



| IBM Software Group

WebSphere Adapters V6.1

IBM WebSphere Adapter for FTP V6.1



@business on demand.

© 2008 IBM Corporation
Converted to video June 23, 2015

This presentation will focus on the WebSphere® Adapter for FTP V6.1.

Agenda

- Overview
- Enterprise metadata discovery
- Business object structure
- Outbound
- Inbound
- Problem determination
- Summary

This section will provide an overview of the WebSphere Adapter for FTP.

Overview: WebSphere Adapter for FTP

- IBM WebSphere Adapter for FTP
 - ▶ Implements the Java™ 2 Enterprise Edition (J2EE) Connector Architecture (JCA), version 1.5 specification
 - ▶ Enables bi-directional connectivity
 - ▶ Events and responses captured as files on the file system
 - ▶ Supports transformation of data to xml
 - When data transformation is involved the business objects are pre-generated, otherwise a fixed structure is used.
 - ▶ Supports transfer of files using SSL
 - ▶ Enhanced in V6.1 to support Federal Information Processing Standards (FIPS)



The IBM WebSphere Adapter for FTP implements the JCA 1.5 specification. It enables bi-directional connectivity, both inbound and outbound, with those Enterprise Information System business applications that can communicate only through files. Events and responses are captured as files on the remote file system. The WebSphere Adapter for FTP supports the same business object structure for both inbound and outbound. The file is seen as a byte array in the business object for pass-through and user-defined object embedded in the business object for user-defined. The adapter supports secure transfer of data using SSL.

Configuration steps

- **Discovery:** Discovering EIS metadata and automatic creation of components to access the EIS
- **Development:** Create application which make use of the discovered components
- **Enablement:** Configuring the runtime with the location of the EIS provider jars/native libraries
- **Administration:** Deploy all required components to runtime, and administer them



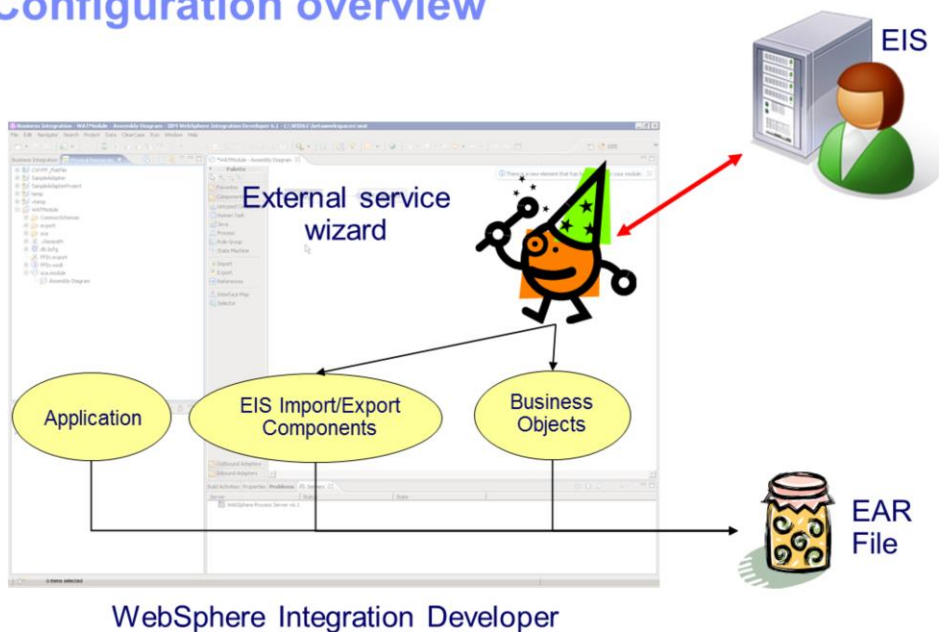
This slide summarizes the steps involved in using adapter as part of your application and how to administer them. At a high level the interaction steps can be broken down into four steps. The first step is the Discovery phase where you use the external service wizard to configure the adapter and generate the necessary artifacts. The next step is the development phase where you use the generated artifacts to create your application. The third step is the Enablement step where you specify the necessary dependency files required at runtime by the adapter. For example the SAP adapter uses the sapjco.jar file to communicate with the SAP system. The last step is to create the Enterprise Broker Archive (EAR) file, deploy it to the runtime and administer the application. For the FTP adapter, there is no enablement step as it requires no dependencies.

Section

Enterprise metadata discovery

This section will provide an overview of the discovery process for WebSphere Adapter for FTP.

Configuration overview



This slide depicts the steps involved in configuring the adapters and creating a deployable application. WebSphere Integration Developer provides an external service wizard that connects to the EIS, introspects and imports the metadata information and creates the necessary artifacts. The artifacts generated by the wizard are the import or export components and the business object definitions. The interaction style you choose when configuring using the wizard defines the generation of either an import or an export component. The supported interaction styles are inbound and outbound. Once the artifacts are created for you, you can use these generated artifacts as part of your application flow. The finished application project is exported as an ear file that you deploy to the runtime.

Enterprise service discovery

- Enables the generation of business object definitions and other artifacts required by SCA
- FTP file business object
 - ▶ Pre-defined structure for non Data Transformation Framework (DTF) flows
 - ▶ Adapter is unaware of the content of the file for non Data Transformation Framework (DTF) flows.
 - ▶ Adapter can parse files based on a configured delimiter.
 - ▶ Adapter can use pre generated business object structures.
- Service discovery for FTP is used
 - ▶ To specify activation specification and managed connection factory properties as input to build the service description
 - ▶ To specify any custom adapter properties

Enterprise Service Discovery in WebSphere Integration Developer implements the Enterprise MetaData Discovery Specification. The Enterprise Service Discovery wizard steps you through configuration of the adapter properties, service descriptions, and business object discovery, and results in the generation of the artifacts required for integration with SCA applications. With the WebSphere Adapter for FTP, there is no need for “discovery” of business objects; there is a pre-defined structure for the business objects for both inbound and outbound processing. The adapter views the file as the data, and does not look inside the contents of the file for pass through. Therefore; service discovery for the FTP Adapter is used mainly to specify values for the properties used in activation specification, managed connection factory, and adapter properties.

Discovery steps - Outbound

- Provide connection properties
- Chose to deploy connector along with module or separate
- Specify operations
 - ▶ Select operation type
 - ▶ Select data type(pass-through or user-defined)
 - Pass-through with business graph
 - Pass-through without business graph
 - user-defined
 - ▶ Configure operation
 - Specify operation name
 - Specify data binding class – FTPFileBaseDataBinding
 - Specify data handler – Needed only for user-defined. XML Data Handler
 - Specify Interaction specification properties

This slide summarizes the important steps involved in the discovery process using the external service wizard for outbound interaction style. For ftp adapter, you provide the user name, password and ftp server details. You are prompted for the output directory where the adapter places the processed data. You will also be prompted to specify if you want to deploy the connector module along with the ear file or if you want to deploy it separately. More information about deploying the connector RAR file by itself is covered in the overview presentation.

The next step involves the configuration of operations supported by the adapter one at a time. With V6.1 it is not mandatory to create a business graph. You can just choose to create a business object with out the graph. The “Data type for the operation” field provides you the option of creating the business object definitions with business graph, without business graph, and to define your own data type. The first two options deal with pass-through data, where the adapter does not do any transformation of the data. The user-defined option is used when you want the adapter to convert the data in the business object to a different format (for example, to XML).

The next panel in the wizard prompts you for the operation name, the data binding, and the data handler. FTPFileBaseDataBinding is the only supported data binding. You need to specify the data handler only if you need the adapter to transform the data coming in the business object. The only supported handler is the XML Data handler.

Another major enhancement in V6.1 is the ability to specify properties at interaction specification level for each operation. For example, for your create operation, if you always want to create files in one specific folder, you can specify the output directory at the interaction specification level. This will take precedence over any value you provided at the initial connection properties screen. Also, you do not have to provide any value for the output directory at the business object level in your application.

Discovery steps - Inbound

- Provide connection properties
 - ▶ Activation spec properties
 - ▶ Chose to deploy connector along with module or separate
- Specify function selector – map events to export function name
 - ▶ FilenameFunctionSelector
 - Rule based function selector.
 - ▶ For user-defined – use EmbeddedFunctionSelector
- Specify operations
 - ▶ Operation type is always emit
 - ▶ Select data type(pass-through or user-defined)
 - Business graph optional
 - ▶ Configure operation
 - Specify operation name
 - Specify data binding class - FTPFileBaseDataBinding
 - Specify data handler – Needed only for user-defined. XML data handler

This slide summarizes the important steps involved in the discovery process using the external service wizard for inbound interaction style. For the FTP adapter, there are no user name or password fields. You are prompted for the directory the adapter should poll for any files. You also specify if you want to deploy the connector module along with the ear file or if you want to deploy it separately. More information about deploying the connector RAR file by itself is covered in the overview presentation.

Another important configuration for inbound is the function selector. In Inbound, Function Selectors are required in order to map between events generated by resource adapters and the appropriate SCA export function name. There are two function selectors that are supported by the FTP adapter. These are FilenameFunctionSelector and EmbeddedNameFunctionSelector. The FilenameFunctionSelector is a rule-based function selector that can match a regular expression on a file name to an object name. This function selector is used for pass-through data or when the adapter cannot determine the business object name from the contents of the incoming file. The EmbeddedNameFunctionSelector is used in case of content-specific business objects, where the object name is embedded in the event file.

The next step involves the configuration of operations. With V6.1 you can have more than one operation defined for inbound interaction style. You can just choose to create a business object with out the graph. The “Data type for the operation” field provides you the option of creating the business object definitions with business graph, without business graph, and to define your own data type. The first two options deal with pass-through data where the adapter does not do any transformation of the data. The user-defined option is used when you want the adapter to convert the XML data in the polled file to be mapped to

the user-defined business object.

The next panel in the wizard prompts you for the operation name, the data binding and the data handler. `FTPFileBaseDataBinding` is the only supported data binding. You need to specify the data handler only if you need the adapter to transform the data coming in the business object. The only supported handler is the XML Data handler.

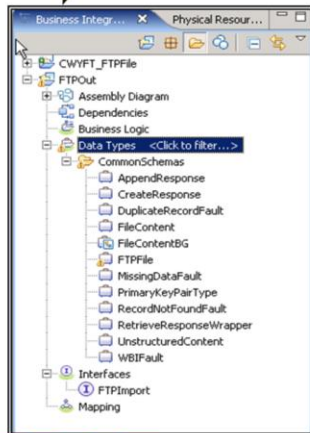
Section

Business object structure

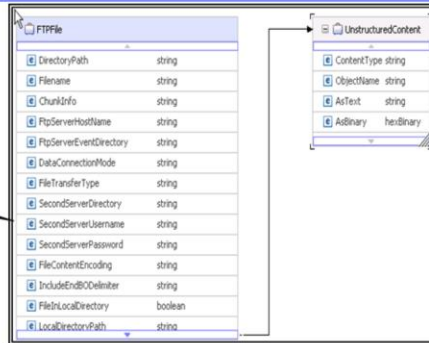
The next section will provide an overview of the business objects created by the external service wizard for pass-through and user-defined scenarios.

Business object structure

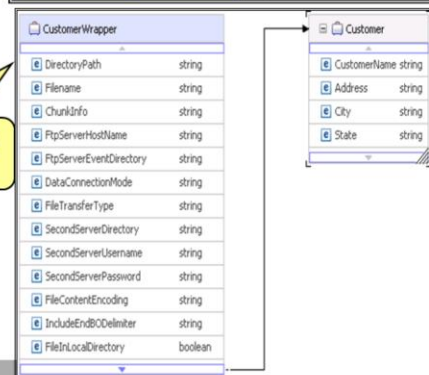
List of all business objects created for pass-through scenario



FTPFile business object with the protocol specific attributes



Wrapper object created for user-defined type



11

This slide shows the screen capture of the business objects created and the business object structure for both pass-through and user-defined scenarios. You can see the FTPFile business object with all the protocol specific information like directoryName, file name and so on. The Content attribute should be set to unstructured as the adapter just takes the value specified in the unstructured object and write it as is in the output file. The screen capture on the right side bottom shows an example for the user-defined type. The customer object that you imported is wrapped by a wrapper object with protocol specific information. The content attribute is set to the user-defined object you imported during the discovery process.

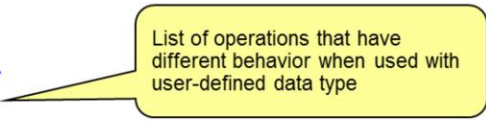
Section

Outbound

This section provides an overview of operations types supported for outbound and provides details on the attributes that define the behavior of each operation type

Outbound operations - FTP

- Outbound operations
 - ▶ Pass-through
 - ▶ User-defined
- List of operations
 - ▶ Create
 - ▶ Append
 - ▶ Overwrite
 - ▶ Retrieve
 - ▶ Delete
 - ▶ Exists
 - ▶ List
 - ▶ ExecuteFTPScript
 - ▶ ServerToServerFileTransfer



List of operations that have different behavior when used with user-defined data type

You can see the list of operations supported by the adapter for outbound on this slide. Create, append, retrieve and overwrite are the only operations that support data transformation.

Delete operation deletes the file that you specify and not the contents of the file for both user-defined and pass-through cases. Exists checks for the existence of a file in the specified directory and returns a Boolean value in the Existsresponse business object. List returns a list of all file names in the directory specified

ExecuteFTPScript operation provides the option to run a script file whose location and parameters are specified in the business object.

ServerToServerFileTransfer allows the transfer of files from one ftp server to another.

Outbound operations - Create

- Create – File with specified name created in output directory with the specified content. File name returned to indicate success
 - ▶ If file already exists, DuplicateRecordException is thrown
 - ▶ Staging directory used if specified
- Attributes
 - ▶ Generate Unique file
 - Available as Interaction spec property and as attribute of FTPFile business object or the user-defined business object wrapper
 - Generates unique file name with format "*ftpa<randomnumber>.tmp*"

This slide summarizes the behavior of the Create operation. A file with the specified file name is created in the specified directory with the content sent across in the business object. The file name is returned to the calling application indicating a successful response. If the file to be created already exists, a DuplicateRecordException is created.

This staging directory, if present, is used for 'create' operations where the specified file is written to the staging directory first, then moved to the original specified output directory. If the staging directory is not present, the file is directly written to the output directory.

The adapter will generate a unique file name when the property 'generateUniqueFile' is set to true. At this time the adapter ignores the value present in file name property. The name of the unique file generated by FTP adapter is a random number prefixed by 'ftpa'. The extension of the file is '.tmp'. For example, *ftpa23423.tmp*

The 'generateUniqueFile' property is available at the interaction specification level when configuring the create operation and as an attribute in the FTPFile business object.

Outbound operations – Create (continued)

▪ Attributes (continued)

▶ Sequence file

- Available as Managed Connection Factory property
- Sequence # appended to the specified file name. ex: Customer.1.txt
- Sequence file used to store information in the form
`<dirPath>Customer.txt=1`
- In clustered environment, sequence file should be accessible by all cluster members
- "*Generate Unique File*" has precedence over sequence file



If the sequence file name with a complete path is provided for the 'Sequence file' property in the "Service configurations properties" panel of the external service wizard, the adapter will append a sequence number to the output file name specified in the request. For example, if the output file name that you specified is Customer.txt, a file with the name Customer.n.txt is created, where 'n' is the sequence number for a particular request. The sequence will start with 1. If another request with an output file name set to Order.txt is received, a new sequence starting with 1 is generated for Order.txt. If the file name already exists, the adapter will return a DuplicateRecordException.

The sequence file will store the sequences as `<dirPath>Customer.txt = 1` where `<dirPath>` is the output directory path and 1 is the sequence number with which the current file has been created. The next time a request comes with the same output file name and directory path, the sequence number is incremented and used to create the Customer.2.txt file.

With file sequencing, set the output directory and file name in the connection properties panel in the external service wizard rather than setting it at business object attribute level for each request. If the directory path and file name are specified in the business object, these values will take precedence over the values specified in the connection properties screen. The properties that you set in the connection properties screen are saved as managed connection factory properties.

When the application using the adapter is deployed to a clustered environment, the FileSequenceLog must be a file on a mapped drive that is accessible by all of the clusters. Make sure that the resource adapter has the write permission for the sequence log file, or an IOException is returned. If the Sequence file property has a value, and if the generateUniqueFile property is set to true, the 'generateUniqueFile' property takes precedence. If the sequence file is deleted manually, the sequences are lost and will start from 1 again. You can also reset the sequence by changing the sequence value in the sequence file.

Outbound operations - Create

The screenshot displays two windows from the IBM WebSphere Adapter for FTP V6.1 interface. On the left, a table lists attributes for the 'FTPFile' business object. The 'GenerateUniqueFile' attribute is highlighted with a red box. A yellow callout points to it with the text: 'Generate unique file attribute in the business object'. On the right, the 'External Service' dialog shows the configuration for the 'createFTPFile' operation. The 'Generate a unique file' checkbox is also highlighted with a red box. A yellow callout points to it with the text: 'Generate unique file option at Interaction specification level.' The dialog includes fields for remote directory, default target file name, local directory, and local archive directory, along with checkboxes for 'File in local directory', 'Archive file in the local directory for create operation', and 'Create new file if the file does not exist'. Navigation buttons at the bottom include '< Back', 'Next >', 'Finish', and 'Cancel'.

Attribute Name	Type
SecondServerDirectory	string
SecondServerUsername	string
SecondServerPassword	string
FileContentEncoding	string
IncludeEndBODelimiter	string
FileInLocalDirectory	boolean
LocalDirectoryPath	string
LocalArchivingEnabledForCreate	boolean
LocalArchiveDirForCreate	string
StagingDirectory	string
GenerateUniqueFile	boolean
CreateFileIfNotExists	boolean
ScriptFileParameters	string []
ScriptFunctionClassName	string

This slide shows the screen captures of the generate unique file attribute in the business object and as a property at the interaction specification level for a 'create' operation.

Outbound operations - Create

External Service
Service Configuration Properties
For this service, specify security and connection configuration properties.

Deploy connector project:

Connection properties:

Connection properties

FTP system connection information

Host name:

Directory:

Protocol:

Port number:

<< Advanced

Advanced connection configuration

User name:

Password:

The staging directory is used to store files temporarily. It is used to resolve write conflicts.

Staging directory:

Default target file name:

To add sequence numbers to target file names, specify the location of a sequence file.

Sequence file:

Encoding used by FTP server:

Optionally populate the fully qualified class name of the custom parser which is used to parse the "I" output. This is used only when the "I" output deviates from standard output.

Custom parser class name:

Location and name of the sequence file used to keep track of sequence numbers

Here is the “Service Configuration Properties” panel in the external service wizard where you can provide the location and name of the sequence file used to store the present sequence number for a particular file name.

Outbound operations - Append

- Append – Content appended to the file specified. File name returned to indicate success
 - ▶ If file does not exist, RecordNotFoundException is thrown
 - ▶ Create if file not exists
 - Available at Interaction spec and as attribute of FTPFile business object or the user-defined business object wrapper
 - ▶ Generate unique file
 - Available as Interaction specification property and as attribute of FTPFile business object or the user-defined business object wrapper
 - Value provided for file name is ignored
 - Generates unique file name with format `"ftpa<randomnumber>.tmp"`

This slide summarizes the behavior of the append operation. The content specified in the business object sent with the append request is appended to the file specified in the request. When using the user-defined data type, the data is transformed into xml and appended to the file specified. Filename is returned back to the calling application indicating a successful response. If the file specified does not exist, a DuplicateRecordException is thrown.

Setting the createFileIfNotExists property either at the interaction specification level for append operation or at the business object level will result in creation of a new file if one with a name specified in the append request does not exist.

Generate unique file property is also supported for the append operation. The behavior is the same as for the create operation.

Outbound operations - Append

The screenshot displays the 'External Service' configuration wizard. On the left, a list of properties for the 'FTPFile' business object is shown. The 'CreateFileIfNotExists' property is highlighted with a red box. A yellow callout bubble points to this property with the text: 'CreateFileIfNotExists option at business object level'.

On the right, the 'Operations' section shows two operations: 'createFTPFile' and 'appendFTPFile'. The 'appendFTPFile' operation is selected. Below it, the 'Operation properties' section is visible. Under 'InteractionSpec properties for 'appendFTPFile'', the 'FTP system connection information' is shown. The 'Create new file if the file does not exist' and 'Generate a unique file' options are highlighted with red boxes. A yellow callout bubble points to these options with the text: 'CreateFileIfNotExists option at Interaction spec level' and 'Generate a unique file option at interaction spec'.

At the bottom of the slide, the text reads: 'IBM WebSphere Adapter for FTP V6.1' and '© 2008 IBM Corporation'.

This slide shows the screen capture of the CreateFileIfNotExists property at the business object level and at the interaction specification level when configuring the append operation using the external service wizard.

Outbound operations - Overwrite

- Overwrite – Overwrite the file in the directory with the content specified in the request. File name returned to indicate success
 - ▶ If file does not exist, RecordNotFoundException is thrown
 - ▶ Create if file not exists
 - Available at Interaction spec and as attribute of FTPFile business object or the user-defined business object wrapper
 - ▶ Generate unique file
 - Available as Interaction spec property and as attribute of FTPFile business object or the user-defined business object wrapper
 - Value provided for file name is ignored
 - Generates unique file name with format "*ftpa<randomnumber>.tmp* "



The overwrite operation, as the name suggests, overwrites the contents of the existing file with the contents specified in the overwrite request. The CreateFileIfNotExists and GenerateUniqueFile properties are supported for the overwrite operation. The behavior of setting these two properties is the same as described for append operation

Outbound operations - Retrieve

- Retrieve – Retrieve the contents of the file specified
 - ▶ Data transformation now supported in V6.1
 - ▶ In V6.0.2 file contents were returned as binary array
 - ▶ RetrieveResponseWrapper object used to populate the retrieved file contents
 - ▶ If file does not exist, RecordNotFoundException is thrown
 - ▶ Delete on Retrieve
 - Available at Interaction spec and as attribute of FTPFile business object or the user-defined business object wrapper
 - File deleted on retrieve
 - ▶ Archive directory for Delete on retrieve
 - Available as Interaction spec property and as attribute of FTPFile business object or the user-defined business object wrapper
 - File is archived in the directory specified
 - ▶ *SplitCriteria* and *SplitClassName*
 - Used to retrieve multiple user define objects from the file
 - SplitCriteria – ex:###;\n or ###;,\$. Ability to support Windows® or UNIX® new lines as delimiters
 - SplitClassName – com.ibm.j2ca.utils.filesplit.SplitByDelimiter

In the previous versions of the ftp adapter, the retrieve operation retrieved the contents of a file only in binary format. With V6.1 the adapter supports data transformation for retrieve operation. Adapter can read the xml content in the file and return the user-defined object in the RetrieveResponseWrapper object. Delete on retrieve property is available to be configured at the interaction specification level and as an attribute in the FTPFile business object. If this property is set to true, the adapter deletes the file after retrieving its contents. Archive directory for delete on retrieve is a property that can be used along with delete on retrieve property. The directory path that you specify for “Archive directory for Delete on retrieve” is the location where the adapter archives the files after deleting it from its original location.

Outbound operations - Retrieve

FTPFile

<input type="checkbox"/>	FileInLocalDirectory	boolean
<input type="checkbox"/>	LocalDirectoryPath	string
<input type="checkbox"/>	LocalArchivingEnabledForCreate	boolean
<input type="checkbox"/>	LocalArchiveDirForCreate	string
<input type="checkbox"/>	StagingDirectory	
<input type="checkbox"/>	GenerateUniqueFile	
<input type="checkbox"/>	CreateFileIfNotExist	
<input type="checkbox"/>	ScriptFileParameters	string []
<input type="checkbox"/>	SplittingFunctionClassName	string
<input type="checkbox"/>	SplitCriteria	string
<input type="checkbox"/>	DeleteOnRetrieve	boolean
<input type="checkbox"/>	ArchiveDirectoryForRetrieve	string
<input type="checkbox"/>	Content	UnstructuredContent

External Service

Operations
Add, edit or remove operations that will be used by the adapter to access native functions.

Operations:

- createFTPFile (http://www.ibm.com/xmlns/prod/websphere)2caftp(ftpfile)FTPFile) : (http://w...
- appendFTPFile (http://www.ibm.com/xmlns/prod/websphere)2caftp(ftpfile)FTPFile) : (http://w...
- retrieveFTPFile (http://www.ibm.com/xmlns/prod/websphere)2caftp(ftpfile)FTPFile) : (http://w...

Operation properties:

InteractionSpec properties for 'retrieveFTPFile'

FTP system connection information

Remote directory on FTP system:

Default target 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 retrieve operation

Remote archive directory for retrieve operation:

Advanced >>

< Back Next > Finish Cancel

Callouts:

- Split criteria options at business object level
- Delete on retrieve and archive directory options at business object level
- Delete on retrieve and archive directory options at Interaction spec level

IBM WebSphere Adapter for FTP V6.1 © 2008 IBM Corporation 22

This slide shows the screen capture of the properties available at the business object level and at the interaction specification level when configuring the retrieve operation using the external service wizard.

Outbound operations – Delete, exists, list

- **Delete**
 - ▶ Deletes the file from the file system. No output
 - ▶ If file does not exist, RecordNotFoundException is thrown
- **Exists**
 - ▶ Checks for existence of file in the directory specified
 - ▶ Search includes searches the subfolders
- **List**
 - ▶ Lists all the files in the directory specified
- **ExecuteFTPScript**
 - ▶ Parameter substitution
 - parameters required by the FTP script file can be passed as part of the request data
- **ServerToServerFileTransfer**
 - ▶ Transfer files between ftp servers

This slide summarizes the rest of the operations supported by ftp adapter for outbound interaction style. Delete operation deletes the file from the folder specified. Exists checks for existence of file in the specified folder and any subfolders. List operation lists all the files in the directory specified. ExecuteFTPScript operation has been enhanced in V6.1. You have the option to send parameters at runtime. The parameters can be set in the business object as attributes.

Section

Inbound

The next section will provide an overview of operations types supported for inbound and provides details of the attributes that define the behavior of each operation type

Event management

- Inbound event delivery
 - ▶ Adapter polls event files periodically based on user-configured file mask, poll quantity and poll period
 - ▶ Adapter can deliver same event to multiple endpoints
 - Multiple Activation specs with same attribute values
- Event table (event store)
 - ▶ Used to store event information
 - Need to configure data source on the runtime
 - ▶ Adapter populates the event table with the current status of each event and deletes them when delivery is successful.
 - ▶ Each event delivered to it's corresponding endpoint as part of a unique XA transaction
 - ▶ Provides recovery and guaranteed event delivery.
- Event table not mandatory in V6.1
 - ▶ Adapter uses in memory representation of event table to process events

The event manager is a framework for delivering inbound events. An event table must exist before inbound processing can occur. The event table is created automatically in a Cloudscape database by the adapter upon deployment of the application to the WebSphere Process Server runtime. All of the information regarding the events polled is stored in the event table. The adapter polls event files periodically based on user-configured file mask, poll quantity and poll period properties. It then processes and transmits these events to various predefined endpoints. Delivery of events is done through the use of XA transactions between the recording of the event in the event table and the delivery of the event to the endpoint.

Defining an event table is not mandatory in V6.1, so you can leave the event table related properties empty in the activation spec. The adapter uses in memory representation of the event table to process the events. But “Assured-once” delivery is not supported and if the adapter fails or runtime has to restart, all event information is lost.

Supported inbound operations and features

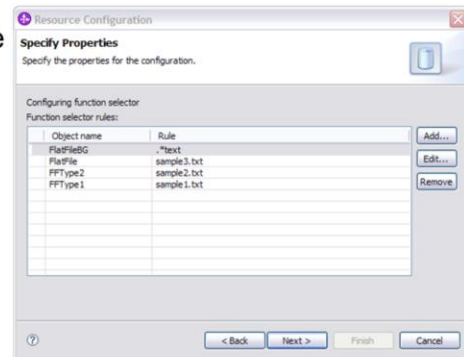
- Operation type
 - ▶ Emit
- Data transformation supported for inbound
- Can define multiple methods/operations for inbound.
- Optional features to split large event files
 - ▶ SplitBySize
 - ▶ SplitByDelimiter
 - Chunking based on the value provided for SplitCriteria.
 - Common delimiters include comma (,), semicolon (;), quote (" '), braces ({}), or slashes (/ \).
 - ▶ Each chunk represents a business object
 - ▶ Reassemble
 - Chunk information included "chunkFileName" of the business object
 - Includes event id
 - Event ID is in the form of event file location and timestamp – including current chunk number and total number of chunks.

The inbound operation supported is the emit operation. It reads the file from the event directory. With V6.1 you can now define more than one operation in your export. Function selectors are used to map the incoming event to an appropriate method in export file. More information about function selectors is provided later in the presentation

An optional feature for inbound processing includes the capability to split large event files into chunks. Configuration of this functionality is specified in the properties `SplittingFunctionClassName` and `SplitCriteria`. If the input file is larger than the `SplitCriteria` property, then the file is split into chunks based on the value size in bytes specified or based on the specified delimiter. Each chunk represents a business object. To reassemble the chunks, information is included in the `chunkFileName` attribute of the business object. This includes event ID. The event ID is in the form of event file location and timestamp, including the current chunk number and total number of chunks. Reassembly can be handled by business logic implemented in a Java component, for example.

Function selector

- **Function selector**
 - ▶ Required to map events to the appropriate SCA export function name
 - ▶ FilenameFunctionSelector and EmbeddedNameFunctionSelector.
- **FilenameFunctionSelector**
 - ▶ Rule-based function selector that can match a regular expression on a file name to an object name
 - ▶ Used when Object name cannot be determined from event file
- **EmbeddedNameFunctionSelector**
 - ▶ Used for user-defined object types



27

IBM WebSphere Adapter for FTP V6.1

© 2008 IBM Corporation

In Inbound, Function Selectors are required in order to map between events generated to the appropriate SCA export function name. There are two function selectors provided by the adapter foundation classes that are supported by the ftp adapter. These are FilenameFunctionSelector and EmbeddedNameFunctionSelector.

The FilenameFunctionSelector is a rule-based function selector that can match a regular expression on a file name to an object name. This is represented in properties as a 2-column table, with N rows. The adapter polls on the event directory for any events. When a file is found in the event directory, the adapter retrieves the file and places the event details in the event table. FilenameFunctionSelector will evaluate the regular expression you specified in the rule field by using the event file name and maps it to an object name. Once the object name is determined, the function selector can determine the native method name and then map the event to the correct operation in the export. FilenameFunctionSelector is used when the object name cannot be determined from the event file.

The EmbeddedNameFunctionSelector is used in case of user-defined business objects, where the object name is embedded in the event file. For example, if the business object is Customer, the event file has the object name (Customer) embedded in the event file. This function selector should be configured with a data handler. The data binding should be the adapter-specific DataBinding(FTPfileBaseDataBinding). The data binding should be configured to use the same data handler configured during creating the function selector configuration.

Section

Problem determination

This section covers problem determination for the FTP adapter

Problem determination

- Covered in the Adapters Overview presentation – Recap:
 - ▶ WebSphere Process Server log files
 - SystemOut.log and SystemErr.log
 - ▶ Adapter Log and Trace files configured in WebSphere Integration Developer External service wizard
 - Also can set RAR custom properties in the Administrative console of the Process Server
 - ▶ Different logging level levels can be set
- Enabling trace for Adapters in WebSphere Process Server :
 - ▶ Set the tracing level string as `"com.ibm.j2ca.ftp.*=finest"`

Listed here are the log files for WebSphere Process Server. The adapter logs the entries in the SystemOut.log and SystemErr.log files. You can also enable tracing for more detailed information. Note the trace string used to turn on tracing of the ftp adapter.

Section

Summary

This section is a summary of the topics covered in this presentation

Summary

- WebSphere Adapter for FTP enables integration with SCA Applications and Enterprise Information System applications that can communication only through files
 - ▶ Inbound and Outbound support
- Provides support for Enterprise Service Discovery for discovering services
- Supports data transformation for both inbound and inbound
- Optional feature splits large event files based on size or a specified delimiter.

In summary, the WebSphere Adapter for FTP enables integration with SCA business integration applications and Enterprise Information System applications that can communicate only through files in a file system. The adapter supports both inbound and outbound interaction. Enterprise service discovery is used for discovery services and creates the service description with specified values for custom adapter properties. The business object supported by the FTP adapter is of a single, pre-defined structure and the adapter is unaware of the content of the file. An optional feature for inbound processing of large event files includes the capability to split files based on size in bytes or a specified delimiter for data transformation framework (DTF) flows.

Reference information

- WebSphere Adapter for FTP User Guide
- Java Connector Architecture
 - ▶ <http://java.sun.com/j2ee/connector/index.jsp>
- Enterprise metadata discovery
 - ▶ <http://www.ibm.com/developerworks/java/library/j-emd/>
- WebSphere Adapter Information Center
 - ▶ <http://www-306.ibm.com/software/integration/wbiadapters/library/infocenter/>
- WebSphere Process Integration Information Center
 - ▶ <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/index.jsp>

Additional reference information can be found at these addresses.

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM WebSphere

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

J2EE, Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. 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 in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

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

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.

