetryConnectionOnStartup)" on page 74.

g. In the **Time interval for polling unchanged files** field, specify the time interval for which the adapter needs to monitor the files for any updates in the content before polling. The adapter polls those files that are not changed during the specified time interval. For more information, see "Time interval for polling unchanged files (fileUnchangedTimeInterval)" on page 64.

- **Event delivery configuration**
  a. In the **Type of delivery** field, select the delivery method. The methods are described in "Delivery type (DeliveryType)" on page 59.
  b. If you want to ensure that events are delivered only once and to only one export, select **Ensure once-only delivery**. This option might reduce performance but does not result in duplicate or missing an event delivery. For more information, see "Ensure once-only event delivery (AssuredOnceDelivery)" on page 57.
  c. In the **Retry limit for failed events** field, specify the number of times that the adapter will attempt to redeliver an event before marking it as failed. For more information, see "Retry limit for failed events (FailedEventRetryLimit)" on page 66.
- **Event persistence configuration**
  a. Select **Auto create event table** if you want the adapter to create the Event Persistence table. For more information, see "Auto create event table property (EP_CreateTable)" on page 57.
  b. In the **Event recovery table name** field, specify the name of the table that the adapter uses for event persistence. For more information, see "Event recovery table name property (EP_EventTableName)" on page 60
  c. In the **Event recovery data source (JNDI) name** field, specify the JNDI name of the data source that event persistence will use to connect to the JDBC database. For more information, see "Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)" on page 60
  d. In the **User name used to connect to event data source** field, specify the user name that the event persistence will use to connect to the database from the data source. For more information, see "User name used to connect to event data source property (EP_UserName)" on page 80
  e. In the **Password used to connect to event data source** field, specify the password that the event persistence will use to connect to the database from the data source. For more information, see "Password used to connect to event data source property (EP_Password)" on page 69.
  f. In the **Database schema name** field, specify the schema name of the database that the event persistence will use. For more information, see "Database schema name property (EP_SchemaName)" on page 59.
- **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 this pattern property (EventFileMask)" on page 73.
  b. In the **Sort event files** field, specify the sorting order of the event files being polled. For more information, see "Sort event files property (SortEventFiles)" on page 77.
  c. Select the **Enable remote verification** check box to enable remote verification. This property checks if the control and data connections are established with the same host (typically, the machine from which you establish a connection to the FTP server). The connection fails if the

control and data connections are not established. By default, the **Enable remote verification** check box is selected.

Note: This property is applicable only to FTP and FTPS protocols.
For more information, see "Enable remote verification property (enableRemoteVerification)" on page 73

d. In the **Encoding used by FTP server** field, specify the encoding of the FTP server. For more information, see "Encoding used by FTP server property (EISEncoding)" on page 60.

e. In the **File content encoding** field, specify the encoding used to read the event files. For more information, see "File content encoding property (FileContentEncoding)" on page 62.

f. In the **FTP server connection mode** field, specify the data connection mode used by the FTP server during file transfers. For more information, see "FTP server connection mode property (DataConnectionMode)" on page 61.

g. In the **File transfer type** field, specify the file transfer type used during inbound processing. For more information, see "File transfer type property (FileTransferType)" on page 65.

h. In the **Number of files to get at a time** field, specify the number of files retrieved from the remote FTP URL. For more information, see "Number of files to get at a time property (ftpGetQuantity)" on page 65.

i. In the **Number of poll periods between downloads** field, specify how frequently the adapter polls the FTP server. For more information, see "Number of poll periods between downloads property (ftpPollFrequency)" on page 65.

j. In the **Custom parser class name** field, specify the fully qualified class name of the custom parser that is used to parse the ls output. For more information, see "Custom parser class name property (CustomParserClassName)" on page 58.

k. Select **Pass only file name and directory, not the content** to specify that the file content of the event file is not sent to the export. For more information, see "Pass only file name and directory, not the content property (FilePassByReference)" on page 65.

l. Select **Include business object delimiter in the file content** to specify that the delimiter will be sent with the business object content for further processing. For more information, see "Include business object delimiter in the file content property (IncludeEndBODelimiter)" on page 67

m. Select **Split file content based on the size (bytes) or delimiter** to use the size in bytes or the delimiter to split the file content. For more information, see "Splitting function class name property" on page 79

n. In the **Specify criteria to split file content** field, specify that different values will be taken, based on the value of the SplittingFunctionClassName property. For more information, see "Specify criteria to split file content property (SplitCriteria)" on page 77.

o. In the **Split function class name** field, specify the fully qualified class name of the class file to be used to enable file splitting. For more information, see "Splitting function class name property" on page 79.

p. In the **Run FTP script file before downloading files** field, specifies the path of the script file that will be executed before downloading the files from the FTP server. For more information, see "Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)" on page 67.

q. In the **Run FTP script file after downloading files** field, specifies the path of the script file that will be executed after downloading the files from the FTP server. For more information, see "Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)" on page 66.

- **FTP archiving configuration**
  a. In the **Local archive directory** field, specify the absolute path of the local Archive directory. For more information, see "Local archive directory property (LocalArchiveDirectory)" on page 67.
  b. In the **File extension for local archive** field, specify the file extension used to archive the original event file. For more information, see "File extension for local archive property (originalArchiveExt)" on page 69.
  c. In the **Success file extension for local archive** field, specify the file extension used to archive all the successfully processed business objects. For more information, see "Success file extension for local archive property (SuccessArchiveExt)" on page 80.
  d. In the **Failure file extension for local archive** field, specify the file extension used to archive business objects in the event file that are not successfully processed. For more information, see "Failure file extension for local archive property (FailedArchiveExt)" on page 61.
  e. In the **Remote archive directory** field, specify the directory. For more information, see "Remote archive directory property (ftpArchiveDirectory)" on page 72.
  f. In the **File extension for remote archive** field, specify the file extension or suffix that the adapter uses to rename the remote FTP file. For more information, see "File extension for remote archive property (ftpRenameExt)" on page 62.
- **Socks proxy server connection information**
  a. In the **Host name** field, specify the host name of the machine used as a proxy server through which the adapter requests are routed to the FTP server. For more information, see "Host name property (SocksProxyHost)" on page 76.
  b. In the **Port number** field, specify the port number of the proxy server through which the adapter requests are routed to the FTP server. For more information, see "Port number property (SocksProxyPort)" on page 77.
  c. In the **User name** field, specify the user name for authenticating the proxy server. For more information, see "User name property (SocksProxyUserName)" on page 77.
  d. In the **Password** field, specify the password used to authenticate the proxy server. For more information, see "Password property (SocksProxyPassword)" on page 76.
- **Secure configuration**
  a. If you want to compare the host key of the SFTP server with the host keys known to the adapter:
     1) Select the **Enable remote server authentication for SFTP protocol** check box. The host key file has to be available with the host keys of the trusted server before the first attempt to connect to SFTP server is made. For more information, see Enable server verification property (EnableServerVerification)
     2) In the **Host key file** field, specify the absolute file path to the host key file. The host key file is created by the administrator and contains

the host keys of all the trusted servers. The Host key file property points to the file on the adapter workstation. For more information, see Host key file property (HostKeyFile)

b. If you want to enable public key authentication, specify the following properties:

   1) In the **Private key file** field, specify the private key used to authenticate to the Secure shell server. For more information, see "Private key file property (PrivateKeyFilePath)" on page 71.

   2) In the **Passphrase** field, specify the phrase used for enhanced security by encrypting the private key. For more information, see Passphrase property (Passphrase)

c. Specify the following properties for the FTPS protocol:

   1) In the **FTPS connection mode** field, specify the connection mode (Implicit or Explicit) to connect to the FTPS server, when FTPS is selected as protocol. For more information, see FTPS connection mode property (ftpsConnectionMode).

   2) In the **Data channel protection level** field, select the level of the data channel protection that you want to use:

      – Select **Private**, if the data transfer between the Adapter and the FTPS server has to be in an encrypted form.

      – Select **Clear**, if the data transfer between the Adapter and the FTPS server has to be in clear text form.

      For more information, see "Data channel protection level (dataProtectionLevel)" on page 149.

   3) In the **Keystore type** field, specify type of the keystore. For more information, see Keystore type property (keyStoreType).

   4) In the **Truststore file** field, specify the path of the truststore file that contains the certificates of the servers trusted by the adapter. For more information, see Truststore file property (trustStorePath).

   5) In the **Truststore password** field, specify the password of the truststore file. It is used to check the integrity of the truststore data. If this value is not specified, the integrity check will not be performed. For more information, see Truststore password property (trustStorePassword).

   6) In the **Keystore file** field, specify the path of the keystore file. The Keystore file contains the private key entry of the FTPS client and also contains a certificate chain for the corresponding public key. For more information, see Keystore file property (keyStorePath).

      Note: Both Keystore file and Truststore file properties share the properties of Keystore type.

   7) In the **Keystore password** field, specify the password of the keystore. It is used to check the integrity of the keystore data. If this value is not specified, integrity check will not be performed. For more information, see Keystore password property (keyStorePassword).

   8) In the **Key password** field, specify the password of the key that is used to recover the keys from the keystore. For more information, see Key password property (keyPassword).

- **Bidi properties**
- **Logging and tracing**

a. If you have multiple instances of the adapter, expand and set Adapter ID to a value that is unique for this instance. For more information about this property, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/rbp_ftp_resource_adapter_props.html.

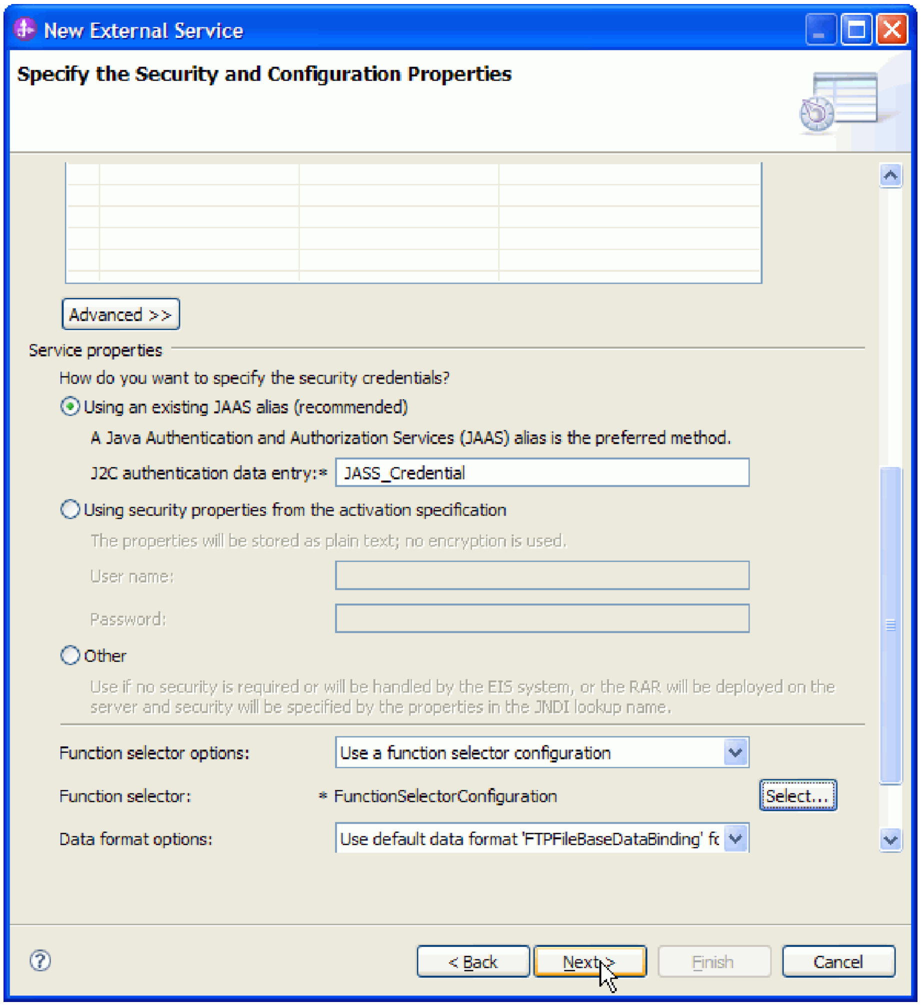
b. Select **Disguise user data as 'XXX' in log and trace files** if you want to prevent sensitive user data from being written to log and trace files. For more information, see Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) .

6. Specify the required security credentials in the **Service Properties** area:
   - To use a J2C authentication alias, select the **Using an existing JAAS alias (recommended)** field, and specify the name of the alias in the **J2C Authentication Data Entry** field. You can specify an existing authentication alias or create one at any time before deploying the module. The name is case sensitive and includes the node name.
   - To use activation specification properties, select the **Using security properties from the activation specification** field, and type the values in the **User name** and **Password** fields.
   - **User name** - Specifies the name of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see "User name property (UserName)" on page 80.
   - **Password** - Specifies the password of the user who has privileges to connect to the FTP server and perform FTP operations. For more information, see "Password property (Password)" on page 69.
   - To administer the user name and password from other mechanism, select **Other**.
7. Select one of the options from the **Function selector** field. A function selector assigns incoming messages or requests to the correct operation on the service.
   - **Function selector options**

     For example, select **Use a Function Selector configuration**. If choosing to use this option, click **Next**.
   - **Function selector**

     If choosing this option, complete the following steps:

     a. Click **Select** next to the **Function Selector** field.



Specify the Security and Configuration Properties

Service properties

How do you want to specify the security credentials?

● Using an existing JAAS alias (recommended)

A Java Authentication and Authorization Services (JAAS) alias is the preferred method.

J2C authentication data entry:* JASS_Credential

○ Using security properties from the activation specification

The properties will be stored as plain text; no encryption is used.

User name:

Password:

○ Other

Use if no security is required or will be handled by the EIS system, or the RAR will be deployed on the server and security will be specified by the properties in the JNDI lookup name.

Function selector options: Use a function selector configuration

Function selector: * FunctionSelectorConfiguration

Data format options: Use default data format 'FTPFileBaseDataBinding' fc

Figure 16. Specify the Security and Configuration Properties window

b. In the Select Function Selector window, select the option, **Use existing function selector from the list**. A list of available function selectors is displayed. Select the function selector (this example uses FilenameFunctionSelector properties). Click **Next**.

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

8. Click **Finish** in the New Function Selector Configuration window.
9. Click **Next** in the Service Configuration Properties window.

Results

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

What to do next

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

If you have selected the **Data format options** as Use a data binding configuration for all operations, proceed to Configuring the data binding and data handler.

Activation specification properties

Activation specification properties are properties that hold the inbound event processing configuration information for a message endpoint.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint (a message driven bean).

You set the activation specification properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the administrative console.

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

Table 157. Activation specification properties

Property name		
In the wizard	In the administrative console	Description
"Ensure once-only event delivery (AssuredOnceDelivery)" on page 117	AssuredOnceDelivery	Specifies whether the adapter provides assured once delivery of events.
"Auto create event table property (EP_CreateTable)" on page 117	`EP_CreateTable`	Tells the adapter whether to create the Event Persistence table
"Create Table property (CreateTable)" on page 118	`CreateTable`	When set to `true`, the event table and related indexes are created
"Custom parser class name property (CustomParserClassName)" on page 118	`CustomParserClassName`	Fully qualified class name of the custom parser which is used to parse the `ls –l` output
"Data channel protection level (dataProtectionLevel)" on page 118	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
"Database Password property (DatabasePassword)" on page 119	`DatabasePassword`	Password used by event persistence for retrieving the JDBC database connection from the data source
"Database schema name property (EP_SchemaName)" on page 119	`EP_SchemaName`	Schema name of the database used by event persistence
"Database Username property (DatabaseUsername)" on page 119	`DatabaseUsername`	User name used by event persistence for retrieving the JDBC database connection from the data source
"FTP server connection mode property (DataConnectionMode)" on page 121	`DataConnectionMode`	Data connection mode used by the FTP server during file transfers

Table 157. Activation specification properties (continued)

"FTPS connection mode property (ftpsConnectionMode)" on page 121	ftpsConnectionMode	Specifies the FTPS connection mode used to set up connection to the FTPS server.
(Not available)	DefaultObjectName	Supported for compatibility with earlier versions
"Delivery type (DeliveryType)" on page 119	DeliveryType	Determines the order in which events are delivered by the adapter to the export.
"Encoding used by FTP server property (EISEncoding)" on page 120	EISEncoding	Encoding of the FTP server
(Not available)	EventContentType	Supported for compatibility with earlier versions
"Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)" on page 120	EP_DataSource_JNDIName	JNDI name of the data source used by event persistence to get the JDBC database connection
"Event recovery table name property (EP_EventTableName)" on page 121	EP_TableName	Name of the table that is used by the adapter for event persistence
"Failure file extension for local archive property (FailedArchiveExt)" on page 122	FailedArchiveExt	File extension used to archive business objects in the event file that are not successfully processed
"File content encoding property (FileContentEncoding)" on page 122	FileContentEncoding	Encoding used to read the event files
"File extension for remote archive property (ftpRenameExt)" on page 122	ftpRenameExt	File extension or suffix that the adapter uses to rename the remote FTP file
"Keystore file property (keyStorePath)" on page 123	keyStorePath	Specifies the path of the keystore that contains the private key entries.
"Keystore password property (keyStorePassword)" on page 123	keyStorePassword	Specifies the password that is used to encrypt the keystore.
"Key password property (keyPassword)" on page 123	keyPassword	Specifies the password that is used to encrypt the key.
"Keystore type property (keyStoreType)" on page 124	keyStoreType	Specifies the type of the keystore.
"Pass only file name and directory, not the content property (FilePassByReference)" on page 125	FilePassByReference	Specifies that the file content of the event file is not sent to the export
"File transfer type property (FileTransferType)" on page 125	FileTransferType	File transfer type used during inbound processing
"Number of files to get at a time property (ftpGetQuantity)" on page 126	ftpGetQuantity	Determines the number of files retrieved from the remote FTP URL
"Number of poll periods between downloads property (ftpPollFrequency)" on page 126	ftpPollFrequency	Determines how frequently the adapter polls the FTP server
Retry limit for failed events	FailedEventRetryLimit	The number of times the adapter attempts to redeliver an event before marking the event as failed.

Table 157. Activation specification properties (continued)

"Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)" on page 127	ftpScriptFileExecutedAfterInbound	Specifies the path of the script file that will be executed after downloading the files from the FTP server
"Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)" on page 127	ftpScriptFileExecutedBeforeInbound	Specifies the path of the script file that will be executed before downloading the files from the FTP server
"Host name property (HostName)" on page 127	HostName	Host name of the FTP Server to which the connection is established
"Include business object delimiter in the file content property (IncludeEndBODelimiter)" on page 128	IncludeEndBODelimiter	When set to true, the delimiter is sent with the business object content for further processing
"Local archive directory property (LocalArchiveDirectory)" on page 128	LocalArchiveDirectory	Absolute path of the local Archive directory
"Local directory property (LocalEventDirectory)" on page 128	LocalEventDirectory	Local system directory into which the adapter downloads event files from the FTP site
"Maximum connections (MaximumConnections)" on page 129	MaximumConnections	The maximum number of connections that the adapter can use for inbound event delivery.
"Minimum connections (MinimumConnections)" on page 129	MinimumConnections	The minimum number of connections that the adapter can use for inbound event delivery.
"File extension for local archive property (originalArchiveExt)" on page 129	OriginalArchiveExt	File extension used to archive the original event file
Passphrase property	passPhrase	Used for enhanced security by encrypting the private key
"Password property (Password)" on page 130	Password	Password of the user who has privileges to connect to the FTP server and perform FTP operations
"Password used to connect to event data source property (EP_Password)" on page 130	EP_Password	Password used during event persistence
"Interval between polling periods (PollPeriod)" on page 130	PollPeriod	The length of time that the adapter waits between polling periods.
"Maximum events in polling period (PollQuantity)" on page 131	PollQuantity	The number of events the adapter delivers to the export during each poll period.
"Port number property (PortNumber)" on page 131	PortNumber	Port number of the FTP server
"Private key file property (PrivateKeyFilePath)" on page 131	PrivateKeyFilePath	Private key used to authenticate to the Secure shell server
"Protocol property (Protocol)" on page 132	Protocol	Specifies if the connection to the FTP server is normal FTP or secure FTP.
"Retrieve files with this pattern property (EventFileMask)" on page 134	EventFileMask	Filter for the event files

Table 157. Activation specification properties (continued)

Retry EIS connection on startup	RetryConnectionOnStartup	Controls whether the adapter retries the connection to the FTP server if it cannot connect at startup.
Time between retries in case of system connection failure (milliseconds)	RetryInterval	The length of time that the adapter waits between attempts to establish a new connection after an error during inbound operations.
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.
"Remote archive directory property (ftpArchiveDirectory)" on page 132	`ftpArchiveDirectory`	Relative path of the archive directory on the FTP server
"Remote directory property (EventDirectory)" on page 133	`EventDirectory`	Remote directory of the FTP server from where the event files are retrieved for inbound processing
Enable server verification	EnableServerVerification	Enables the remote server verification for SFTP protocol
Host key file	HostKeyFile	The absolute path of the host key file that contains the host keys of the trusted servers
"Host name property (SocksProxyHost)" on page 137	`SocksProxyHost`	Host name of the machine used as a proxy server
"Password property (SocksProxyPassword)" on page 137	`SocksProxyPassword`	Password used to authenticate the proxy server
"Port number property (SocksProxyPort)" on page 137	`SocksProxyPort`	Port number of the proxy server
"User name property (SocksProxyUserName)" on page 137	`SocksProxyUserName`	User name used to authenticate the proxy server
"Sort event files property (SortEventFiles)" on page 138	`SortEventFiles`	Determines the sorting order of event files being polled
"Specify criteria to split file content property (SplitCriteria)" on page 138	`SplitCriteria`	Takes different values based on the value of the SplittingFunctionClassName property
"Splitting function class name property" on page 139	`SplittingFunctionClassName`	Takes the fully qualified class name of the class file to be used to enable file splitting
"Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 140	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling.
"Success file extension for local archive property (SuccessArchiveExt)" on page 140	`SuccessArchiveExt`	File extension used to archive all the successfully processed business objects
"Truststore file property (trustStorePath)" on page 124	trustStorePath	Specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.
"Truststore password property (trustStorePassword)" on page 124	trustStorePassword	Specifies the password of the truststore.

Table 157. Activation specification properties (continued)

"Time interval for polling unchanged files (fileUnchangedTimeInterval)" on page 125	fileUnchangedTimeInterval	Specifies the time interval for the adapter to monitor the files for any updates in the content.
"User name property (UserName)" on page 140	`UserName`	Name of the user who has privileges to connect to the FTP server and perform FTP operations
"User name used to connect to event data source property (EP_UserName)" on page 141	`EP_UserName`	User name used by event persistence for getting the database connection
Rule editor to filter files	ruleString	The collection of rules used to filter the events.
"Enable remote verification property (enableRemoteVerification)" on page 134	`enableRemoteVerification`	Used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

Ensure once-only event delivery (AssuredOnceDelivery)

This property specifies whether to provide ensure once-only event delivery for inbound events.

Table 158. 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

Auto create event table property (EP_CreateTable)

Tells the adapter whether to create the Event Persistence table. If the value is `true` and table does not exist then the adapter creates the table. If the value is `false` the adapter does not create the table.

Table 159. Auto create event table property characteristics

Required	No
Default	`true`
Property type	Boolean
Globalized	No

Create Table property (CreateTable)

When set to `true`, the event table and related indexes are created. For troubleshooting table creation errors, set this property to `false`. The table and indexes can then be created manually.

Table 160. Create Table property characteristics

Required	No
Default	`true`
Property type	Boolean
Globalized	No

Custom parser class name property (CustomParserClassName)

Fully qualified class name of the custom parser which is used to parse the `ls –l` output. Used only when the `ls –l` output deviates from standard output.

Table 161. Custom parser class name property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the "PBSZ 0" command before issuing the PROT command.

Table 162. Data channel protection level property characteristics

Required	No

Possible values	`Private - Data is transferred in encrypted form`
	`Clear - Data is transferred as clear text`

Default	`Private - Data is transferred in encrypted form`
Property type	String

Table 162. Data channel protection level property characteristics (continued)

Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Database Password property (DatabasePassword)

Password used by event persistence for retrieving the JDBC database connection from the data source.

Table 163. Database Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database schema name property (EP_SchemaName)

Schema name of the database used by event persistence.

Table 164. Database schema name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database Username property (DatabaseUsername)

User name used by event persistence for retrieving the JDBC database connection from the data source.

Table 165. Database username property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Delivery type (DeliveryType)

This property specifies the order in which events are delivered by the adapter to the export.

Table 166. 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.
Globalized	No
Bidi supported	No

Encoding used by FTP server property (EISEncoding)

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

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

Specify any Java-supported encoding set for this attribute.

Table 167. Encoding used by FTP server property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)

JNDI name of the data source used by event persistence to get the JDBC database connection. The data source must be created in WebSphere Process Server. The database name specified while creating the data source must exist.

Table 168. Event recovery data source (JNDI) name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery table name property (EP_EventTableName)

Name of the table that is used by the adapter for event persistence. When using multiple activation specifications, this value must be unique for each. The same table name must not be used by other instances of same adapter or a different adapter. If the table does not exist in the database, the adapter will create the table.

Table 169. Event recovery table name property characteristics

Required	No
Default	FTPTABLE
Property type	String
Globalized	Yes

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Accepts either active or passive settings.

Table 170. FTP server connection mode property characteristics

Required	No
Default	active
Property type	String
Globalized	No

FTPS connection mode property (ftpsConnectionMode)

This property is used to specify the connection mode when establishing a connection with the FTPS server. The WebSphere Adapter for FTP now supports both Implicit and Explicit connection modes. This property is used when you select either FTP over secure sockets layer (SSL) protocol or FTP over transport layer security (TLS) protocol.

Table 171. FTPS connection mode property characteristics

Required	No
Possible values	Explicit Implicit
Default	Explicit
Property type	String

Table 171. FTPS connection mode property characteristics (continued)

Usage	This property represents the mode used to connect to the FTPS server. When this property is set to: • Explicit connection mode, initially the connection is established as a normal FTP connection. To send sensitive information, such as password the adapter switches to a secure FTP connection by issuing an `AUTH` command. **Note:** The default port for Explicit connection mode is 21. • Implicit connection mode, the connection is established as a secure FTP connection. All communications between the adapter and the server continues in a secure mode. There is no exchange of clear text information between the Adapter and the server. **Note:** The default port for Implicit connection mode is 990.
Globalized	No
Bidi supported	No

Failure file extension for local archive property (FailedArchiveExt)

File extension used to archive business objects in the event file that are not successfully processed. This property is used only when LocalArchiveDirectory is valid and exists.

Table 172. Failure file extension for local archive property characteristics

Required	No
Default	`fail`
Property type	String
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used to read the event files based on the EndBODelimiter property and during string to byte[] conversions. If not specified, the adapter attempts to read without any specific encoding. You can specify any Java supported encoding set.

Table 173. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File extension for remote archive property (ftpRenameExt)

File extension or suffix that the adapter uses to rename the remote FTP file after the connector has polled for it. Renaming the file prevents the connector from polling the same file in the next poll cycle. The adapter can be configured to rename the processed event file and move it to an archive directory.

Table 174. File extension for remote archive property characteristics

Required	No
Default	None

Table 174. File extension for remote archive property characteristics (continued)

Property type	String
Globalized	Yes

Keystore file property (keyStorePath)

This property specifies the path of the keystore that contains the private key entries.

Table 175. Keystore file property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the absolute path of the keystore file on the adapter machine (on which the adapter is running). The keystore file contains the private key entry of the FTPS client. It is also accompanied by a certificate chain for the corresponding public key. The keystore data is used to authenticate the clients identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Keystore password property (keyStorePassword)

This property specifies the password that is used to encrypt the keystore.

Table 176. Keystore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the keystore. It is used to check the integrity of the keystore data. If the value is not specified, integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Key password property (keyPassword)

This property specifies the password that is used to encrypt the key.

Table 177. Key password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the key that is used to recover the key from the keystore. The property is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes

Table 177. Key password property characteristics (continued)

Bidi supported	No

Keystore type property (keyStoreType)

This property specifies the type of keystore.

Table 178. Keystore type property characteristics

Required	No
Possible values	JKS and PKCS12
Default	JKS
Property type	String
Usage	This property specifies the type of the keystore. It is applicable only if you select FTP over SSL or FTP over TLS as the protocol. This property is also applicable for the type of the truststore.
Globalized	No
Bidi supported	No

Truststore file property (trustStorePath)

This property specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.

Table 179. Truststore file property characteristics

Required	This property is required only if you set the protocol as FTP over SSL or FTP over TLS
Default	None
Property type	String
Usage	This property specifies the absolute path of the truststore file on the adapter machine (on which the adapter is running). The truststore file contains the certificates of FTPS servers trusted by the adapter and is used to authenticate the servers identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Truststore password property (trustStorePassword)

This property specifies the password of the truststore.

Table 180. Truststore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password for the truststore. It is used to check the integrity of the truststore data. If the value is not specified, the integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes

Table 180. Truststore password property characteristics (continued)

Bidi supported	No

Time interval for polling unchanged files (fileUnchangedTimeInterval)

This property specifies the time interval for the adapter to monitor the files for any updates in the content. The adapter polls only those files that are not changed during the specified time interval.

Table 181. Time interval for polling unchanged file

Required	No
Default	0
Unit of measure	Milliseconds
Property type	Integer
Usage	This property enables the adapter to poll 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. If the value is set to '0' the adapter polls the files instantly and does not check if the files are being modified.
Globalized	No
Bidi supported	No

Pass only file name and directory, not the content property (FilePassByReference)

Specifies that the file content of the event file is not sent to the export.

If set to `true`, the file is appended with a timestamp and sent to the LocalArchiveDirectory. The timestamp prevents errors and overwrites to the file when another file with the same name is received. This property can be set to `true` only when the LocalArchiveDirectory property is set and the specified directory exists. This property is used only for PassThrough inbound processing. When enabled, the file is not split into chunks.

Table 182. Pass only file name and directory, not the content property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during inbound processing. Accepts either ASCII or binary.

Table 183. File transfer type property characteristics

Required	No
Default	`binary`

Table 183. File transfer type property characteristics (continued)

Property type	String
Globalized	no

Number of files to get at a time property (ftpGetQuantity)

Determines the number of files retrieved from the remote FTP URL with each remote poll.

Table 184. Number of files to get at a time property characteristic

Required	Yes
Default	10
Property type	Integer
Globalized	No

Number of poll periods between downloads property (ftpPollFrequency)

Determines how frequently the adapter polls the FTP server, measured in the number of standard poll cycles. For example, if PollPeriod is set to 10000, and FTPPollFrequency is set to 6, the adapter polls the LocalEventDirectory every 10 seconds and polls the remote EventDirectory every 60 seconds. The adapter performs FTP polling only if you specify a value for this property. If PollPeriod is 0, you consider it as 1 for calculation. If the calculation evaluates to 0, the adapter does not perform FTP polling.

Table 185. Number of poll periods between downloads property characteristics

Required	Yes
Default	5
Property type	Integer
Globalized	No

Retry limit for failed events (FailedEventRetryLimit)

This property specifies the number of times that the adapter attempts to redeliver an event before marking the event as failed.

Table 186. Retry limit for failed events details

Required	No
Possible values	Integers
Default	5
Property type	Integer

Table 186. Retry limit for failed events details (continued)

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

Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)

Specifies the path of the script file that will be executed after downloading the files from the FTP server.

Table 187. Run FTP script file after downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)

Specifies the path of the script file that will be executed before downloading the files from the FTP server.

Table 188. Run FTP script file before downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Host name property (HostName)

Host name of the FTP Server to which the connection is established during inbound processing.

Table 189. Create Table property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Include business object delimiter in the file content property (IncludeEndBODelimiter)

When set to `true`, the delimiter is sent with the business object content for further processing. This property is valid only when splitting the event files based on a delimiter.

Table 190. Include business object delimiter in the file content property characteristics

Required	No
Default	`false`
Property type	String
Globalized	No

Local archive directory property (LocalArchiveDirectory)

Absolute path of the local Archive directory. The directory must be valid and exist.

Table 191. Local archive directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the local archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${LOCALARCHIVE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **LocalArchiveDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Local directory property (LocalEventDirectory)

Local system directory into which the adapter downloads event files from the FTP site. You must specify a value for this property to enable the adapter to process events.

Table 192. Local directory property characteristics

Required	Yes
Default	None
Property type	String

Table 192. Local directory property characteristics (continued)

Usage	You can use a WebSphere Application Server environment variable to represent the local event directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${LOCAL_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **LocalEventDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Maximum connections (MaximumConnections)

This property specifies the maximum number of connections that the adapter can use for inbound event delivery.

Table 193. Maximum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. The adapter considers any positive entry less than 1 to be equal to 1. Typing a negative value for this property may result in run time errors.
Globalized	No
Bidi supported	No

Minimum connections (MinimumConnections)

This property specifies the minimum number of connections that the adapter can use for inbound event delivery.

Table 194. Minimum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. Any value less than 1 is treated as 1 by the adapter. Typing a negative value or 1 for this property may result in run time errors.
Globalized	No
Bidi supported	No

File extension for local archive property (originalArchiveExt)

File extension used to archive the original event file. This preserves the entire event file for reference in case any of its business objects fail. This property is used only when LocalArchiveDirectory is valid and exists.

Table 195. File extension for local archive property characteristics

Required	No
Default	`original`
Property type	String
Globalized	Yes

Password property (Password)

Password of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the password is included in the URL specified in the EventDirectory property.

Table 196. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password used to connect to event data source property (EP_Password)

The password used during event persistence to get the database connection from the data source.

Table 197. Password used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Interval between polling periods (PollPeriod)

This property specifies the length of time that the adapter waits between polling periods.

Table 198. 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 199. 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
Bidi supported	No

Passphrase property (passPhrase)

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

Table 200. Passphrase property property characteristics

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

Port number property (PortNumber)

Port number of the FTP server through which the connection is established during inbound processing.

Table 201. Port number property characteristics

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

Private key file property (PrivateKeyFilePath)

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

Table 202. Private key property characteristics

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

Protocol property (Protocol)

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

For example:

Normal connection: `FTP`

FTP over SSL connection: `FTPS_SSL`

FTP over TLS connection: `FTPS_TLS`

FTP over SSH connection: `SFTP`

Table 203. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Remote archive directory property (ftpArchiveDirectory)

Relative path of the archive directory on the FTP server. The directory must exist. There are several options for using this property to specify archiving:

- Specifying a value for this property, but no value for the FTPRenameExt property causes the adapter to append a timestamp to the event file name and move it to the FTP server archive directory specified in this property.
- Specifying a value for this property and the FTPRenameExt property causes the adapter to rename the processed event file name with a timestamp and the value specified in FTPRenameExt and moves it to the FTP server archive directory specified in this property.
- Specifying no value either for this property or the FTPRenameExt property causes the adapter to delete the processed event file without archiving it.
- Specifying no value for this property but specifying a value for the FTPRenameExt property causes the adapter to rename the processed event file, adding a timestamp and the value specified in FTPRenameExt.

The value of remote archive directory property accepts both the absolute and relative paths of the directory. If the value does not start with a forward slash, the adapter considers the path to be relative to your home directory.

Table 204. Remote archive directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the remote archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${REMOTEARCHIVE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. The archive directory located on the FTP server and used in inbound configuration represents the absolute path of the archive directory. It does not contain any host name or URL information. This directory is located on the same FTP server where the Event Directory is located, for example: /home/archive. **Note:** The **FTPArchiveDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Remote directory property (EventDirectory)

Remote directory of the FTP server from where the event files are retrieved for inbound processing. If the value of Remote directory is set to <HOME_DIR>, the adapter polls for event files in the user's home directory.

The value of event directory property accepts both the absolute and relative paths of the directory. If the value does not start with a forward slash, the adapter considers the path to be relative to the user's home directory.

Table 205. Remote directory property characteristics

Required	Yes
Default	<HOME_DIR>
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the remote directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${REMOTE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **EventDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Retrieve files with this pattern property (EventFileMask)

Filter for the event files. The file filter is a well-qualified expression consisting of alphanumeric characters and the * and ? wild cards.

Table 206. Retrieve files with this pattern property characteristics

Required	Yes
Default	`*.*`
Property type	String
Globalized	Yes

Enable remote verification property (enableRemoteVerification)

When a client connects to the FTP server, two kinds of connections or channels are established; a command connection (also known as control connection), and a data connection. The command connection is the one through which the FTP commands are sent (and replies to these commands received) to the server and the data connection is the channel through which the data transfer takes place between the client and the server.

This property is used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

The verification is done while establishing a data connection to perform data transfer.

Note: This property is applicable only to FTP and FTPS protocols.

Table 207. Enable Remote verification property characteristics

Required	No
Possible values	`True` `False`
Default	`True`
Property type	Boolean
Usage	This property verifies if the data connection and the control connection are from the same host system. By default, the remote verification property is set to TRUE by the FTP server. When this property is set to: • True, during run time, the adapter checks if the data connection is established with the same host as the control connection. If the data connection is established from a different host than the control connection, then an exception is thrown and the connection fails. • False, remote verification is not performed. **Note:** Disabling the remote verification leads to low security. Precaution must be taken before disabling the remote verification.
Globalized	No
Bidi supported	No

Retry EIS connection on startup (RetryConnectionOnStartup)

This property controls whether the adapter attempts to connect again to the FTP server if it cannot connect at startup.

Table 208. 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 FTP server 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 FTP server, for example, when you are building and testing the application that receives events from the adapter. If the adapter cannot connect, the adapter writes log and trace information and stops. The administrative console shows the application status as `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 establish a new connection.

Table 209. 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

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 210. Number of times to retry the system connection details

Required	No
Possible values	0 and positive integers
Default	0
Property type	Integer
Usage	This property controls how many times the adapter retries the connection if the adapter cannot connect to the FTP server to perform inbound processing. A value of 0 indicates an infinite number of retries. To control whether the adapter retries if it cannot connect to the FTP server when it is first started, use the RetryConnectionOnStartup property.
Globalized	No
Bidi supported	No

Enable server verification property (EnableServerVerification)

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

Table 211. Enable server verification property details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	When this property is set to: • True, server authentication is enabled • False, server authentication is disabled The adapter checks for the HostKeyFile property in the path of the file that contains the host keys of the trusted servers.
Globalized	Yes
Bidi supported	No

Host key file property (HostKeyFile)

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

Table 212. Host key file property characteristics

Required	This property has to be specified if the EnableServerVerification property is enabled.
Default	None
Property type	String

Table 212. Host key file property characteristics (continued)

Usage	This is used by the adapter to verify the host key of the remote server with the host keys of the trusted servers specified in this file.
Globalized	Yes
Bidi supported	No

Host name property (SocksProxyHost)

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

Table 213. Host name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SocksProxyPassword)

Password used to authenticate the proxy server.

Table 214. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Port number property (SocksProxyPort)

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

Table 215. Port number property characteristics

Required	No
Default	`1080`
Property type	Integer
Globalized	No

User name property (SocksProxyUserName)

User name used to authenticate the proxy server.

Table 216. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Sort event files property (SortEventFiles)

Determines the sorting order of event files being polled. Supported values are:

- by file name – sort ascending on file name
- by time stamp – sort ascending on last modified timestamp
- no sort – not sorted

Event file ordering from which events need to be delivered is valid only if the activation specification DeliveryType property is set to `ORDERED`. File name sorting is provided based on the locale of the FTP server. The ICU4J package is used to track the locales and their corresponding rules.

Table 217. Sort event files property characteristics

Required	No
Default	no sort (= not sorted)
Property type	String
Globalized	No

Specify criteria to split file content property (SplitCriteria)

This property takes different values based on the value of the SplittingFunctionClassName property. For example: To specify that a file is to be split every 5 KB, set the SplitCriteria property to `5000`.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
  - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
  - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=0, chunking is disabled.

When FilePassByReference is enabled during inbound PassThrough, the event file is not split.

Note: For input files that contain multiple COBOL copybook records, in order to enable file splitting by size you must provide the correct length of each record. To determine the size of each record, use one of these methods:

1. Open the Business Object in a text editor.
   a. For example:

```
<element name="CustomerNumber">
<annotation>
<appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
<td:typeDescriptorElement>
<td:initialValue kind="SPACE"/>
<td:simpleInstanceTD accessor="readWrite" attributeInBit="false"
contentSize="5" offset="0" size="5">
<td:sharedType>
<td:stringTD addrUnit="byte" alignment="byte" characterSize="1"
lengthEncoding="fixedLength" paddingCharacter=" "
prefixLength="0" width="5"/>
</td:sharedType>
</td:simpleInstanceTD>
```

```
</td:typeDescriptorElement>
</appinfo>
</annotation>
<simpleType>
   <restriction base="string">
    <maxLength value="5"/>
  </restriction>
 </simpleType>
</element>
```

Each element in the business object has a corresponding `<element>` entry.

b. Look for a restriction tag for each element tag (the COBOL data binding requires a fixed-width data handler).

c. Add up the `maxLength` attribute values for each of the elements. In this example, the value is 5. The sum of the `maxLength` values is the size of each record of type DFHCOMMAREA.

2. Open the Business Object in a text editor.

   a. Look for the complex type tag with the business object name value in the name attribute. In the example that follows, the business object name is DFHCOMMAREA.

   b. Locate a namespace-appended tag called `aggregateInstanceTD` and use the value for the attribute `contentSize`. In this example, the value is 117. This is the size of each record of type DFHCOMMAREA.

```
<complexType name="DFHCOMMAREA">
<annotation>
<appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
<td:typeDescriptorCT>
<td:aggregateInstanceTD accessor="readWrite" attributeInBit="false"
contentSize="117" offset="0" size="117">
```

Table 218. Specify criteria to split file content property characteristics

Required	No
Default	`0`
Property type	String
Globalized	Yes

Splitting function class name property

This value takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The `com.ibm.j2ca.utils.filesplit.SplitByDelimiter` class that splits the event file based on delimiter.
- The `com.ibm.j2ca.utils.filesplit.SplitBySize` class that splits the event file based on the event file size.

Optionally, you can provide a custom file splitter class and use it by inputting the class name into the SplittingFunctionClassName property.

The delimiter or file size is provided in the SplitCriteria property. If the EventContentType property is set to null, it is automatically set to a class name that performs splitting based on file size.

Table 219. Splitting function class name property characteristics

Required	No

Table 219. Splitting function class name property characteristics (continued)

Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Globalized	No

Stop the adapter when an error is encountered while polling (StopPollingOnError)

This property specifies whether the adapter will stop polling for events when it encounters an error during polling.

Table 220. Stop the adapter when an error is encountered while polling details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If this property is set to `True`, the adapter stops polling when it encounters an error. If this property is set to `False`, the adapter logs an exception when it encounters an error during polling and continues polling.
Globalized	No
Bidi supported	No

Success file extension for local archive property (SuccessArchiveExt)

File extension used to archive all the successfully processed business objects. This property is used only when LocalArchiveDirectory is valid and exists. For example, `12345.order > 12345.order.success`

Table 221. Success file extension for local archive property characteristics

Required	No
Default	`success`
Property type	String
Globalized	Yes

User name property (UserName)

Name of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the user name is included in the URL specified in the EventDirectory property.

Table 222. User name property characteristics

Required	No
Default	None
Property type	String

Table 222. User name property characteristics (continued)

Globalized	Yes

User name used to connect to event data source property (EP_UserName)

User name used by event persistence for getting the database connection from the data source.

Table 223. User name used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Rule editor to filter files

This property is used to filter event files based on a set of rules

Table 224. Rule editor to filter files

Required	Optional
Default	None
Property type	String
Usage	During an inbound processing, if the value in the rule table is specified, then the event files are fetched after filtering, based on the specified rules before polling those event files.
Globalized	Yes
Bidi supported	No

Support for relative path in remote directories

The relative path is supported for various remote directories like output directory, event directory, archive directory, and staging directory.

Inbound processing

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

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

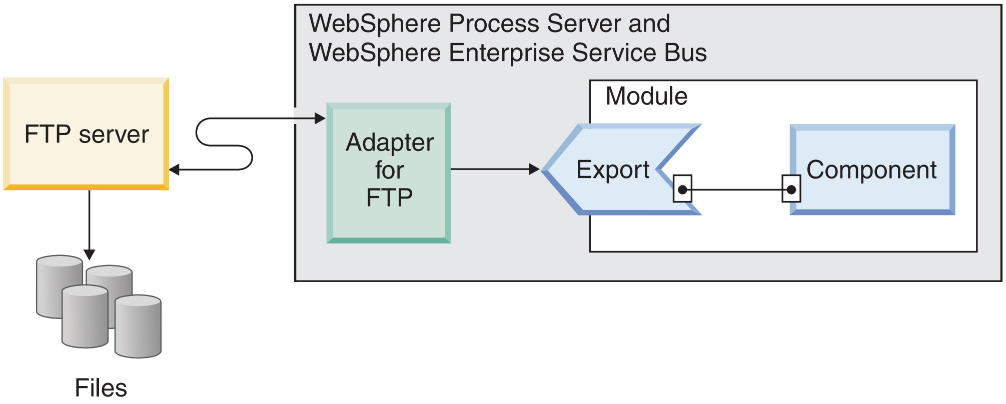

WebSphere Process Server and
WebSphere Enterprise Service Bus
Module
FTP server
Adapter for FTP
Export
Component
Files

Figure 17. Inbound processing flow

The adapter polls files from the event directory of the FTP server at regular intervals based on the FTPPollFrequency property. When a file arrives in the event directory, the adapter reads the entire file and downloads the file to a local event directory on the adapter workstation. The adapter downloads the files from the FTP server sequentially, that is one file at a time, and cannot download all the files simultaneously. After the file is downloaded, the adapter either archives the file in the FTP server in an archive directory given by the FTPArchiveDirectory property or deletes it based on your configuration. The event directory, archive directory, poll frequency, and poll quantity (the number of files to poll in a single poll cycle) are all configurable properties.

Note: If the Remote directory is set to <HOME_DIR>, the adapter polls for event files in `user's` home directory.

Note: The value of an event directory property accepts both the absolute and relative paths of the directory. If the value does not begin with a forward slash (/), the adapter considers the path to be relative to the home directory of the user.

For example, if the value in the remote directory property is set to “ftpuser/event”, the adapter considers this to be the path relative to the home directory. If the home directory is set to “/usr/ftp", then the adapter will poll the directory “/usr/ftp/ftpuser/event” for event files.

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

Inbound event processing consists of the following steps:

1. FTP server generates events in the form of files.
2. The Adapter for FTP polls the event directory.
3. The files are downloaded to the adapter.
4. The files are split based on the SplittingFunctionClassName and SplitCriteria properties. The event file is split into several chunks and each chunk is posted to the export separately. This reduces memory loading during event processing.
   - If splitting is done based on a delimiter, the class that performs this function and the split criteria are provided.
   - If splitting is done based on file size, the class name that performs this function is provided.

- If splitting is done based on other criteria, you must provide your own file splitting class.

5. The adapter sends the data, including the location of the polled document and the host name of the machine that the file was retrieved from, to the export through a function selector, where the configured data binding is invoked, to convert the text record into a business object.

Processing of files using FTP scripts

In addition to processing the files downloaded from the event directory during polling, WebSphere Adapter for FTP can also be used to process the files downloaded using the FTP scripts.

You can specify the scripts to be run before or after polling the event directory using the properties, “Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)” on page 67 and “Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)” on page 66. The script files can contain FTP commands, such as `mget` and `get`, to download the files from the remote directories on the FTP server to the local event directory of the machine where the adapter is installed. The WebSphere Adapter for FTP processes the files that are downloaded to the local event directory configured in the activation specification properties and delivers the processed business objects to the consuming service.

Following is an example of a script:

```
lcd C:\FTPAdapter\localevent
cd /ftpDir1
mget *.txt
cd /ftpDir2
get abc.xml
```

Where, `C:\FTPAdapter\localevent` is the local event directory of WebSphere Adapter for FTP, and ftpDir1 and ftpDir2 are directories that exist on the FTP server. The adapter executes the script and downloads the files to the local event directory. The adapter then processes the files and delivers it to the consuming service.

Note:

1. You must place the files that have been downloaded using the script to the configured local event directory for the adapter to process it. Use the FTP command `lcd` to change the local working directory to the localEventDirectory before you download any files using the script.
2. The files downloaded to the local event directory using the commands, `mget` or `get` will be deleted from the FTP server by the FTP adapter after you download the files. This is to ensure that the files are not downloaded again during the next poll cycle.
3. Use the script file to download the files only from remote directories and not from the event directory of the FTP adapter.

Supported inbound operation

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

Event file locking

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

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

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

Rule-Based filtering of events

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

- FileName
- File Size
- 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.

Though using the rule is optional and specifying an event file mask is mandatory, the rule takes a higher precedence over the event file mask, when both a rule and an event file mask are specified. Event file mask is effective only when there is no rule specified. By default, an event file mask has "*.*" as the 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 225. 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

Table 225. Metadata filtering properties (continued)

Property	Valid operators	Value	Prerequisites
LastModified	Greater than, Less than, Greater than or equal to, Less than or equal to, Equal to, Not equal to. **Note:** Select 'Equal to' operator when you choose the days of 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

Managed (J2C) connection factory properties

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

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

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

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

Table 226. 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.
“Custom parser class name property (CustomParserClassName)” on page 148	CustomParserClassName	Specifies the fully qualified class name of the custom parser that is used to parse the `ls –l` output.
“Data channel protection level (dataProtectionLevel)” on page 149	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
“Default target file name property (Filename)” on page 149	Filename	Specifies the name of the file to be used during outbound operations.
“Maximum retries on connection failure (connectionRetryLimit)” on page 153	connectionRetryLimit	Specifies the number of times the adapter attempts to connect to the FTP server to reestablish the connection.
“Directory property (OutputDirectory)” on page 149	OutputDirectory	Specifies the output directory in the FTP server.

Table 226. Managed connection factory properties (continued)

“Disguise user data as "XXX" in log and trace files (HideConfidentialTrace) ” on page 150	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
Enable server verification	EnableServerVerification	Enables the remote server verification for SFTP protocol
“Encoding used by FTP server property (EISEncoding)” on page 150	EISEncoding	Specifies the encoding of the FTP server.
“FTPS connection mode property (ftpsConnectionMode)” on page 152	`ftpsConnectionMode`	Specifies the FTPS connection mode used to set up connection to the FTPS server.
Host key file	HostKeyFile	The absolute path of the host key file that contains the host keys of the trusted servers
“Host name property (HostName)” on page 153	HostName	Specifies the host name of the FTP server.
“Host name property (SecondServerHostName)” on page 154	secondServerHostName	Specifies the host name of the second FTP server.
“Host name property (SocksProxyHost)” on page 154	SocksProxyHost	Specifies the name of the workstation that is used as a proxy server.
“Keystore file property (keyStorePath)” on page 155	keyStorePath	Specifies the path of the keystore that contains the private key entries.
“Keystore password property (keyStorePassword)” on page 155	keyStorePassword	Specifies the password that is used to encrypt the keystore.
“Key password property (keyPassword)” on page 155	keyPassword	Specifies the password that is used to encrypt the key.
“Keystore type property (keyStoreType)” on page 156	keyStoreType	Specifies the type of the keystore.
Passphrase property	passPhrase	Used for enhanced security by encrypting the private key
“Password property (Password)” on page 157	Password	Specifies the password of the user with privileges to connect to the FTP server and perform FTP operations.
“Password property (SecondServerPassword)” on page 157	SecondServerPassword	Specifies the password of the Second FTP server to which the file is transferred during a server to server file transfer outbound operation.
“Password property (SocksProxyPassword)” on page 157	SocksProxyPassword	Specifies the password used to authenticate the proxy server.
“Port number property (PortNumber)” on page 158	PortNumber	Specifies the port number of the FTP server.
“Port number property (SecondServerPortNumber)” on page 158	SecondServerPortNumber	Specifies the port number of the second FTP server.
“Port number property (SocksProxyPort)” on page 158	SocksProxyPort	Specifies the port number of the proxy server.

Table 226. Managed connection factory properties (continued)

"Private key file property (PrivateKeyFilePath)" on page 158	PrivateKeyFilePath	Private key used to authenticate to the secure shell server.
"Protocol property (Protocol)" on page 159	Protocol	Specifies if the connection to the FTP server is normal FTP or secure FTP.
"Protocol property (SecondServerProtocol)" on page 159	SecondServerProtocol	Specifies the protocol used to connect to the second server.
"Connection retry interval (in milliseconds) (connectionRetryInterval)" on page 159	connectionRetryInterval	Specifies the time interval between attempts to reconnect to the FTP server if the connection fails
"Second Server Directory property (SecondServerDirectory)" on page 160	SecondServerDirectory	Specifies the directory path of the second FTP server to which the ServerToServerFileTransfer outbound operation is performed.
"Sequence file property (FileSequenceLog)" on page 160	FileSequenceLog	Specifies the full path of the file where the sequence number is stored for the outbound Create process.
"Staging directory property (StagingDirectory)" on page 161	StagingDirectory	Specifies the directory that the file is first created in to.
"Truststore file property (trustStorePath)" on page 156	trustStorePath	Specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.
"Truststore password property (trustStorePassword)" on page 156	trustStorePassword	Specifies the password of the truststore.
"User name property (SecondServerUserName)" on page 161	SecondServerUserName	Specifies the user name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.
"User Name property (SocksProxyUserName)" on page 162	SocksProxyUserName	Specifies the user name used to authenticate to the proxy server.
"User name property (Username)" on page 162	Username	Specifies the name of the user.
"Enable remote verification property (enableRemoteVerification)" on page 151	`enableRemoteVerification`	Used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

Adapter ID (AdapterID)

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

Table 227. Adapter ID details

Required	Yes
Default	`001`
Property type	String

Table 227. 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, `FTPRA`, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to `001`, the component ID is `FTPRA001`. If you run multiple instances of the same adapter, ensure that the first eight 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 FTP to `001` and `002`. The component IDs for those instances, `FTPRA001` and `FTPRA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. 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 `FTPRAInstance`. 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 WebSphere Integration Developer 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

Custom parser class name property (CustomParserClassName)

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

Table 228. Custom parser class name property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the “PBSZ 0” command before issuing the PROT command.

Table 229. Data channel protection level property characteristics

Required	No

Possible values	`Private - Data is transferred in encrypted form`
	`Clear - Data is transferred as clear text`

Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Default target file name property (Filename)

Specifies the name of the file that is used during outbound operations.

Table 230. Default target file name property characteristics

Required	Yes
Default	`Yes`
Property type	String
Usage	Use the WebSphere Application Server environment variable to represent the file name directory. Specify the name of the environment variable within braces, preceded by a $ symbol. For example: `${FILENAME}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation for more information.
Globalized	No

Directory property (OutputDirectory)

This is the Output directory in the FTP Server that the outbound operation is performed on. If the value of the Directory is set to <HOME_DIR>, the adapter performs the outbound operations in your home directory.

The value of output directory property accepts both the absolute and relative paths of the directory. If the value does not start with a forward slash, the adapter considers the path to be relative to your home directory.

Table 231. Directory property characteristics

Required	Yes
Default	<HOME_DIR>
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the output directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${OUTPUT_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation for more information.
Globalized	Yes

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 232. 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 WebSphere Integration Developer 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

Encoding used by FTP server property (EISEncoding)

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

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

Specify any Java-supported encoding set for this attribute.

Table 233. Encoding used by FTP server property characteristics

Required	No
Default	None
Property type	String
Globalized	No

Enable server verification property (EnableServerVerification)

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

Table 234. Enable server verification property details

Required	No
Possible values	`True` `False`
Default	False
Property type	Boolean
Usage	When this property is set to: • True, server authentication is enabled • False, server authentication is disabled The adapter checks for the HostKeyFile property in the path of the file that contains the host keys of the trusted servers.
Globalized	Yes
Bidi supported	No

Enable remote verification property (enableRemoteVerification)

When a client connects to the FTP server, two kinds of connections or channels are established; a command connection (also known as control connection), and a data connection. The command connection is the one through which the FTP commands are sent (and replies to these commands received) to the server and the data connection is the channel through which the data transfer takes place between the client and the server.

This property is used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

The verification is done while establishing a data connection to perform data transfer.

Note: This property is applicable only to FTP and FTPS protocols.

Table 235. Enable Remote verification property characteristics

Required	No
Possible values	`True` `False`
Default	`True`
Property type	Boolean
Usage	This property verifies if the data connection and the control connection are from the same host system. By default, the remote verification property is set to TRUE by the FTP server. When this property is set to: • True, during run time, the adapter checks if the data connection is established with the same host as the control connection. If the data connection is established from a different host than the control connection, then an exception is thrown and the connection fails. • False, remote verification is not performed. **Note:** Disabling the remote verification leads to low security. Precaution must be taken before disabling the remote verification.
Globalized	No
Bidi supported	No

FTPS connection mode property (ftpsConnectionMode)

This property is used to specify the connection mode when establishing a connection with the FTPS server. The WebSphere Adapter for FTP now supports both Implicit and Explicit connection modes. This property is used when you select either FTP over secure sockets layer (SSL) protocol or FTP over transport layer security (TLS) protocol.

Table 236. FTPS connection mode property characteristics

Required	No
Possible values	`Explicit` `Implicit`
Default	`Explicit`
Property type	String

Table 236. FTPS connection mode property characteristics (continued)

Usage	This property represents the mode used to connect to the FTPS server. When this property is set to: • Explicit connection mode, initially the connection is established as a normal FTP connection. To send sensitive information, such as password the adapter switches to a secure FTP connection by issuing an AUTH command. **Note:** The default port for Explicit connection mode is 21. • Implicit connection mode, the connection is established as a secure FTP connection. All communications between the adapter and the server continues in a secure mode. There is no exchange of clear text information between the Adapter and the server. **Note:** The default port for Implicit connection mode is 990.
Globalized	No
Bidi supported	No

Host key file property (HostKeyFile)

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

Table 237. Host key file property characteristics

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

Host name property (HostName)

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

Table 238. Host name property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Maximum retries on connection failure (connectionRetryLimit)

This property specifies the number of times the adapter will attempt to reestablish a connection to the FTP server, when the adapter encounters an error related to the outbound connection.

Note: If connection timeout is configured at the FTP server, the appropriate values for connectionRetryLimit and connectionRetryInterval needs to be set. The values for properties should be set so that the FTP adapter retries the outbound request automatically if any connection error occurs due to timeout.

Table 239. Maximum retries on connection failure property characteristics

Required	No
Possible values	Integers equal to and greater than zero
Default	0
Property type	Integer
Usage	When this property is set to: **0** • The adapter does not attempt to reconnect to the FTP server, if an error occurs during startup or while establishing a connection. • The adapter does not verify if the connection to the FTP server is valid when there is an outbound request during runtime. **>0** • The adapter attempts to reconnect to the FTP server for the specified number of times, if an error occurs during startup or while establishing a connection. • The adapter verifies if the connection to the FTP server is valid when there is an outbound request during runtime. If the connection is not valid, it is terminated and a new connection is created to process the request. If the adapter fails to establish a connection after trying for the specified number of times, a connection error is generated. If the adapter is successful in reestablishing the connection, the outbound operation is completed.
Globalized	No
Bidi supported	No

Host name property (SecondServerHostName)

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

Table 240. Host name property characteristics

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

Host name property (SocksProxyHost)

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

Table 241. Host name property characteristics

Required	No

Table 241. Host name property characteristics (continued)

Default	None
Property type	String
Globalized	Yes

Keystore file property (keyStorePath)

This property specifies the path of the keystore that contains the private key entries.

Table 242. Keystore file property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the absolute path of the keystore file on the adapter machine (on which the adapter is running). The keystore file contains the private key entry of the FTPS client. It is also accompanied by a certificate chain for the corresponding public key. The keystore data is used to authenticate the clients identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Keystore password property (keyStorePassword)

This property specifies the password that is used to encrypt the keystore.

Table 243. Keystore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the keystore. It is used to check the integrity of the keystore data. If the value is not specified, integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Key password property (keyPassword)

This property specifies the password that is used to encrypt the key.

Table 244. Key password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the key that is used to recover the key from the keystore. The property is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.

Table 244. Key password property characteristics (continued)

Globalized	Yes
Bidi supported	No

Keystore type property (keyStoreType)

This property specifies the type of keystore.

Table 245. Keystore type property characteristics

Required	No
Possible values	JKS and PKCS12
Default	JKS
Property type	String
Usage	This property specifies the type of the keystore. It is applicable only if you select FTP over SSL or FTP over TLS as the protocol. This property is also applicable for the type of the truststore.
Globalized	No
Bidi supported	No

Truststore file property (trustStorePath)

This property specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.

Table 246. Truststore file property characteristics

Required	This property is required only if you set the protocol as FTP over SSL or FTP over TLS
Default	None
Property type	String
Usage	This property specifies the absolute path of the truststore file on the adapter machine (on which the adapter is running). The truststore file contains the certificates of FTPS servers trusted by the adapter and is used to authenticate the servers identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Truststore password property (trustStorePassword)

This property specifies the password of the truststore.

Table 247. Truststore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password for the truststore. It is used to check the integrity of the truststore data. If the value is not specified, the integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.

Table 247. Truststore password property characteristics (continued)

Globalized	Yes
Bidi supported	No

Passphrase property (passPhrase)

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

Table 248. Passphrase property property characteristics

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

Password property (Password)

Specifies, the password of the user with privileges to connect to the FTP server and perform FTP operations.

Table 249. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SecondServerPassword)

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

Table 250. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SocksProxyPassword)

Specifies the password used to authenticate the proxy server.

Table 251. Password property characteristics

Required	No
Default	None
Property type	String

Table 251. Password property characteristics (continued)

Globalized	Yes

Port number property (PortNumber)

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

Table 252. Port number property characteristics

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

Port number property (SecondServerPortNumber)

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

Table 253. Port number property characteristics

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

Port number property (SocksProxyPort)

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

Table 254. Port number property characteristics

Required	No
Default	1080
Property type	Integer
Globalized	No

Private key file property (PrivateKeyFilePath)

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

Table 255. Private key property characteristics

Required	No
Default	None
Property type	String
Usage	Absolute path of the file which contains the private key. Used to authenticate the user to the secure shell server.

Table 255. Private key property characteristics (continued)

Example	`c:\temp\key.ppk`
Globalized	Yes
Bidi supported	No

Protocol property (Protocol)

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

For example:

Normal connection: `FTP`

FTP over SSL connection: `FTPS_SSL`

FTP over TLS connection: `FTPS_TLS`

FTP over SSH connection: `SFTP`

Table 256. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Protocol property (SecondServerProtocol)

Specifies the protocol that is used to establish a connection to the second server. The FTP protocol is used in establishing the connection.

Table 257. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Connection retry interval (in milliseconds) (connectionRetryInterval)

This property specifies the time interval between attempts to reconnect to the FTP server if the connection fails.

Table 258. Connection retry interval (in milliseconds) property characteristics

Required	No
Possible values	Integers equal to and greater than 0
Default	60000
Unit of measure	Milliseconds
Property type	Integer

Table 258. Connection retry interval (in milliseconds) property characteristics (continued)

Usage	This property is applicable only if the value of the property “Maximum retries on connection failure” is set to greater than 0. When the adapter encounters an error while establishing a connection to the FTP server, this property specifies the time interval the adapter waits between attempts to reestablish a connection.
Globalized	No
Bidi supported	No

Second Server Directory property (SecondServerDirectory)

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

Table 259. Second Server Directory property characteristics

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

Sequence file property (FileSequenceLog)

Specifies the full path of the file where the sequence number will be stored for outbound Create processing.

When the FileSequenceLog property is specified, the adapter generates a unique sequence number to insert into the file name when processing the Create operation.

The sequence of numbers will continue to increment after multiple adapter restarts.

The sequence number is inserted into the file name in the following format:

```
filename.number.extension
```

For example `Customer.3.txt`

When the FileSequenceLog property is not specified or contains an invalid value, no sequence number is generated.

Table 260. Sequence file property characteristics

Required	No
Default	None
Property type	String

Table 260. Sequence file property characteristics (continued)

Usage	**Important:** Unless they are part of a cluster, it is not recommended that two adapter instances access the same sequence file, because concurrent requests result in delay while processing batch requests.
Globalized	No

Staging directory property (StagingDirectory)

During an outbound create operation, a file is first created in the staging directory before it is moved to the directory specified in the DirectoryPath property. The staging directory is also used for the Append and Overwrite operations, where the specified file is copied to StagingDirectory (if present), then appended or overwritten with content and moved back to the original specified directory. If the StagingDirectory is not present, the operation is run in the actual required directory. When you work with a staging directory you can avoid file writing conflicts, which can occur when multiple users are reading the file or while the file is being overwritten during an append and update operation.

The value of staging directory property accepts both the absolute and relative paths of the directory. If the value does not start with a forward slash, the adapter considers the path to be relative to the home directory of the user.

Table 261. Staging directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the staging directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${STAGING_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation for more information.
Globalized	Yes

User name property (SecondServerUserName)

Specifies the user name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 262. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User Name property (SocksProxyUserName)

Specifies the user name used to authenticate the proxy server.

Table 263. User Name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

User name property (Username)

Specifies the name of the user with privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this attribute if the User name is included in the URL specified in the FtpUrl property.

Table 264. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Activation specification properties

Activation specification properties are properties that hold the inbound event processing configuration information for a message endpoint.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint (a message driven bean).

You set the activation specification properties using the external service wizard and can change them using the WebSphere Integration Developer Assembly Editor, or after deployment through the administrative console.

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

Table 265. Activation specification properties

Property name		
In the wizard	In the administrative console	Description
“Ensure once-only event delivery (AssuredOnceDelivery)” on page 166	AssuredOnceDelivery	Specifies whether the adapter provides assured once delivery of events.
“Auto create event table property (EP_CreateTable)” on page 167	`EP_CreateTable`	Tells the adapter whether to create the Event Persistence table
“Create Table property (CreateTable)” on page 167	`CreateTable`	When set to `true`, the event table and related indexes are created

Table 265. Activation specification properties (continued)

"Custom parser class name property (CustomParserClassName)" on page 167	CustomParserClassName	Fully qualified class name of the custom parser which is used to parse the ls –l output
"Data channel protection level (dataProtectionLevel)" on page 168	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
"Database Password property (DatabasePassword)" on page 168	DatabasePassword	Password used by event persistence for retrieving the JDBC database connection from the data source
"Database schema name property (EP_SchemaName)" on page 168	EP_SchemaName	Schema name of the database used by event persistence
"Database Username property (DatabaseUsername)" on page 169	DatabaseUsername	User name used by event persistence for retrieving the JDBC database connection from the data source
"FTP server connection mode property (DataConnectionMode)" on page 170	DataConnectionMode	Data connection mode used by the FTP server during file transfers
"FTPS connection mode property (ftpsConnectionMode)" on page 171	ftpsConnectionMode	Specifies the FTPS connection mode used to set up connection to the FTPS server.
(Not available)	DefaultObjectName	Supported for compatibility with earlier versions
"Delivery type (DeliveryType)" on page 169	DeliveryType	Determines the order in which events are delivered by the adapter to the export.
"Encoding used by FTP server property (EISEncoding)" on page 169	EISEncoding	Encoding of the FTP server
(Not available)	EventContentType	Supported for compatibility with earlier versions
"Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)" on page 170	EP_DataSource_JNDIName	JNDI name of the data source used by event persistence to get the JDBC database connection
"Event recovery table name property (EP_EventTableName)" on page 170	EP_TableName	Name of the table that is used by the adapter for event persistence
"Failure file extension for local archive property (FailedArchiveExt)" on page 171	FailedArchiveExt	File extension used to archive business objects in the event file that are not successfully processed
"File content encoding property (FileContentEncoding)" on page 171	FileContentEncoding	Encoding used to read the event files
"File extension for remote archive property (ftpRenameExt)" on page 172	ftpRenameExt	File extension or suffix that the adapter uses to rename the remote FTP file
"Keystore file property (keyStorePath)" on page 172	keyStorePath	Specifies the path of the keystore that contains the private key entries.
"Keystore password property (keyStorePassword)" on page 172	keyStorePassword	Specifies the password that is used to encrypt the keystore.
"Key password property (keyPassword)" on page 173	keyPassword	Specifies the password that is used to encrypt the key.

Table 265. Activation specification properties (continued)

“Keystore type property (keyStoreType)” on page 173	keyStoreType	Specifies the type of the keystore.
“Pass only file name and directory, not the content property (FilePassByReference)” on page 174	FilePassByReference	Specifies that the file content of the event file is not sent to the export
“File transfer type property (FileTransferType)” on page 175	FileTransferType	File transfer type used during inbound processing
“Number of files to get at a time property (ftpGetQuantity)” on page 175	ftpGetQuantity	Determines the number of files retrieved from the remote FTP URL
“Number of poll periods between downloads property (ftpPollFrequency)” on page 175	ftpPollFrequency	Determines how frequently the adapter polls the FTP server
Retry limit for failed events	FailedEventRetryLimit	The number of times the adapter attempts to redeliver an event before marking the event as failed.
“Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)” on page 176	ftpScriptFileExecutedAfterInbound	Specifies the path of the script file that will be executed after downloading the files from the FTP server
“Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)” on page 176	ftpScriptFileExecutedBeforeInbound	Specifies the path of the script file that will be executed before downloading the files from the FTP server
“Host name property (HostName)” on page 177	HostName	Host name of the FTP Server to which the connection is established
“Include business object delimiter in the file content property (IncludeEndBODelimiter)” on page 177	IncludeEndBODelimiter	When set to true, the delimiter is sent with the business object content for further processing
“Local archive directory property (LocalArchiveDirectory)” on page 177	LocalArchiveDirectory	Absolute path of the local Archive directory
“Local directory property (LocalEventDirectory)” on page 177	LocalEventDirectory	Local system directory into which the adapter downloads event files from the FTP site
“Maximum connections (MaximumConnections)” on page 178	MaximumConnections	The maximum number of connections that the adapter can use for inbound event delivery.
“Minimum connections (MinimumConnections)” on page 178	MinimumConnections	The minimum number of connections that the adapter can use for inbound event delivery.
“File extension for local archive property (originalArchiveExt)” on page 179	OriginalArchiveExt	File extension used to archive the original event file
Passphrase property	passPhrase	Used for enhanced security by encrypting the private key
“Password property (Password)” on page 179	Password	Password of the user who has privileges to connect to the FTP server and perform FTP operations
“Password used to connect to event data source property (EP_Password)” on page 179	EP_Password	Password used during event persistence

Table 265. Activation specification properties (continued)

"Interval between polling periods (PollPeriod)" on page 179	PollPeriod	The length of time that the adapter waits between polling periods.
"Maximum events in polling period (PollQuantity)" on page 180	PollQuantity	The number of events the adapter delivers to the export during each poll period.
"Port number property (PortNumber)" on page 180	`PortNumber`	Port number of the FTP server
"Private key file property (PrivateKeyFilePath)" on page 181	`PrivateKeyFilePath`	Private key used to authenticate to the Secure shell server
"Protocol property (Protocol)" on page 181	`Protocol`	Specifies if the connection to the FTP server is normal FTP or secure FTP.
"Retrieve files with this pattern property (EventFileMask)" on page 183	`EventFileMask`	Filter for the event files
Retry EIS connection on startup	RetryConnectionOnStartup	Controls whether the adapter retries the connection to the FTP server if it cannot connect at startup.
Time between retries in case of system connection failure (milliseconds)	RetryInterval	The length of time that the adapter waits between attempts to establish a new connection after an error during inbound operations.
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.
"Remote archive directory property (ftpArchiveDirectory)" on page 181	`ftpArchiveDirectory`	Relative path of the archive directory on the FTP server
"Remote directory property (EventDirectory)" on page 182	`EventDirectory`	Remote directory of the FTP server from where the event files are retrieved for inbound processing
Enable server verification	EnableServerVerification	Enables the remote server verification for SFTP protocol
Host key file	HostKeyFile	The absolute path of the host key file that contains the host keys of the trusted servers
"Host name property (SocksProxyHost)" on page 186	`SocksProxyHost`	Host name of the machine used as a proxy server
"Password property (SocksProxyPassword)" on page 186	`SocksProxyPassword`	Password used to authenticate the proxy server
"Port number property (SocksProxyPort)" on page 186	`SocksProxyPort`	Port number of the proxy server
"User name property (SocksProxyUserName)" on page 186	`SocksProxyUserName`	User name used to authenticate the proxy server
"Sort event files property (SortEventFiles)" on page 187	`SortEventFiles`	Determines the sorting order of event files being polled
"Specify criteria to split file content property (SplitCriteria)" on page 187	`SplitCriteria`	Takes different values based on the value of the SplittingFunctionClassName property

Table 265. Activation specification properties (continued)

"Splitting function class name property" on page 188	`SplittingFunctionClassName`	Takes the fully qualified class name of the class file to be used to enable file splitting
"Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 189	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling.
"Success file extension for local archive property (SuccessArchiveExt)" on page 189	`SuccessArchiveExt`	File extension used to archive all the successfully processed business objects
"Truststore file property (trustStorePath)" on page 173	trustStorePath	Specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.
"Truststore password property (trustStorePassword)" on page 174	trustStorePassword	Specifies the password of the truststore.
"Time interval for polling unchanged files (fileUnchangedTimeInterval)" on page 174	fileUnchangedTimeInterval	Specifies the time interval for the adapter to monitor the files for any updates in the content.
"User name property (UserName)" on page 189	`UserName`	Name of the user who has privileges to connect to the FTP server and perform FTP operations
"User name used to connect to event data source property (EP_UserName)" on page 190	`EP_UserName`	User name used by event persistence for getting the database connection
Rule editor to filter files	ruleString	The collection of rules used to filter the events.
"Enable remote verification property (enableRemoteVerification)" on page 183	`enableRemoteVerification`	Used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

Ensure once-only event delivery (AssuredOnceDelivery)

This property specifies whether to provide ensure once-only event delivery for inbound events.

Table 266. Ensure once-only event delivery details

Required	Yes
Possible values	`True` `False`
Default	`True`
Property type	Boolean

Table 266. Ensure once-only event delivery details (continued)

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

Auto create event table property (EP_CreateTable)

Tells the adapter whether to create the Event Persistence table. If the value is `true` and table does not exist then the adapter creates the table. If the value is `false` the adapter does not create the table.

Table 267. Auto create event table property characteristics

Required	No
Default	`true`
Property type	Boolean
Globalized	No

Create Table property (CreateTable)

When set to `true`, the event table and related indexes are created. For troubleshooting table creation errors, set this property to `false`. The table and indexes can then be created manually.

Table 268. Create Table property characteristics

Required	No
Default	`true`
Property type	Boolean
Globalized	No

Custom parser class name property (CustomParserClassName)

Fully qualified class name of the custom parser which is used to parse the `ls –l` output. Used only when the `ls –l` output deviates from standard output.

Table 269. Custom parser class name property characteristics

Required	No
Default	None
Property type	String

Table 269. Custom parser class name property characteristics (continued)

Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the "PBSZ 0" command before issuing the PROT command.

Table 270. Data channel protection level property characteristics

Required	No

Possible values	`Private - Data is transferred in encrypted form`
	`Clear - Data is transferred as clear text`

Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Database Password property (DatabasePassword)

Password used by event persistence for retrieving the JDBC database connection from the data source.

Table 271. Database Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database schema name property (EP_SchemaName)

Schema name of the database used by event persistence.

Table 272. Database schema name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Database Username property (DatabaseUsername)

User name used by event persistence for retrieving the JDBC database connection from the data source.

Table 273. Database username property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Delivery type (DeliveryType)

This property specifies the order in which events are delivered by the adapter to the export.

Table 274. 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.
Globalized	No
Bidi supported	No

Encoding used by FTP server property (EISEncoding)

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

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

- When both EISEncoding at the adapter level and EISEncoding at the activation specification level are set, the value at the activation specification level takes precedence.

Specify any Java-supported encoding set for this attribute.

Table 275. Encoding used by FTP server property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery data source (JNDI) name property (EP_DataSource_JNDIName)

JNDI name of the data source used by event persistence to get the JDBC database connection. The data source must be created in WebSphere Process Server. The database name specified while creating the data source must exist.

Table 276. Event recovery data source (JNDI) name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Event recovery table name property (EP_EventTableName)

Name of the table that is used by the adapter for event persistence. When using multiple activation specifications, this value must be unique for each. The same table name must not be used by other instances of same adapter or a different adapter. If the table does not exist in the database, the adapter will create the table.

Table 277. Event recovery table name property characteristics

Required	No
Default	`FTPTABLE`
Property type	String
Globalized	Yes

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Accepts either active or passive settings.

Table 278. FTP server connection mode property characteristics

Required	No
Default	`active`
Property type	String
Globalized	No

FTPS connection mode property (ftpsConnectionMode)

This property is used to specify the connection mode when establishing a connection with the FTPS server. The WebSphere Adapter for FTP now supports both Implicit and Explicit connection modes. This property is used when you select either FTP over secure sockets layer (SSL) protocol or FTP over transport layer security (TLS) protocol.

Table 279. FTPS connection mode property characteristics

Required	No
Possible values	`Explicit` `Implicit`
Default	`Explicit`
Property type	String
Usage	This property represents the mode used to connect to the FTPS server. When this property is set to: • Explicit connection mode, initially the connection is established as a normal FTP connection. To send sensitive information, such as password the adapter switches to a secure FTP connection by issuing an `AUTH` command. **Note:** The default port for Explicit connection mode is 21. • Implicit connection mode, the connection is established as a secure FTP connection. All communications between the adapter and the server continues in a secure mode. There is no exchange of clear text information between the Adapter and the server. **Note:** The default port for Implicit connection mode is 990.
Globalized	No
Bidi supported	No

Failure file extension for local archive property (FailedArchiveExt)

File extension used to archive business objects in the event file that are not successfully processed. This property is used only when LocalArchiveDirectory is valid and exists.

Table 280. Failure file extension for local archive property characteristics

Required	No
Default	`fail`
Property type	String
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used to read the event files based on the EndBODelimiter property and during string to byte[] conversions. If not specified, the adapter attempts to read without any specific encoding. You can specify any Java supported encoding set.

Table 281. File content encoding property characteristics

Required	No
Default	None
Property type	String

Table 281. File content encoding property characteristics (continued)

Globalized	No

File extension for remote archive property (ftpRenameExt)

File extension or suffix that the adapter uses to rename the remote FTP file after the connector has polled for it. Renaming the file prevents the connector from polling the same file in the next poll cycle. The adapter can be configured to rename the processed event file and move it to an archive directory.

Table 282. File extension for remote archive property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Keystore file property (keyStorePath)

This property specifies the path of the keystore that contains the private key entries.

Table 283. Keystore file property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the absolute path of the keystore file on the adapter machine (on which the adapter is running). The keystore file contains the private key entry of the FTPS client. It is also accompanied by a certificate chain for the corresponding public key. The keystore data is used to authenticate the clients identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Keystore password property (keyStorePassword)

This property specifies the password that is used to encrypt the keystore.

Table 284. Keystore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the keystore. It is used to check the integrity of the keystore data. If the value is not specified, integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Key password property (keyPassword)

This property specifies the password that is used to encrypt the key.

Table 285. Key password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password of the key that is used to recover the key from the keystore. The property is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Keystore type property (keyStoreType)

This property specifies the type of keystore.

Table 286. Keystore type property characteristics

Required	No
Possible values	JKS and PKCS12
Default	JKS
Property type	String
Usage	This property specifies the type of the keystore. It is applicable only if you select FTP over SSL or FTP over TLS as the protocol. This property is also applicable for the type of the truststore.
Globalized	No
Bidi supported	No

Truststore file property (trustStorePath)

This property specifies the path of the truststore file that contains the certificates of the FTPS servers trusted by the adapter.

Table 287. Truststore file property characteristics

Required	This property is required only if you set the protocol as FTP over SSL or FTP over TLS
Default	None
Property type	String
Usage	This property specifies the absolute path of the truststore file on the adapter machine (on which the adapter is running). The truststore file contains the certificates of FTPS servers trusted by the adapter and is used to authenticate the servers identity while establishing a SSL connection.
Globalized	No
Bidi supported	No

Truststore password property (trustStorePassword)

This property specifies the password of the truststore.

Table 288. Truststore password property characteristics

Required	No
Default	None
Property type	String
Usage	This property specifies the password for the truststore. It is used to check the integrity of the truststore data. If the value is not specified, the integrity check will not be executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS.
Globalized	Yes
Bidi supported	No

Time interval for polling unchanged files (fileUnchangedTimeInterval)

This property specifies the time interval for the adapter to monitor the files for any updates in the content. The adapter polls only those files that are not changed during the specified time interval.

Table 289. Time interval for polling unchanged file

Required	No
Default	0
Unit of measure	Milliseconds
Property type	Integer
Usage	This property enables the adapter to poll 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. If the value is set to '0' the adapter polls the files instantly and does not check if the files are being modified.
Globalized	No
Bidi supported	No

Pass only file name and directory, not the content property (FilePassByReference)

Specifies that the file content of the event file is not sent to the export.

If set to `true`, the file is appended with a timestamp and sent to the LocalArchiveDirectory. The timestamp prevents errors and overwrites to the file when another file with the same name is received. This property can be set to `true` only when the LocalArchiveDirectory property is set and the specified directory exists. This property is used only for PassThrough inbound processing. When enabled, the file is not split into chunks.

Table 290. Pass only file name and directory, not the content property characteristics

Required	No

Table 290. Pass only file name and directory, not the content property characteristics (continued)

Default	`false`
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during inbound processing. Accepts either ASCII or binary.

Table 291. File transfer type property characteristics

Required	No
Default	`binary`
Property type	String
Globalized	no

Number of files to get at a time property (ftpGetQuantity)

Determines the number of files retrieved from the remote FTP URL with each remote poll.

Table 292. Number of files to get at a time property characteristic

Required	Yes
Default	`10`
Property type	Integer
Globalized	No

Number of poll periods between downloads property (ftpPollFrequency)

Determines how frequently the adapter polls the FTP server, measured in the number of standard poll cycles. For example, if PollPeriod is set to 10000, and FTPPollFrequency is set to 6, the adapter polls the LocalEventDirectory every 10 seconds and polls the remote EventDirectory every 60 seconds. The adapter performs FTP polling only if you specify a value for this property. If PollPeriod is 0, you consider it as 1 for calculation. If the calculation evaluates to 0, the adapter does not perform FTP polling.

Table 293. Number of poll periods between downloads property characteristics

Required	Yes
Default	`5`
Property type	Integer
Globalized	No

Retry limit for failed events (FailedEventRetryLimit)

This property specifies the number of times that the adapter attempts to redeliver an event before marking the event as failed.

Table 294. 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

Run FTP script file after downloading files property (ftpScriptFileExecutedAfterInbound)

Specifies the path of the script file that will be executed after downloading the files from the FTP server.

Table 295. Run FTP script file after downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Run FTP script file before downloading files property (ftpScriptFileExecutedBeforeInbound)

Specifies the path of the script file that will be executed before downloading the files from the FTP server.

Table 296. Run FTP script file before downloading files property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Host name property (HostName)

Host name of the FTP Server to which the connection is established during inbound processing.

Table 297. Create Table property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Include business object delimiter in the file content property (IncludeEndBODelimiter)

When set to `true`, the delimiter is sent with the business object content for further processing. This property is valid only when splitting the event files based on a delimiter.

Table 298. Include business object delimiter in the file content property characteristics

Required	No
Default	`false`
Property type	String
Globalized	No

Local archive directory property (LocalArchiveDirectory)

Absolute path of the local Archive directory. The directory must be valid and exist.

Table 299. Local archive directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the local archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${LOCALARCHIVE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **LocalArchiveDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Local directory property (LocalEventDirectory)

Local system directory into which the adapter downloads event files from the FTP site. You must specify a value for this property to enable the adapter to process events.

Table 300. Local directory property characteristics

Required	Yes
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the local event directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${LOCAL_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **LocalEventDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Maximum connections (MaximumConnections)

This property specifies the maximum number of connections that the adapter can use for inbound event delivery.

Table 301. Maximum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. The adapter considers any positive entry less than 1 to be equal to 1. Typing a negative value for this property may result in run time errors.
Globalized	No
Bidi supported	No

Minimum connections (MinimumConnections)

This property specifies the minimum number of connections that the adapter can use for inbound event delivery.

Table 302. Minimum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. Any value less than 1 is treated as 1 by the adapter. Typing a negative value or 1 for this property may result in run time errors.
Globalized	No
Bidi supported	No

File extension for local archive property (originalArchiveExt)

File extension used to archive the original event file. This preserves the entire event file for reference in case any of its business objects fail. This property is used only when LocalArchiveDirectory is valid and exists.

Table 303. File extension for local archive property characteristics

Required	No
Default	original
Property type	String
Globalized	Yes

Password property (Password)

Password of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the password is included in the URL specified in the EventDirectory property.

Table 304. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password used to connect to event data source property (EP_Password)

The password used during event persistence to get the database connection from the data source.

Table 305. Password used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Interval between polling periods (PollPeriod)

This property specifies the length of time that the adapter waits between polling periods.

Table 306. 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

Table 306. Interval between polling periods details (continued)

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 307. 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
Bidi supported	No

Passphrase property (passPhrase)

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

Table 308. Passphrase property property characteristics

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

Port number property (PortNumber)

Port number of the FTP server through which the connection is established during inbound processing.

Table 309. Port number property characteristics

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

Private key file property (PrivateKeyFilePath)

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

Table 310. Private key property characteristics

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

Protocol property (Protocol)

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

For example:

Normal connection: `FTP`

FTP over SSL connection: `FTPS_SSL`

FTP over TLS connection: `FTPS_TLS`

FTP over SSH connection: `SFTP`

Table 311. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Remote archive directory property (ftpArchiveDirectory)

Relative path of the archive directory on the FTP server. The directory must exist. There are several options for using this property to specify archiving:

- Specifying a value for this property, but no value for the FTPRenameExt property causes the adapter to append a timestamp to the event file name and move it to the FTP server archive directory specified in this property.
- Specifying a value for this property and the FTPRenameExt property causes the adapter to rename the processed event file name with a timestamp and the value specified in FTPRenameExt and moves it to the FTP server archive directory specified in this property.
- Specifying no value either for this property or the FTPRenameExt property causes the adapter to delete the processed event file without archiving it.

- Specifying no value for this property but specifying a value for the FTPRenameExt property causes the adapter to rename the processed event file, adding a timestamp and the value specified in FTPRenameExt.

The value of remote archive directory property accepts both the absolute and relative paths of the directory. If the value does not start with a forward slash, the adapter considers the path to be relative to your home directory.

Table 312. Remote archive directory property characteristics

Required	No
Default	None
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the remote archive directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${REMOTEARCHIVE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. The archive directory located on the FTP server and used in inbound configuration represents the absolute path of the archive directory. It does not contain any host name or URL information. This directory is located on the same FTP server where the Event Directory is located, for example: /home/archive. **Note:** The **FTPArchiveDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Remote directory property (EventDirectory)

Remote directory of the FTP server from where the event files are retrieved for inbound processing. If the value of Remote directory is set to <HOME_DIR>, the adapter polls for event files in the user's home directory.

The value of event directory property accepts both the absolute and relative paths of the directory. If the value does not start with a forward slash, the adapter considers the path to be relative to the user's home directory.

Table 313. Remote directory property characteristics

Required	Yes
Default	<HOME_DIR>
Property type	String
Usage	You can use a WebSphere Application Server environment variable to represent the remote directory. Specify the name of the environment variable in braces, preceded by a $ symbol. For example: `${REMOTE_DIRECTORY}`. See the topic on http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp?topic=/com.ibm.wsadapters.jca.ftp.doc/doc/tbp_ftp_defineenvironvars.html in this documentation. **Note:** The **EventDirectory** must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Retrieve files with this pattern property (EventFileMask)

Filter for the event files. The file filter is a well-qualified expression consisting of alphanumeric characters and the * and ? wild cards.

Table 314. Retrieve files with this pattern property characteristics

Required	Yes
Default	`*.*`
Property type	String
Globalized	Yes

Enable remote verification property (enableRemoteVerification)

When a client connects to the FTP server, two kinds of connections or channels are established; a command connection (also known as control connection), and a data connection. The command connection is the one through which the FTP commands are sent (and replies to these commands received) to the server and the data connection is the channel through which the data transfer takes place between the client and the server.

This property is used to verify if the host system requesting the data transfer to or from the FTP server is the same host system on which the adapter is running.

The verification is done while establishing a data connection to perform data transfer.

Note: This property is applicable only to FTP and FTPS protocols.

Table 315. Enable Remote verification property characteristics

Required	No
Possible values	`True` `False`
Default	`True`
Property type	Boolean
Usage	This property verifies if the data connection and the control connection are from the same host system. By default, the remote verification property is set to TRUE by the FTP server. When this property is set to: • True, during run time, the adapter checks if the data connection is established with the same host as the control connection. If the data connection is established from a different host than the control connection, then an exception is thrown and the connection fails. • False, remote verification is not performed. **Note:** Disabling the remote verification leads to low security. Precaution must be taken before disabling the remote verification.
Globalized	No
Bidi supported	No

Retry EIS connection on startup (RetryConnectionOnStartup)

This property controls whether the adapter attempts to connect again to the FTP server if it cannot connect at startup.

Table 316. 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 FTP server 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 FTP server, for example, when you are building and testing the application that receives events from the adapter. If the adapter cannot connect, the adapter writes log and trace information and stops. The administrative console shows the application status as `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 establish a new connection.

Table 317. 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

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 318. Number of times to retry the system connection details

Required	No
Possible values	0 and positive integers
Default	0
Property type	Integer
Usage	This property controls how many times the adapter retries the connection if the adapter cannot connect to the FTP server to perform inbound processing. A value of 0 indicates an infinite number of retries. To control whether the adapter retries if it cannot connect to the FTP server when it is first started, use the RetryConnectionOnStartup property.
Globalized	No
Bidi supported	No

Enable server verification property (EnableServerVerification)

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

Table 319. Enable server verification property details

Required	No
Possible values	True False
Default	False
Property type	Boolean
Usage	When this property is set to: • True, server authentication is enabled • False, server authentication is disabled The adapter checks for the HostKeyFile property in the path of the file that contains the host keys of the trusted servers.
Globalized	Yes
Bidi supported	No

Host key file property (HostKeyFile)

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

Table 320. Host key file property characteristics

Required	This property has to be specified if the EnableServerVerification property is enabled.
Default	None
Property type	String

Table 320. Host key file property characteristics (continued)

Usage	This is used by the adapter to verify the host key of the remote server with the host keys of the trusted servers specified in this file.
Globalized	Yes
Bidi supported	No

Host name property (SocksProxyHost)

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

Table 321. Host name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Password property (SocksProxyPassword)

Password used to authenticate the proxy server.

Table 322. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Port number property (SocksProxyPort)

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

Table 323. Port number property characteristics

Required	No
Default	`1080`
Property type	Integer
Globalized	No

User name property (SocksProxyUserName)

User name used to authenticate the proxy server.

Table 324. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Sort event files property (SortEventFiles)

Determines the sorting order of event files being polled. Supported values are:

- by file name – sort ascending on file name
- by time stamp – sort ascending on last modified timestamp
- no sort – not sorted

Event file ordering from which events need to be delivered is valid only if the activation specification DeliveryType property is set to ORDERED. File name sorting is provided based on the locale of the FTP server. The ICU4J package is used to track the locales and their corresponding rules.

Table 325. Sort event files property characteristics

Required	No
Default	no sort (= not sorted)
Property type	String
Globalized	No

Specify criteria to split file content property (SplitCriteria)

This property takes different values based on the value of the SplittingFunctionClassName property. For example: To specify that a file is to be split every 5 KB, set the SplitCriteria property to 5000.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
  - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
  - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=0, chunking is disabled.

When FilePassByReference is enabled during inbound PassThrough, the event file is not split.

Note: For input files that contain multiple COBOL copybook records, in order to enable file splitting by size you must provide the correct length of each record. To determine the size of each record, use one of these methods:

1. Open the Business Object in a text editor.
   a. For example:

```
<element name="CustomerNumber">
<annotation>
<appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
<td:typeDescriptorElement>
<td:initialValue kind="SPACE"/>
<td:simpleInstanceTD accessor="readWrite" attributeInBit="false"
contentSize="5" offset="0" size="5">
<td:sharedType>
<td:stringTD addrUnit="byte" alignment="byte" characterSize="1"
lengthEncoding="fixedLength" paddingCharacter=" "
prefixLength="0" width="5"/>
</td:sharedType>
</td:simpleInstanceTD>
```

```
</td:typeDescriptorElement>
</appinfo>
</annotation>
<simpleType>
   <restriction base="string">
    <maxLength value="5"/>
  </restriction>
 </simpleType>
</element>
```

Each element in the business object has a corresponding `<element>` entry.

b. Look for a restriction tag for each element tag (the COBOL data binding requires a fixed-width data handler).

c. Add up the `maxLength` attribute values for each of the elements. In this example, the value is 5. The sum of the `maxLength` values is the size of each record of type DFHCOMMAREA.

2. Open the Business Object in a text editor.

   a. Look for the complex type tag with the business object name value in the name attribute. In the example that follows, the business object name is DFHCOMMAREA.

   b. Locate a namespace-appended tag called `aggregateInstanceTD` and use the value for the attribute `contentSize`. In this example, the value is 117. This is the size of each record of type DFHCOMMAREA.

```
<complexType name="DFHCOMMAREA">
<annotation>
<appinfo source="http://www.ibm.com/cam/2005/typedescriptor">
<td:typeDescriptorCT>
<td:aggregateInstanceTD accessor="readWrite" attributeInBit="false"
contentSize="117" offset="0" size="117">
```

Table 326. Specify criteria to split file content property characteristics

Required	No
Default	`0`
Property type	String
Globalized	Yes

Splitting function class name property

This value takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The `com.ibm.j2ca.utils.filesplit.SplitByDelimiter` class that splits the event file based on delimiter.
- The `com.ibm.j2ca.utils.filesplit.SplitBySize` class that splits the event file based on the event file size.

Optionally, you can provide a custom file splitter class and use it by inputting the class name into the SplittingFunctionClassName property.

The delimiter or file size is provided in the SplitCriteria property. If the EventContentType property is set to null, it is automatically set to a class name that performs splitting based on file size.

Table 327. Splitting function class name property characteristics

Required	No

Table 327. Splitting function class name property characteristics (continued)

Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Globalized	No

Stop the adapter when an error is encountered while polling (StopPollingOnError)

This property specifies whether the adapter will stop polling for events when it encounters an error during polling.

Table 328. Stop the adapter when an error is encountered while polling details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If this property is set to `True`, the adapter stops polling when it encounters an error. If this property is set to `False`, the adapter logs an exception when it encounters an error during polling and continues polling.
Globalized	No
Bidi supported	No

Success file extension for local archive property (SuccessArchiveExt)

File extension used to archive all the successfully processed business objects. This property is used only when LocalArchiveDirectory is valid and exists. For example, `12345.order > 12345.order.success`

Table 329. Success file extension for local archive property characteristics

Required	No
Default	`success`
Property type	String
Globalized	Yes

User name property (UserName)

Name of the user who has privileges to connect to the FTP server and perform FTP operations. You do not need to specify a value for this property if the user name is included in the URL specified in the EventDirectory property.

Table 330. User name property characteristics

Required	No
Default	None
Property type	String

Table 330. User name property characteristics (continued)

Globalized	Yes

User name used to connect to event data source property (EP_UserName)

User name used by event persistence for getting the database connection from the data source.

Table 331. User name used to connect to event data source property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Rule editor to filter files

This property is used to filter event files based on a set of rules

Table 332. Rule editor to filter files

Required	Optional
Default	None
Property type	String
Usage	During an inbound processing, if the value in the rule table is specified, then the event files are fetched after filtering, based on the specified rules before polling those event files.
Globalized	Yes
Bidi supported	No

Wrapper and interaction specification properties

Wrapper properties are attributes of the wrapper business object that enable an application programmer to control an operation for the business objects in a wrapper. Interaction specification properties control the interaction for an operation for the entire adapter.

The external service wizard sets the interaction specification properties when you configure the adapter. You can change some, but not all, of these properties. However, you can change some properties for outbound operations. Use the assembly editor to change these properties, which reside in the method binding of the import. You set the wrapper properties using the WebSphere Integration Developer test client or programmatically at run time.

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

Table 333. Interaction specification properties

Property name		Description
In the wizard	In the wrapper business object	

Table 333. Interaction specification properties (continued)

Remote archive directory for retrieve operation	ArchiveDirectoryForRetrieve	The adapter optionally archives the file to this folder before it is deleted during a Retrieve operation.
Create new file if the file does not exist	CreateFileIfNotExists	If the file does not exist on the FTP server, the adapter creates the file when this property is set to `true` during Append and Overwrite operations.
FTP server connection mode	DataConnectionMode	Data connection mode used by the FTP server during file transfers.
Delete the file after retrieve operation	DeleteOnRetrieve	The adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.
Remote directory on FTP system	DirectoryPath	Absolute path of the directory on the FTP server where the outbound operation must be performed.
"Data channel protection level (dataProtectionLevel)" on page 192	dataProtectionLevel	Specifies the protection level of a data channel in case of FTPS protocol.
File content encoding	FileContentEncoding	Encoding used while writing to the file.
File in local directory	FileInLocalDirectory	If set to `true` during a create operation, the file content is picked from the local directory path of the adapter workstation.
Default target file name	Filename	Name of the file in the directory provided by the DirectoryPath property.
File transfer type	FileTransferType	File transfer type used during outbound operations.
Generate a unique file	GenerateUniqueFile	The adapter creates a unique file name when this property is set to `true`.
Host name property	SecondServerHostName	Host name of the second FTP server.
Delimiter between business objects in the file property	IncludeEndBODelimiter	File content is appended with this value.
Local archive directory for create operation	LocalArchiveDirForCreate	When `LocalArchivingEnabledForCreate` is set to `true` during a create operation, the file is saved to the local workstation in this directory.
Archive file in the local directory for create operation	LocalArchivingEnabledForCreate	When set to `true`, the file is saved to the local workstation during a create operation.
Local directory	LocalDirectoryPath	The file is picked from this directory.
(Not available)	ResumeFailedTransfer	When this property is set to `true` during a create operation, the adapter resumes the transfer of files from the point at which the transfer of file was interrupted due to connection error.
Port number	SecondServerPortNumber	Port number of the second FTP server.
Protocol	SecondServerProtocol	Specifies the protocol used to connect to the second server.
Script File Parameters	ScriptFileParameters	The parameters required by the FTP script file.
Directory	SecondServerDirectory	Directory path of the second FTP server during a ServerToServerFileTransfer operation.
Password	SecondServerPassword	Password of the second FTP server during a ServerToServerFileTransfer operation.

Table 333. Interaction specification properties (continued)

User name	SecondServerUsername	User name of the second FTP server during a ServerToServerFileTransfer operation.
Specify criteria to split file content	SplitCriteria	The delimiter that separates the business objects in the event file.
Split function class name	SplittingFunctionClassName	The fully qualified class name of the class file to be used to enable file splitting.
Staging directory	StagingDirectory	The file is first created into this directory.
Temporary file name	TemporaryFilename	Specifies the temporary file name for the create operation.

Archive file in the local directory for create operation property (LocalArchivingEnabledForCreate)

During outbound create operations, when the file content is coming as part of the business object from a J2EE application and this property is set to `true`, the file is saved to the local workstation in the LocalArchiveDirForCreate directory before performing the outbound operation.

Table 334. Archive file in the local directory for create operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Create new file if the file does not exist property (CreateFileIfNotExists)

During outbound Append and Overwrite operations, if the file does not exist on the FTP server, the adapter creates the file when this property is set to `true`. If this property is `false` and file does not exist, the adapter sends an error.

Table 335. Create new file if the file does not exist property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Data channel protection level (dataProtectionLevel)

This property specifies the protection level of the data transferred over the data channel. It specifies the type of data channel protection that the FTP adapter and the server use.

Protection Buffer Size (PBSZ) and Data Channel Protection level (PROT) commands are issued by the FTP adapter before opening a data channel to specify the protection level on the data channel. By default, the adapter issues the “PBSZ 0” command before issuing the PROT command.

Table 336. Data channel protection level property characteristics

Required	No

Possible values	`Private - Data is transferred in encrypted form`
	`Clear - Data is transferred as clear text`

Default	`Private - Data is transferred in encrypted form`
Property type	String
Usage	This property is used for selecting the protection level for the data channel. Following are the protection values: • Private – Indicates that the data transfer will be integrity and confidentiality protected. • Clear – Indicates that the data channel will carry the raw data of the file transfer between the FTP adapter and the server without any security.
Globalized	No
Bidi supported	No

Delete the file after retrieve operation (DeleteOnRetrieve)

During an outbound Retrieve operation, the adapter deletes the file from the FTP server after it is retrieved when this property is set to `true`.

Table 337. Delete the file after retrieve operation property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

Default target file name property (Filename)

Name of the file to be used during outbound operations.

Table 338. Default target file name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Delimiter between business objects in the file property (IncludeEndBODelimiter)

File content is appended with this value. Used during the outbound create, append, and overwrite operations.

Table 339. Include business object delimiter in the file content property characteristics

Required	No

Table 339. Include business object delimiter in the file content property characteristics (continued)

Default	For the create and overwrite operations, no default value is set. For the append operation, the default value is `<EndBO>`. For the append operation, the following rules apply: • If the delimiter is set to `null` in the business object wrapper, no delimiter is used to separate the business objects. • If the IncludeEndBODelimiter property is not set in the business object wrapper, and the value in the interaction specification is also `null`, the default is `<EndBO>`. • If a specific delimiter value is specified in the business object wrapper, the specified value will be appended. • If both the business object wrapper and the interaction specification have specified values, the business object wrapper value takes precedence.
Property type	String
Globalized	Yes

Directory property (SecondServerDirectory)

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

Table 340. Directory property characteristics

Required	No
Default	None
Property type	String
Usage	For interaction specification properties, the directory located on the FTP server and used in outbound operation represents the absolute path of the FTP directory. For example: `/home/usr/output`. It does not contain any host name or URL information. For wrapper business object properties, the URL of the second server to which the ServerToServerFileTransfer outbound operation is performed. For example: The syntax for specifying the FTP URL is: `ftp://[UserId:password@]FTPserver[:port]/DirectoryForSecondServer`.
Globalized	Yes

File content encoding property (FileContentEncoding)

Encoding used while writing to the file. If this property is not specified, the adapter tries to read without using any specific encoding. You can specify any Java supported encoding set.

Table 341. File content encoding property characteristics

Required	No
Default	None
Property type	String
Globalized	No

File in local directory property (FileInLocalDirectory)

During outbound create operations, if this property is set to `true`, the file content is not available in the business object. The file is retrieved from the local directory on the adapter workstation. During outbound retrieve operations, if this property is set to `true`, the file content is not sent to the J2EE application as part of the business object. The file is saved to the local directory of the adapter workstation.

Table 342. File in local directory property characteristics

Required	No
Default	`false`
Property type	Boolean
Globalized	No

File transfer type property (FileTransferType)

File transfer type used during outbound operations. Takes either ASCII or binary.

Table 343. File transfer type property characteristics

Required	No
Default	`binary`
Property type	String
Globalized	No

FTP server connection mode property (DataConnectionMode)

Data connection mode used by the FTP server during file transfers. Takes either `active` or `passive`. This value is used only when a file transfer is taking place. This property is not used when performing a server to server file transfer outbound operation.

Table 344. FTP server connection mode property characteristics

Required	No
Default	`active`
Property type	String
Possible values	`active` or `passive`
Globalized	No

Generate a unique file (GenerateUniqueFile)

During outbound Create operation, the adapter creates a unique file name when this property is `true`. The adapter ignores any value that is set for the Filename property when this property is set to `true`.

Note: The adapter does not support both GenerateUniqueFile and StagingDirectory options at the same time.

Table 345. Generate unique file property characteristics

Required	No
Default	`false`

Table 345. Generate unique file property characteristics (continued)

Property type	Boolean
Globalized	No
Restrictions	The FTP server must support RFC1123 to use this feature.

Host name property (SecondServerHostName)

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

Table 346. Host name property characteristics

Required	Yes
Default	None
Property type	String
Globalized	Yes

Local archive directory for create operation property (LocalArchiveDirForCreate)

During outbound create operations, when the file content is coming as part of the business object and LocalArchivingEnabledForCreate is set to `true`, the file is saved to the local workstation in this directory.

Table 347. Local archive directory for create property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalArchiveDirForCreate directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Local directory property (LocalDirectoryPath)

During outbound create operations, when FileInLocalDirectory property is set to `true`, the file content is not available in the business object. Instead the file is picked from this directory. During outbound retrieve operations, when FileInLocalDirectory property is set to `true`, the file content is not sent as part of business object. The file is saved to this directory.

Table 348. Local directory property characteristics

Required	No
Default	None
Property type	String
Usage	The LocalDirectoryPath directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Port number property (SecondServerPortNumber)

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

Table 349. Port number property characteristics

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

Protocol property (SecondServerProtocol)

Protocol that is used to establish a connection to the second server. The FTP protocol is used in establishing the connection.

Table 350. Protocol property characteristics

Required	Yes
Default	`FTP`
Property type	String
Globalized	No

Password property (SecondServerPassword)

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

Table 351. Password property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Remote archive directory for retrieve operation property (ArchiveDirectory ForRetrieve)

During an outbound Retrieve operation, the adapter optionally archives the file to this folder before it is deleted. The archive directory must exist.

Table 352. Remote archive directory for retrieve operation property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Remote directory on FTP system property (DirectoryPath)

Absolute path of the directory on the FTP server where the outbound operation must be performed for all operations except ExecuteFTPScript, or the directory path on the local adapter workstation for the ExecuteFTPScript operation only. The directory must exist.

Note: If the value <HOME_DIR> is specified as the DirectoryPath, the outbound operations will be performed in the user's home directory.

Table 353. Remote directory on FTP system property characteristics

Required	No
Default	None
Property type	String
Usage	The DirectoryPath directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

ResumeFailedTransfer

This property supports resuming the transfer of files, which were interrupted due to an error in connection to the FTP server.

Note: This property is applicable only to outbound processing.

Table 354. ResumeFailedTransfer property characteristics

Required	No
Default	`false`
Property type	Boolean
Usage	During a create operation, when this property is set to `true`, the adapter resumes the transfer of files from the point at which the transfer of file was interrupted due to an error in connection.
Globalized	No

Script File Parameters property (ScriptFileParameters)

During an outbound ExecuteFTPScript operation, the parameters required by the FTP script file are set in this property. During run time, the adapter replaces the parameters with these values.

Table 355. Script File Parameters property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

Specify criteria to split file content property (SplitCriteria)

This property accepts different values based on the value of the SplittingFunctionClassName property.

- If the SplittingFunctionClassName property specifies that files are split based on a delimiter, then SplitCriteria contains the delimiter that separates the business objects in the event file.
- If SplittingFunctionClassName is set to a value which does splitting based on size, then the SplitCriteria property contains a valid number that represents the size in bytes.
  - If the event file size is greater than this value, the adapter splits the file into chunks of this size and the chunks are posted.
  - If the event file size is less than this value, the entire event file is posted. When SplitCriteria=`0`, chunking is disabled.

Table 356. Specify criteria to split file content property characteristics

Required	No
Default	`0`
Property type	String
Globalized	Yes

Split function class name property (SplittingFunction ClassName)

Takes the fully qualified class name of the class file to be used to enable file splitting. Requires two values:

- The com.ibm.j2ca.utils.filesplit.SplitByDelimiter class that splits the event file based on delimiter.
- The com.ibm.j2ca.utils.filesplit.SplitBySize class that splits the event file based on the event file size.

The delimiter or file size is provided in the SplitCriteria property.

Table 357. Split function class name property characteristics

Required	No
Default	`com.ibm.j2ca.utils.filesplit.SplitBySize`
Property type	String
Globalized	No

Staging directory property (StagingDirectory)

During outbound create operations, the file will be created in this directory first. When the file creation is complete, the file is copied to the directory specified in the DirectoryPath property. This staging directory is also used for Append and Overwrite operations where the specified file is copied to the StagingDirectory, if present. The appended or overwritten content is then moved back to the original specified directory. If StagingDirectory is not specified, the operation is run in the actual required directory.

Note: The adapter does not support both StagingDirectory and GenerateUniqueFile options at the same time.

Table 358. Staging directory property characteristics

Required	No
Default	None
Property type	String
Usage	The StagingDirectory directory must be created manually, on the machine where the adapter runs, before the adapter is started, as the adapter does not create this directory automatically.
Globalized	Yes

Temporary file name property (TemporaryFilename)

This property specifies the temporary file name for the create operation. After successful creation of the file, the file gets renamed to the value specified in the 'Default target file name' property.

Table 359. Temporary file name property characteristics

Required	No
Possible values	All valid file names
Default	None
Property type	String
Usage	This property is used in the create operation. If the temporary file name is specified, the file is created with the temporary file name. After the file is successfully created, the file is renamed to the value that is specified in the 'Default target file name' property.
Example	xyz.tmp
Globalized	No

User name property (SecondServerUsername)

User name of the second FTP server to which the file is transferred during a server to server file transfer outbound operation.

Table 360. User name property characteristics

Required	No
Default	None
Property type	String
Globalized	Yes

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).

Printed in USA