SUN SEEBEYOND

eWAY™ BATCH ADAPTER USER'S GUIDE

Release 5.1.3

Sun microsystems

Part Number: 820-0981

Version 20070419143857

Contents

Chapter 1

Introducing the Batch eWay

This chapter provides a brief overview of the Sun SeeBeyond Batch eWay Intelligent Adapter, and presents an outline of the information provided in this user's guide.

What's in This Chapter

1.1 About the Batch eWay Intelligent Adapter

All eWays provide a communication bridge between the eGate environment and one or more external systems.

The Batch eWay Intelligent Adapter, referred to as the Batch eWay throughout this document, performs a variety of FTP and FTP-related operations (depending on your specific needs, network environment, record-processing, file transfer, and external system requirements). The Batch eWay enables eGate to use an FTP connection to exchange data with other network hosts for the purpose of receiving and delivering objects stored in files.

1.1.1 Batch eWay OTDs

The Batch eWay provides Object Type Definitions (OTDs) that enable the creation of file transfer operations using FTP.

The Batch eWay includes seven specific OTDs:

- **BatchFTP**: The FTP OTD connects to external FTP servers.
- **BatchFTPOverSSL**: The FTP over SSL OTD provides secure data transfer using Secure Sockets Layer (SSL) protocol.
- **BatchSCP**: The SCP OTD provides secure data transfer using Secure Copy Protocol with Secure Shell (SSH) as an underlying protocol.

- **BatchSFTP**: The SFTP OTD provides secure data transfer using SSH File Transfer Protocol (SFTP protocol). SFTP protocol provides a range of operations on remote files, such as directory listings, and remote file removal.
- **BatchLocalFile**: The local file OTD picks up or puts data files to local file systems.
- **BatchRecord**: The record-processing OTD extracts records out of files, parses files into specific records, and defines the content of files as records.
- **BatchInbound**: The inbound OTD receives a file, renames the file with GUID file name, and triggers the Business Process or Collaboration.

Note: *The Batch eWay supports standard FTP in accordance with RFC-959.*

1.2 What's New in This Release

The Batch eWay Intelligent Adapter includes the following changes and new features:

New for Version 5.1.3

- Added support: Supports automatic deployment of EAR files to WebLogic Application Server (version 9.1).

New for Version 5.1.2

- New getIfExists (readIfExists) functionallity added to BatchFTP and BatchLocalFile OTD for use with JCD and Business Processes.

New for Version 5.1.1

- Supports automatic deployment of EAR files to Sun Java System Application Server.

New for Version 5.1.0

- Version Control: An enhanced version control system allows you to effectively manage changes to the eWay components.
- Manual Connection Management: Establishing a connection can now be performed automatically (configured as a property) or manually (using OTD methods from the Java Collaboration). This includes Automatic and manual modes of connection for BatchFTPOverSSL, BatchSCP, and BatchSFTP OTDs.
- Multiple Drag-and-Drop Component Mapping from the Deployment Editor: The Deployment Editor now allows you to select multiple components from the Editor's component pane, and drop them into your Environment component.
- Support for Runtime LDAP Configuration: eWay configuration properties now support LDAP key values.
- MDB Pool Size Support: Provides greater flow control (throttling) by specifying the maximum and minimum MDB pool size.
- Connection Retry Support: Allows you to specify the number of attempts to reconnect, and the interval between retry attempts, in the event of a connection failure.

- Enhanced support for user-defined heuristics.
- Added SSL support for Batch server and client authentication.
- Added support for pre and post-transfer operations in BatchFTPOverSSL and BatchSFTP OTDs.

Many of these features are documented further in the *Sun SeeBeyond eGate™ Integrator User's Guide* or the *Sun SeeBeyond eGate™ Integrator System Administrator Guide*.

1.3 What's in This Document

This book includes the following chapters:

- **Chapter 1 "Introducing the Batch eWay"** provides an overview of the Batch eWay Intelligent Adapter.
- **Chapter 2 "Installing the Batch eWay"**provides the supported operating systems and system requirements for the Batch eWay. It also includes directions for installing the eWay and accessing the accompanying documentation and sample Projects.
- **Chapter 3 "Configuring the Batch eWay"** describes the process of configuring the Batch eWay to run in your environment.
- **Chapter 4 "Understanding Batch eWay OTDs"** provides an overview of the Object Type Definitions (OTDs) available with the Batch eWay.
- **Chapter 5 "Additional Batch eWay Features"**provides information on the Data Streaming, SSH Tunneling, and SOCKS features of the Batch eWay:
- **Chapter 6 "Using the Batch eWay with eInsight"** describes how to use the Batch eWay with the Sun Java Composite Application Platform Suite's eInsight Business Process Manager and the Web Services interface.
- **Chapter 7 "Using the Batch eWay with Java Collaborations"** describes how to use the Batch eWay using the eGate Integrator and the Collaboration Editor (Java).

1.3.1 Batch eWay Javadoc

The Batch eWay Javadoc documents the available Java methods provided with the Batch eWay. The Javadoc is uploaded with the eWay's documentation file, **BatcheWayDocs.sar**, and downloaded from the Documentation tab of the Enterprise Manager. To access the full Javadoc, extract the Javadoc to an easily accessible folder, and double-click the **index.html** file.

1.3.2 Scope of the Document

This user's guide provides a description of the Batch eWay Intelligent Adapter. It includes directions for installing the eWay, configuring the eWay properties, and implementing the eWay's sample Projects. This document is also intended as a

reference guide, listing available properties, functions, and considerations. For a reference of available Batch eWay Java methods, see the associated Javadoc.

1.3.3 Intended Audience

This guide is intended for experienced computer users who have the responsibility of helping to set up and maintain a fully functioning Java Composite Application Platform Suite system. This person must also understand any operating systems on which the Java Composite Application Platform Suite will be installed (Windows and UNIX), and must be thoroughly familiar with Windows-style GUI operations.

1.3.4 Text Conventions

The following conventions are observed throughout this document.

Table 1 Text Conventions

Text Convention	Used For	Examples
Bold	Names of buttons, files, icons, parameters, variables, methods, menus, and objects	▪ Click **OK**. ▪ On the **File** menu, click **Exit**. ▪ Select the **eGate.sar** file.
`Monospaced`	Command line arguments, code samples; variables are shown in ***`bold italic`***	`java -jar` ***`filename`***`.jar`
Blue bold	Hypertext links within document	See **Text Conventions** on page 14
Blue underlined	Hypertext links for Web addresses (URLs) or email addresses	http://www.sun.com

1.4 Sun Microsystems, Inc. Web Site

The Sun Microsystems web site is your best source for up-to-the-minute product news and technical support information. The site's URL is:

http://www.sun.com

1.5 Documentation Feedback

We appreciate your feedback. Please send any comments or suggestions regarding this document to:

CAPS_docsfeedback@sun.com

1.6 Additional Licensing Considerations

Sun Microsystems, Inc. has integrated software from 3SP (J2SSH Maverick) and /n software (IP*Works SSL) with the Batch eWay.

Use of the integrated software libraries outside of the Sun Java Composite Application Platform Suite Batch eWay are prohibited and may not be reversed engineered or redistributed.

n/software adds the following restrictions to use of IP*Works SSL:

IP*Works SSL software libraries may only be used by the Sun Java Composite Application Platform Suite Batch eWay as a run-time component.

Sun Microsystems, Inc. prohibits Sun Java Composite Application Platform Suite Batch eWay end-user(s) from changing, altering or modifying the licensed software (IP*Works SSL), creating derivative works, translations, reverse assembling, reverse compiling, disassembling, or in any way reverse engineering the licensed software.

Sun Microsystems, Inc. prohibits Sun Java Composite Application Platform Suite Batch eWay end-user(s) from sublicensing, renting, distributing, leasing or otherwise transferring or assigning any portion of the licensed software (IP*Works SSL). You may not create any derivative works of the licensed software.

Chapter 2

Installing the Batch eWay

This chapter contains installation information for the Batch eWay.

What's in This Chapter

2.1 Batch eWay System Requirements

The Batch eWay Readme contains the latest information on:

- Supported Operating Systems
- System Requirements
- External System Requirements

The Batch eWay Readme is uploaded with the eWay's documentation file (BatcheWayDocs.sar) and can be accessed from the Documentation tab of the Sun Java Integrator Suite Installer. Refer to the Batch eWay Readme for the latest requirements before installing the Batch eWay.

2.2 Installing the Batch eWay

The Sun Java Composite Application Platform Suite Installer, a web-based application, is used to select and upload eWays and add-on files during the installation process. The following section describes how to install the components required for this eWay.

Note: *When the Repository is running on a UNIX operating system, the eWays are loaded from the Enterprise Manager running on a Windows platform connected to the Repository server using Internet Explorer.*

2.2.1 Installing the eWay on a JavaCAPS Supported System

Follow the directions for installing the Sun Java Composite Application Platform Suite in the *Sun Java Composite Application Platform Suite Installation Guide. After you have installed eGate or eInsight, do the following:*

1. From the Sun Java Composite Application Platform Suite Installer's **Select Sun Java Composite Application Platform Suite Products to Install** table (Administration tab), expand the **eWay** option.
2. Select the products for your Sun Java Composite Application Platform Suite and include the following:
   - **FileeWay** (the File eWay is used by most sample Projects)
   - **Batch eWay**

   To upload the Batch eWay User's Guide, Help file, Javadoc, Readme, and sample Projects, select the following:
   - **BatcheWayDocs**
3. Once you have selected all of your products, click **Next** in the top-right or bottom-right corner of the **Select Sun Java Composite Application Platform Suite Products to Install** box.
4. From the **Selecting Files to Install** box, locate and select your first product's SAR file. Once you have selected the SAR file, click **Next**. Your next selected product appears. Follow this procedure for each of your selected products. The **Installation Status** window appears and installation begins after the last SAR file has been selected.
5. Once your product's installation is finished, continue installing the Sun Java Composite Application Platform Suite as instructed in the *Sun Java Composite Application Platform Suite Installation Guide.*

Adding the eWay to an Existing Suite Installation

If you are adding the eWay to an existing Sun Java Composite Application Platform Suite installation, do the following:

1. Complete steps 1 through 4 above.
2. Once your product's installation is finished, open the Enterprise Designer and select **Update Center** from the Tools menu. The **Update Center Wizard** appears.
3. For Step 1 of the wizard, simply click **Next**.
4. For Step 2 of the wizard, click the **Add All** button to move all installable files to the **Include in Install** field, then click **Next**.
5. For Step 3 of the wizard, wait for the modules to download, then click **Next**.
6. The wizard's Step 4 window displays the installed modules. Review the installed modules and click **Finish.**
7. When prompted, restart the IDE (Integrated Development Environment) to complete the installation.

2.2.2 Installing eWay Enterprise Manager plug-ins

The **Sun SeeBeyond Enterprise Manager** is a Web-based interface that allows you to monitor and manage your Sun Java Composite Application Platform Suite applications. The Enterprise Manager requires an eWay specific "plug-in" for each of your installed eWays. These plug-ins enable the Enterprise Manager to target specific alert codes for each eWay type, as well as to start and stop the inbound eWays.

The *Sun Java Composite Application Platform Suite Installation Guide* describes how to install the Sun SeeBeyond Enterprise Manager. The *Sun SeeBeyond eGate™ Integrator System Administration Guide* describes how to monitor servers, Services, logs, and alerts using the Sun SeeBeyond Enterprise Manager and the command-line client.

The **eWay Enterprise Manager plug-ins** are available from the **List of Components to Download** under the Sun Java Composite Application Platform Suite Installer's **DOWNLOADS** tab.

There are two ways to add the eWay Enterprise Manager plug-ins:

1. From the **Enterprise Manager**:
   - A. From the **Enterprise Manager**'s Explorer toolbar, click the **Configuration** icon.
   - B. Click the **Web Applications Manager** tab, go to the **Auto-Install from Repository** tab, and connect to your Repository.
   - C. Select the application plug-ins you require, and click **Install**. The application plug-ins are installed and deployed.
2. From the **Sun Java Composite Application Platform Suite Installer:**
   - A. From the **Sun Java Composite Application Platform Suite Installer's Download tab**, select the Plug-Ins you require and save them to a temporary directory.
   - B. Log onto the **Sun SeeBeyond Enterprise Manager**. From the **Enterprise Manager**'s Explorer toolbar, click the **Configuration** icon.
   - C. Click the **Web Applications Manager** tab and go to the **Manage Applications** tab.
   - D. Browse for and select the WAR file for the application plug-in that you downloaded, and click **Deploy**. The plug-in is installed and deployed.

Batch eWay Alert Codes

You can view and delete alerts using the Enterprise Manager. An alert is triggered when a specified condition occurs in a Project component. The purpose of the alert is to warn the administrator or user that a condition has occurred.

To View the eWay Alert Codes

1. Add the eWay Enterprise Manager plug-in for this eWay.
2. From the Enterprise Manager's **Explorer** toolbar, click the **Configuration** icon.

3 Click the **Web Applications Manager** tab and go to the **Manage Alert Codes** tab. Your installed alert codes are displayed under the **Results** section. If your eWay alert codes are not available displayed under **Results**, do the following

A From the **Install New Alert Codes** section, browse to and select the eWay alert properties file for the application plug-in that you added. The alert properties files are located in the **alertcodes** folder of your Sun Java Composite Application Platform Suite installation directory.

B Click **Deploy**. The available alert codes for your application are displayed under **Results**. A listing of available this eWay's alert codes is displayed in Table 2.

Table 2 Batch eWay Alert Codes

Alert Code	Description	Detail/Action
BATCH-FTP-EWAY-CONFIG-FAILED	Batch FTP eWay configuration error, message=[{0}].	An error occurred related to configuration. Check IS log for details. Parameter 0 – the error message.
BATCH-FTP-EWAY-CONN-ACQUIRE-FAILED	Batch FTP eWay error when acquiring connection from pool, message=[{0}].	An error occurred when looking up a connection from the connection pool. Check IS log for details. Parameter 0 – the error message.
BATCH-FTP-EWAY-CONN-INIT-FAILED	Batch FTP eWay connection initialization failed, message=[{0}].	An error occurred when initializing a connection. Check IS log for details. Parameter 0 – the error message.
BATCH-FTP-EWAY-CONNECTION-FAILED	Batch FTP eWay connection failed, method=[{0}], message=[{1}].	An error occurred when connecting to remote. Check IS log for details. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message.
BATCH-FTP-EWAY-ERROR	Batch FTP eWay error, message=[{0}].	An error occurred. Check IS log for details. Parameter 0 – the error message.
BATCH-FTP-EWAY-OPERATION-ERROR	Batch FTP eWay error when doing data transfer operation in [{0}], message=[{1}].	An error occurred when doing a data operation, such as put/get, etc. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message. Check IS log for details.

Alert Code	Description	Detail/Action
BATCH-INBOUND-EWAY-CONFIG-FAILED	Batch Inbound eWay configuration error, message=[{0}].	An error occurred related to configuration. Check IS log for details. Parameter 0 – the error message.
BATCH-INBOUND-EWAY-ERROR	Batch Inbound eWay error, message=[{0}].	
BATCH-INBOUND-EWAY-RUNNING	RUNNING Batch Inbound eWay is running.	Inbound eWay is running. Check IS log for details.
BATCH-INBOUND-EWAY-STARTED	Batch Inbound eWay started.	Inbound eWay started. Check IS log for details.
BATCH-INBOUND-EWAY-STOPPED	STOPPED Batch Inbound eWay is stopped.	Batch Inbound eWay is stopped. Check IS log for details, or restart the Batch Inbound eWay.
BATCH-INBOUND-EWAY-STOPPING	STOPPING Batch Inbound eWay is being stopped.	Batch Inbound eWay is stopping. Check IS log for details, or restart the Batch Inbound eWay.
BATCH-INBOUND-EWAY-SUSPENDED	SUSPENDED Batch Inbound eWay is suspended.	Batch Inbound eWay is suspended. Check IS log for details, or restart the Batch Inbound eWay.
BATCH-INBOUND-EWAY-SUSPENDING	SUSPENDING Batch Inbound eWay is suspending.	Batch Inbound eWay is suspending. Check IS log for details, or restart the Batch Inbound eWay.
BATCH-LOCALFILE-EWAY-CONFIG-FAILED	Batch Local File eWay configuration error, message=[{0}].	An error occurred related to configuration. Check IS log for details. Parameter 0 – the error message.
BATCH-LOCALFILE-EWAY-CONN-ACQUIRE-FAILED	Batch Local File eWay error when acquiring connection from pool, message=[{0}].	An error occurred when looking up a connection from connection pool. Check IS log for details. Parameter 0 – the error message.
BATCH-LOCALFILE-EWAY-CONN-INIT-FAILED	Batch Local File eWay connection initialization failed, message=[{0}].	An error occurred when initializing connection. Check IS log for details. Note: A connection object for local file is only an object pooled in the connection pool. It does not physically connect to any host. Parameter 0 – the error message.
BATCH-LOCALFILE-EWAY-ERROR	Batch Local File eWay error, message=[{0}].	A general error occurred. Check IS log for details. Parameter 0 – the error message.

Alert Code	Description	Detail/Action
BATCH-LOCALFILE-EWAY-OPERATION-ERROR	Batch Local File eWay error when doing file operation in [{0}], message=[{1}].	An error occurred when doing local file operations, such as put/get. Check IS log for details. Parameter 0 – the method in which the error occurred Parameter 1 – the error message.
BATCH-REC-EWAY-CONFIG-FAILED	Batch Record eWay configuration error, message=[{0}].	An error occurred related to configuration. Check IS log for details. Parameter 0 – the error message.
BATCH-REC-EWAY-CONN-ACQUIRE-FAILED	Batch Record eWay error when acquiring connection from pool, message=[{0}].	An error occurred when looking up a connection from connection pool. Check IS log for details. Parameter 0 – the error message.
BATCH-REC-EWAY-CONN-INIT-FAILED	Batch Record eWay connection initialization failed, message=[{0}].	An error occurred when initializing connection. Check IS log for details. Note, a connection object for batch record is only an object pooled in the connection pool, it does not physically connect to any host. Parameter 0 – the error message.
BATCH-REC-EWAY-ERROR	Batch Record eWay error, message=[{0}].	A general error occurred. Check IS log for details. Parameter 0 – the error message.
BATCH-REC-EWAY-OPERATION-ERROR	Batch Record eWay error when doing record operation in [{0}], message=[{1}].	An error occurred when doing a record operation, such as put/get, etc. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message. Check IS log for details.
FTP-SSL-EWAY-CONFIG-FAILED	Batch FTP Over SSL eWay configuration error, message=[{0}].	An error occurred related to configuration. Check IS log for details. Parameter 0 – the error message.

Alert Code	Description	Detail/Action
FTP-SSL-EWAY-CONN-ACQUIRE-FAILED	Batch FTP Over SSL eWay error when acquiring connection from connection pool, message=[{0}].	Error when looking up a connection from connection pool. Check IS log for details. Parameter 0 – the error message.
FTP-SSL-EWAY-CONN-INIT-FAILED	Batch FTP Over SSL eWay connection initialization error, message=[{0}].	Error when initializing a connection. Check IS log for details. Parameter 0 – the error message.
FTP-SSL-EWAY-CONNECTION-FAILED	Batch FTP Over SSL eWay connection failed, method=[{0}], message=[{1}].	An error occurred when connecting to remote. Check IS log for details. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message.
FTP-SSL-EWAY-ERROR	Batch FTP Over SSL eWay error, message=[{0}].	General error occurred. Check IS log for details. Parameter 0 – the error message.
FTP-SSL-EWAY-OPERATION-ERROR	Batch FTP Over SSL eWay error when doing data transfer operation in [{0}], message=[{1}].	An error occurred when doing a data operation, such as put/get, etc. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message. Check IS log for details.
SCP-EWAY-CONFIG-FAILED	Batch SCP eWay configuration error, message=[{0}].	Error occurred related to configuration. Check IS log for details. Parameter 0 – the error message.
SCP-EWAY-CONN-ACQUIRE-FAILED	Batch SCP eWay error when acquiring connection from connection pool, message=[{0}].	Error when looking up a connection from connection pool. Check IS log for details. Parameter 0 – the error message.
SCP-EWAY-CONN-INIT-FAILED	Batch SCP eWay connection initialization error, message=[{0}].	Error when initializing a connection. Check IS log for details. Parameter 0 – the error message.
SCP-EWAY-CONNECTION-FAILED	Batch SCP eWay connection failed, method=[{0}], message=[{1}].	Error when connecting to remote. Check IS log for details. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message.

Alert Code	Description	Detail/Action
SCP-EWAY-ERROR	Batch SCP eWay error, message=[{0}].	General error occurred. Check IS log for details. Parameter 0 – the error message.
SCP-EWAY-OPERATION-ERROR	Batch SCP eWay error when doing data transfer operation in [{0}], message=[{1}].	An error occurred when doing a data operation, such as put/get, etc. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message. Check IS log for details.
SFTP-EWAY-CONFIG-FAILED	Batch SFTP eWay configuration error, message=[{0}].	An error occurred related to configuration. Check IS log for details. Parameter 0 – the error message.
SFTP-EWAY-CONN-ACQUIRE-FAILED	Batch SFTP eWay error when acquiring connection from connection pool, message=[{0}].	An error occurred when looking up a connection from connection pool. Check IS log for details. Parameter 0 – the error message.
SFTP-EWAY-CONN-INIT-FAILED	Batch SFTP eWay connection initialization error, message=[{0}].	An error occurred when initializing a connection. Check IS log for details. Parameter 0 – the error message.
SFTP-EWAY-CONNECTION-FAILED	Batch SFTP eWay connection failed, method=[{0}], message=[{1}].	An error occurred when connecting to remote. Check IS log for details. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message.
SFTP-EWAY-ERROR	Batch SFTP eWay error, message=[{0}].	A general error occurred. Check IS log for details. Parameter 0 – the error message.
SFTP-EWAY-OPERATION-ERROR	Batch SFTP eWay error when doing data transfer operation in [{0}], message=[{1}].	An error occurred when doing a data operation, such as put/get, etc. Parameter 0 – the method in which the error occurred. Parameter 1 – the error message. Check IS log for details.

An alert code is a warning that an error has occurred. It is not a diagnostic. The user actions noted above are just some possible corrective measures you may take. Refer to

the log files for more information. For information on managing and monitoring alert codes and logs, see the *Sun SeeBeyond eGate Integrator System Administration Guide.*

2.2.3 After Installation

Once you install the eWay, it must then be incorporated into a Project before it can perform its intended functions. See the *Sun SeeBeyond eGate™ Integrator User's Guide* for more information on incorporating the eWay into an eGate Project.

2.3 Deploying an EAR File

The Sun Java Composite Application Platform Suite Enterprise Designer can be configured to automatically deploy an EAR file to the Sun Java System Application Server. To configure the Enterprise Designer for deployment, follow the directions for deploying applications to the Sun Java System Application Server, provided in the *Sun SeeBeyond eGate Integrator System Administration Guide*. Because automatic deployment is not supported directly from Enterprise Designer for the Weblogic Application Server, additional instructions are provided below.

2.3.1 WebLogic Application Servers

1. Complete the following step before building the EAR file, which is generated in Enterprise Designer and must be deployed manually to the WebLogic Application Server.
2. Set the following properties with explicit values:
   - **State Persistence Base Location** for **BatchFTP**: see **"State Persistence Base Location" on page 56**
   - **State Persistence Base Location** for **BatchLocalFile**: **"State Persistence Base Location" on page 121**
   - **Temp Dir** for **BatchFTPOverSSL**: see **"Temp Dir" on page 80**
3. Use your WebLogic Admin console to deploy the EAR file.

Refer to your application server's documentation for requirements regarding working directories.

2.4 ICAN 5.0 Project Migration Procedures

This section describes how to transfer your current ICAN 5.0 Projects to the Sun Java Composite Application Platform Suite version 5.1.3. To migrate your ICAN 5.0 Projects, do the following:

Export the Project

1. Before you export your Projects, save your current ICAN 5.0 Projects to your Repository.
2. From the Project Explorer, right-click your Project and select **Export** from the shortcut menu. The Export Manager appears.
3. Select the Project that you want to export in the left pane of the Export Manager and move it to the Selected Projects field by clicking the **Add to Select Items** (arrow) button, or click **All** to include all of your Projects.
4. In the same manner, select the Environment that you want to export in the left pane of the Export Manager and move it to the Selected Environments field by clicking the **Add to Select Items** (arrow) button, or click **All** to include all of your Environments.
5. Browse to select a destination for your Project ZIP file and enter a name for your Project in the **ZIP file** field.
6. Click **Export** to create the Project ZIP file in the selected destination.

Install Sun Java Composite Application Platform Suite 5.1.3

7. Install the Sun Java Composite Application Platform Suite 5.1.3, including all eWays, libraries, and other components used by your ICAN 5.0 Projects.
8. Start the Sun SeeBeyond Enterprise Designer.

Import the Project

9. From the Java Integration Suite 5.1 Enterprise Designer's Project Explorer tree, right-click the Repository and select **Import Project** from the shortcut menu. The Import Manager appears.
10. Browse to and select your exported Project file.
11. Click **Import**. A warning message, **"Missing APIs from Target Repository**," may appear at this time. This occurs because various product APIs were installed on the ICAN 5.0 Repository when the Project was created, that are not installed on the Sun Java Composite Application Platform Suite 5.1 Repository. These APIs may or may not apply to your Projects. You can ignore this message if you have already installed all of the components that correspond to your Projects. Click **Continue** to resume the Project import. Close the Import Manager after the Project is successfully imported.

Deploy the Project

12. A new Deployment Profile must be created for each of your imported Projects. When a Project is exported, the Project's components are automatically *"checked in"* to Version Control to write-protected each component. These protected components appear in the Explorer tree with a red padlock in the bottom-left corner of each icon. Before you can deploy the imported Project, the Project's components must first be *"checked out"* of Version Control from both the Project Explorer and the Environment Explorer. To *"check out"* all of the Project's components, do the following:

A From the Project Explorer, right-click the Project and select **Version Control > Check Out** from the shortcut menu. The Version Control - Check Out dialog box appears.

B Select **Recurse Project** to specify all components, and click **OK**.

C Select the Environment Explorer tab, and from the Environment Explorer, right-click the Project's Environment and select **Version Control > Check Out** from the shortcut menu.

D Select **Recurse Environment** to specify all components, and click **OK**.

13 If your imported Project includes File eWays, these must be reconfigure in your Environment prior to deploying the Project. To reconfigure your File eWays, do the following:

A If your Environment includes both inbound and outbound File External Systems, delete one of these, for example, the outbound File External System.

B From the Environment Explorer tree, right-click your remaining File External System, and select **Properties** from the shortcut menu. The Properties Editor appears.

C Set the inbound and outbound directory values, and click **OK**. The File External System can now accommodate both inbound and outbound eWays.

14 Deploy your Projects.

Note: *Only projects developed on ICAN 5.0.2 and above can be imported and migrated successfully into the Java Integration Suite.*

Chapter 3

Configuring the Batch eWay

This chapter explains how to configure the Batch eWay properties.

What's in This Chapter

3.1 Creating and Configuring Batch eWays

All eWays contain a set of parameters with properties unique to that eWay type. After the eWays are established and a Batch External System is created in the Project's Environment, the eWay parameters can be modified for your specific system. The Batch eWay contains Properties templates for the following OTDs:

- **BatchFTP**
- **BatchFTPOverSSL**
- **BatchSCP**
- **BatchSFTP**
- **BatchLocalFile**
- **BatchRecord**
- **BatchInbound**

All Batch eWays contain properties that are accessed from the **Connectivity Map** and the **Environment Explorer** tree. **Connectivity Map** properties most commonly apply to a specific eWay, and may vary from other eWays (of the same type) in the Project. The eWay properties accessed from the **Environment Explorer** tree are commonly global, applying to all eWays (of the same type) in the Project.

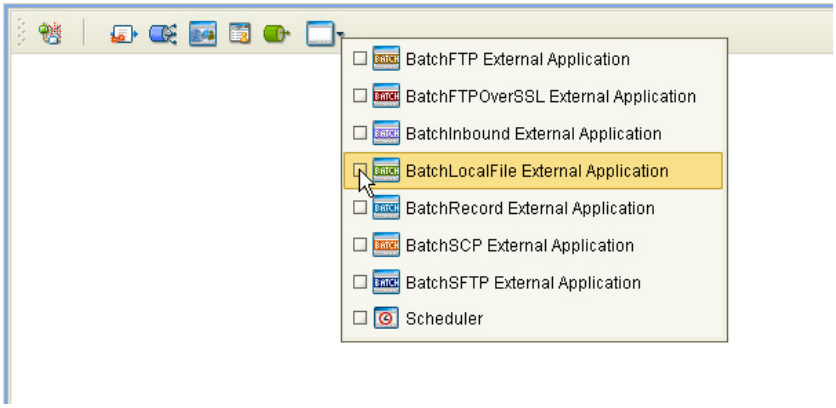
3.1.1 Selecting a Batch External Application

To create a Batch eWay, you must first create a Batch External Application in your Connectivity Map. Batch eWays are located between a Batch External Application and a Service. Services are containers for Collaborations, Business Processes, eTL processes, and so forth.

To create the Batch External Application

1. From the Connectivity Map toolbar, click the **External Applications** icon.
2. Select a **Batch External Application** from the menu (see Figure 1). The selected Batch External Application icon appears on the Connectivity Map toolbar.

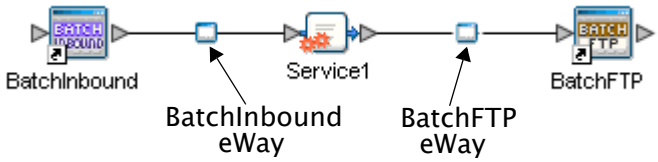
Figure 1 External Applications Selection Menu



3. Drag the new **Batch External Application** from the toolbar onto the Connectivity Map canvas. This represents an external Batch system.

   From the Connectivity Map, you can associate (bind) the External Application with the Service to establish an eWay (see Figure 2).

Figure 2 eWay Location.



When Batch is selected as the External Application, it automatically applies the default Batch eWay properties, provided by the OTD, to the eWay that connects it to the Service. These properties can then be modified for your specific system, using the **Properties Editor**.

3.1.2 Modifying the eWay Properties

A Project's properties can be modified after the eWays are established in the Connectivity Map and the Environment is created.

Modifying the Batch eWay (Connectivity Map) Properties

1. From the Connectivity Map, double click the eWay icon, located in the link between the associated External Application and the Service.
2. The eWay **Properties Editor** opens to the eWay Batch Connectivity Map parameters. Make any necessary modifications and click **OK** to save the settings.

Modifying the Batch eWay (Environment Explorer) Properties

1. From the Environment Explorer tree, right-click the Batch external system. Select **Properties** from the shortcut menu. The **Properties Editor** appears.
2. Make any necessary modifications to the Environment parameters of the Batch eWays, and click **OK** to save the settings.

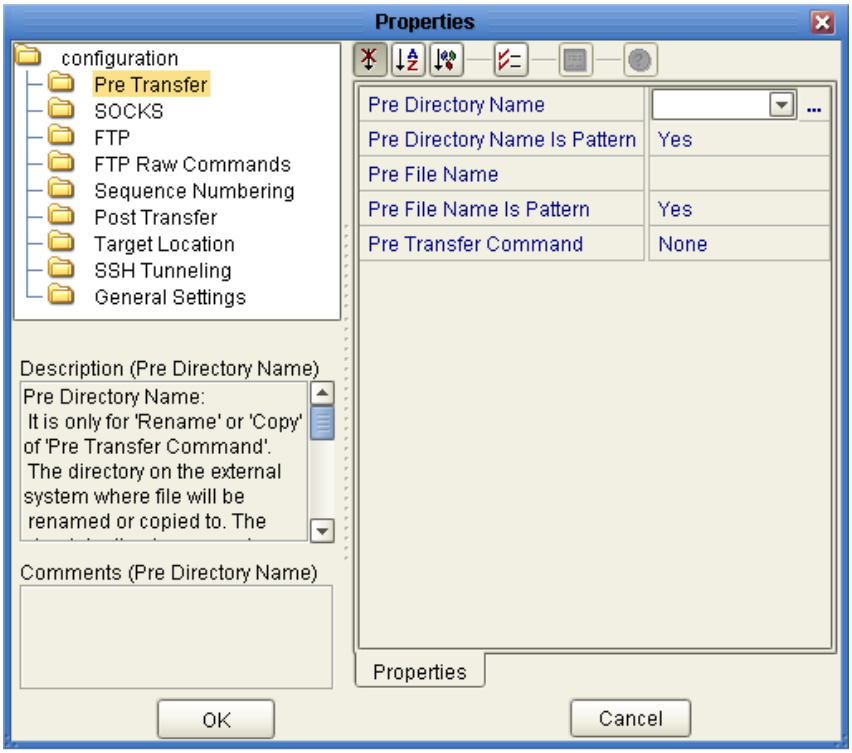
3.1.3 Using the Properties Editor

The Batch eWay properties are modified using the Batch eWay Properties Editor.

To modify the default eWay configuration properties

1. Open the Properties Editor for a Batch eWay (for this example, BatchFTP Connectivity Map Properties.
2. From the upper-right pane of the Properties Editor, select a subdirectory of the configuration directory (for this example, select the Pre-Transfer subdirectory). The editable properties contained in that subdirectory are now displayed in the Properties pane (see **Figure 3 on page 30**).

Figure 3 Batch eWay Properties Editor



3 Click on any property field to make it editable. For example, click on the **Pre Directory Name** parameter. You can now type the Pre Directory Name value into the property field. If a parameter's value is true/false or multiple choice, the field reveals a submenu of property options.

4 Click on the ellipsis (. . .) in the properties field to open a separate configuration dialog box. This is helpful for large values that cannot be fully displayed in the parameter's property field. Enter the property value in the dialog box and click **OK**. The value is now displayed in the parameter's property field.

5 A description of each parameter is displayed in the **Description** box when that parameter is selected, providing an explanation of any required settings or options.

6 The **Comments** box, located below the Description box, provides an area for recording notes and information regarding the currently selected parameter. This is saved for future referral.

7 After modifying the configuration properties, click **OK** to close the Properties Editor and save your current changes.

8 After modifying the configuration properties, click **OK** to close the Properties Editor and save the changes.

Note: *Properties set from the Collaboration override the corresponding properties in the eWay's configuration file. Any properties that are not set from the Collaboration retain their configured default settings.*

3.2 Batch eWay Properties

The Batch eWay's Properties are organized as follows:

3.3 BatchFTP eWay Connectivity Map Properties

This section describes the configuration parameters for the **BatchFTP OTD,** accessed from the Connectivity Map.

The BatchFTP Connectivity Map properties include the following sections:

Caution: *Several of these configuration options allow you to use regular expressions. This advanced feature is useful but must be used carefully. An improperly formed regular expression can cause the creation of undesired data or even the loss of data. You must have a clear understanding of regular-expression syntax and construction before attempting to use this feature. It is recommended that you test such configurations thoroughly before moving them to production.*

3.3.1 Pre Transfer (BatchFTP Connectivity Map)

Pre-transfer operations are those performed before the file transfer. *For more information, see* **"Pre/Post File Transfer Commands" on page 165**.

The **Pre Transfer** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 3.

Table 3 Connectivity Map - BatchFTP - Pre Transfer

Name	Description	Required Value
Pre Directory Name	Specifies the directory name (path) on the external system to which a file is renamed or copied. The value can be a literal or a pattern name. This setting is only for the **Rename** or **Copy** operations of the **Pre Transfer Command** parameter. For outbound transfers, the directory is created if it does not already exist. See **"Pre Directory Name Is Pattern" on page 33**. See **"Using Name Patterns" on page 178**.	Enter the exact name of the directory (with the path), enter a pattern name, or select one of the following values: ▪ %f ▪ %f.%y%y%y%y%M%M%d%d.%h%h%m%m%s%s%S%S%S ▪ %f.copy ▪ %f.rename
Pre Directory Name Is Pattern	Specifies whether the directory name is interpreted as literal or as a name pattern, as follows: ▪ **No**: indicates that the name entered is a literal, an exact match. ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern. See **"Pre Directory Name" on page 33**.	Select **Yes** or **No**. The configured default is **Yes**.
Pre File Name	Specifies the file name on the external system, to which a file is renamed or copied. The value represents the file name. The value can be a literal or pattern name. This setting is only for the **Rename** or **Copy** operations of the **Pre Transfer Command** parameter. Special characters are allowed, for example, the pattern **%f** indicates the original working file name. See **"Pre File Name Is Pattern" on page 34**. See **"Using Name Patterns" on page 178**.	Enter the exact name of the file, enter a pattern name, or select one of the following values: ▪ %f ▪ %f%# ▪ %f.%y%y%y%y%M%M%d%d.%h%h%m%m%s%s%S%S%S ▪ %f.copy ▪ %f.rename

Table 3 Connectivity Map - BatchFTP - Pre Transfer (Continued)

Name	Description	Required Value
Pre File Name Is Pattern	Specifies whether the file name represents a literal or a name pattern, as follows: ▪ **No**: indicates that the name entered is a literal, an exact match. No pattern matching or name expansion is done. ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern. See **"Pre File Name" on page 33**.	Select **Yes** or **No**. The configured default is **Yes**.
Pre Transfer Command	Allows you to execute a desired action directly before the actual file transfer. For an inbound transfer, the file can be made unavailable to other clients polling the target system with the same directory and file pattern or name. For an outbound transfer, you can perform an automatic backup or clean-up of the existing files. The options are: ▪ **Rename**: Rename the target file for protection or recovery. ▪ **Copy**: Copy the target file for backup or recovery. ▪ **None**: Do nothing. To gain proper protection, backup, or recovery, you must choose the appropriate setting that serves your purpose. For example, to recover from failures on an outbound appending transfer, use the **Copy** setting. ***Caution:*** When you are using **Rename**, if the destination file exists, different FTP servers can behave differently. For example, on some UNIX FTP servers, the destination file is overwritten without question. That is, no error or warning message is given. On other FTP servers, a Windows XP server for example, the system generates an error that results in exceptions being thrown in the called OTD method. Be sure you are familiar with the native behavior of the corresponding FTP server. If you are in doubt, try the action at the command prompt. If the action displays an error message, it may result in an exception being thrown in the Collaboration.	Select **Rename**, **Copy**, or **None**. The configured default is **None**. ***Note:*** The **Copy** option could slow system performance, especially if you are copying a large file.

3.3.2 SOCKS (BatchFTP Connectivity Map)

The BatchFTP SOCKS supports two negotiation methods: NO-AUTHENTICATION and USER/PASSWORD. For more information on SOCKS, see **"SOCKS" on page 188**.

The **SOCKS** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 4.

Table 4 Connectivity Map - BatchFTP - SOCKS

Name	Description	Required Value
Socks Enabled	Specifies whether the FTP command connection goes through a SOCKS server. If you choose **No**, the eWay does not connect to a SOCKS server. In this case, all other parameters under the **SOCKS** section are ignored. ***Note:*** *If this parameter is set to* **Yes**, *the host name under the* **FTP** *configuration could fail to resolve some names, such as* **localhost** *or* **127.0.0.1** *correctly. Use real IP or machine names to represent the hosts. See* **"Host Name" on page 54** *for more details.*	Select **Yes** or **No**. The configured default is **No**.
Socks Version	Specifies the SOCKS server version. If you choose **Unknown**, the eWay detects the actual version for you. ***Note:*** For the best performance, specify the version number, 4 or 5.	Select **4**, **5**, or **Unknown.** The configured default is **Unknown**.

3.3.3 FTP (BatchFTP Connectivity Map)

The **FTP** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 5.

Table 5 Connectivity Map - BatchFTP - Pre Transfer

Name	Description	Required Value
Command Connection Timeout	Allows you to set the timeout of the FTP command/control connection socket. Normally, the larger the file you are transferring, the higher this value must be. Of course, the quality of the network connection also affects this setting. The value is in milliseconds. A timeout of zero is interpreted as an infinite timeout.	An integer from **0** to **2147483647**. The configured default is **45000**.
Data Connection Timeout	Allows you to set the timeout of the FTP data connection socket. Normally, a slow or busy network connection requires a higher timeout setting. The value is in milliseconds. A timeout of zero is interpreted as an infinite timeout. For setting the timeout of the command/control connection socket, see the parameter **Command Connection Timeout**.	An integer from **0** to **2147483647**. The configured default is **45000**.

Table 5 Connectivity Map - BatchFTP - Pre Transfer (Continued)

Name	Description	Required Value
Directory Listing Style	Specifies the system that reflects the remote host. This parameter is used to determine the format in which the **LIST** command returns file-listing information. The Directory Listing Style values include User Defined1 - User Defined10 values. These user defined properties allow you to create multiple user-defined FTP heuristic configurations, and make these selectable from the BatchFTP eWay properties. You can create corresponding heuristic configurations in the FtpHeuristics.cfg file under the User Defined sections. For more information on setting user defined FTP heuristic properties, see **"Modifying the FTP Heuristics Configuration File" on page 133**). ***Note:*** This property is superseded by any value specified in the User Defined Directory Listing Style property (see **"User Defined Directory Listing Style." on page 38**). The User Defined Directory Listing Style property value must be blank (empty) to enable the Directory Listing Style property.	One of the following values: ♦ UNIX ♦ AS400 ♦ AS400-UNIX ♦ HCLFTPD 6.0.1.3 ♦ HCLFTPD 5.1 ♦ HP NonStop/Tandem ♦ MPE ♦ MSFTPD 2.0 ♦ MSP PDS (Fujitsu) ♦ MSP PS (Fujitsu) ♦ MVS GDG ♦ MVS PDS ♦ MVS Sequential ♦ Netware 4.11 ♦ NT 3.5 ♦ NT 4.0 ♦ UNIX ♦ UNIX (EUC-JP ♦ UNIX (SJIS) ♦ User Defined ♦ User Defined (1-10) ♦ VM/ESA ♦ VMS ♦ VOS3 PDS (Hitachi) ♦ VOS3 PS (Hitachi) ♦ VOSK (Hitachi) For more information, see **"Using FTP Heuristics" on page 130**.

Table 5 Connectivity Map - BatchFTP - Pre Transfer (Continued)

Name	Description	Required Value
User Defined Directory Listing Style.	Specifies the name of a user-defined directory listing style (heuristics) that is available in the user-created FTP heuristics configuration file located on the logical host. This property works in conjunction with the properties **"Directory Listing Style" on page 37** and **"User Defined Heuristics Configuration File" on page 55**. For details on how to use the User Defined Directory Listing Style see**"Creating a Custom Heuristics Configuration File" on page 132** ***Note:*** The BatchFTP OTD will generate an exception if a selected User Defined Directory Listing Style or the User Defined Heuristics Configuration File path is not defined correctly. If a User Defined Directory Listing Style is specified, a corresponding value must also be provided for the User Defined Heuristics Configuration File property.	A text string value (default to blank) representing the directory listing style (heuristics) name which is defined in a user supplied heuristics configuration file.
Use PASV	Allows you to prompt the eWay to enter either the passive or active mode. Normally, when you connect to an FTP site, the site establishes the data connection to your computer. However, some FTP sites allow passive transfers, meaning that your computer establishes the data connection. By default, the passive mode is used. It is recommended that you use this mode for transfers to and from FTP sites that support it. The passive mode can be required in the following situations: ▪ For users on networks behind some types of router-based firewalls ▪ For users on networks behind a gateway requiring passive transfers ▪ If transfers are erratic ▪ If data-channel errors are prevalent in your environment	Select **Yes** or **No**. The configured default is **Yes**.

Table 5 Connectivity Map - BatchFTP - Pre Transfer (Continued)

Name	Description	Required Value
Mode	Specifies the mode used to transfer data to or from the FTP server, using the **Ascii, Binary,** or **Ebcdic** mode. If you choose Ebcdic, make sure of the following: ▪ Your FTP server supports the EBCDIC mode. ▪ You are processing EBCDIC data.	Select **Ascii, Binary,** or **Ebcdic.** The configured default is **Binary**.

3.3.4 FTP Raw Commands (BatchFTP Connectivity Map)

FTP raw commands are commands that are sent *directly* to the FTP server.

The **FTP Raw Commands** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 6.

Table 6 Connectivity Map - BatchFTP - FTP Raw Commands

Name	Description	Required Value
Post Transfer Raw Commands	Specifies the FTP raw commands to be used directly after the file-transfer command. For example, some SITE commands use a ; (semi-colon) to separate the command set, as displayed in this example: `SITE RECFM=FB;SITE LRECL=50;SITE BLOCKSIZE=32750;SITE TRACKS;SITE PRI=5;SITE SEC=5` These commands are sent one by one, in the sequence they are listed. ***Caution:*** Certain combinations of post-transfer raw commands can cause the loss of data if there is a failure on the FTP server. For example, if the inbound post-transfer command is **Delete**, and your post-transfer raw commands fail, the deleted file is not recoverable.	One or more valid FTP raw commands. ***Note:*** These commands are sent to the FTP server directly and are not interpreted by the eWay in any way.
Pre Transfer Raw Commands	Specifies the FTP raw commands to be used directly before the file-transfer command. For example, some SITE commands use a ; (semi-colon) to separate the command set: `SITE RECFM=FB;SITE LRECL=50;SITE BLOCKSIZE=32750;SITE TRACKS;SITE PRI=5;SITE SEC=5` These commands are sent one by one, in the sequence they are listed.	One or more valid FTP raw commands. ***Note:*** These commands are sent to the FTP server directly and are not interpreted by the eWay in any way.

3.3.5 Sequence Numbering (BatchFTP Connectivity Map)

The **Sequence Numbering** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 7.

Note: *The Synchronized property, under General Settings, must be set to "Yes" to use Sequence Numbering.*

Table 7 Connectivity Map - BatchFTP - Sequence Numbering

Name	Description	Required Value
Max Sequence Number	Use this parameter when you have set up the target directory or file name to contain a sequence number. It tells the eWay that when this value (the **Max Sequence Number**) is reached, to reset the sequence number to the **Starting Sequence Number** value. This parameter is used for the name pattern **%#**. See **"Using Name Patterns" on page 178**.	An integer from **1** to **2147483647**. The value of **Max Sequence Number** *must* be greater than that of **Starting Sequence Number**.
Starting Sequence Number	Use this parameter when you have set up the target directory or file name to contain a sequence number. It tells the eWay which value to start with in the absence of a sequence number from the previous run. This parameter is used for the name pattern **%#**. When the **Max Sequence Number** value is reached, the sequence number rolls over to the **Starting Sequence Number** value. See **"Using Name Patterns" on page 178**.	An integer from **0** to **2147483647**. The value of the **Starting Sequence Number** *must* be less than the **Max Sequence Number** value.

3.3.6 Post Transfer (BatchFTP Connectivity Map)

Post-transfer operations are those performed on remote (ftp) site after the real ftp transfer. For more information on this feature, see **"Pre/Post File Transfer Commands" on page 165**.

The **Post Transfer** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 8.

Table 8 Connectivity Map - BatchFTP - Post Transfer

Name	Description	Required Value
Post Directory Name	Specifies the directory name (path) on the external system to which a file is renamed. The value can be a literal or pattern name. For an outbound transfer (to destination), the directory is created if it does not already exist. This setting is only for the **Rename** operation of the **Post Transfer Command** parameter. Special characters are allowed, for example, the pattern **%f** indicates the original working directory name. The expansion of any special characters is carried out each time this parameter is used. See **"Post Directory Name Is Pattern" on page 42**. See **"Using Name Patterns" on page 178**.	Enter the exact name of the directory (with the path), enter a pattern name, or select one of the following values: ▪ %f ▪ %f%# ▪ %f.%y%y%y%y%M%M %d%d.%h%h%m%m %s%s%S%S%S ▪ %f.rename
Post Directory Name Is Pattern	Specifies whether the pattern entered for the directory represents a literal or a name pattern, as follows: ▪ **No**: indicates that the name entered is a literal, an exact match. ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern. See **"Post Directory Name Is Pattern" on page 42**.	Select **Yes** or **No**. The configured default is **Yes**.

Table 8 Connectivity Map - BatchFTP - Post Transfer (Continued)

Name	Description	Required Value
Post File Name	Specifies the file name to which a file on an external system is renamed. The value represents the file name. The value can be a literal, or pattern name. This setting is only for **Rename** operation of **Post Transfer Command** parameter. Special characters are allowed. For example, the pattern **%f** indicates the original working file name. See **"Post File Name Is Pattern" on page 43**. See **"Using Name Patterns" on page 178**.	Enter the exact name of the file, enter a pattern name, or select one of the following values: ▪ %f ▪ %f.%y%y%y%y%M%M %d%d.%h%h%m%m %s%s%S%S%S ▪ %f.rename
Post File Name Is Pattern	Specifies whether the pattern entered for the file name is interpreted as literal or as a name pattern , as follows: ▪ **No**: indicates that the name entered is literal, an exact match. No pattern matching or name expansion is done. ▪ **Yes**: indicates that the name value you enter is a name pattern. See **"Post File Name" on page 43**.	Select **Yes** or **No**. The configured default is **Yes**

Table 8 Connectivity Map - BatchFTP - Post Transfer (Continued)

Name	Description	Required Value
Post Transfer Command	Allows you to execute a desired action directly after the actual file transfer or during the "commit" phase. For an inbound transfer, you can mark the transferred file as "consumed" by making an automatic backup (**Rename**) or by destroying it permanently (**Delete**). For an outbound transfer, you can make the transferred file available to other clients by renaming it. The options are: ▪ **Rename**: Rename the transferred file. ▪ **Delete**: Delete the transferred file (inbound transfers only). ▪ **None**: Do nothing. ***Caution:*** When you are using **Rename**, if the destination file exists, different FTP servers can behave differently. For example, on some UNIX FTP servers, the destination file is overwritten without question. That is, no error or warning message is given. On other FTP servers, a Windows XP server for example, the system generates an error that results in exceptions being thrown in the called OTD method. Be sure you are familiar with the native behavior of the corresponding FTP server. If you are in doubt, try the action at the command prompt. If the action displays an error message, it is likely to result in an exceptions being thrown in the Collaboration.	**Select Rename**, **Delete**, or **None**. The configured default is **None**.

3.3.7 Target Location (BatchFTP Connectivity Map)

The **Target Location** section allows you to configure the parameters for the **Target Location** (remote location) of the FTP directories and files.

The **Target Location** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 9.

Table 9 Connectivity Map - BatchFTP - Target Location

Name	Description	Required Value
Append	Specifies whether to overwrite or append the data to the existing file. Use this parameter for outbound FTP transfers only. Choose the appropriate setting as follows: ▪ If you select **Yes** and the target file already exists, the data is appended to the existing file. ▪ If you select **No**, the eWay overwrites the existing file on the remote system. If a file with the same name does not exist, both **Yes** and **No** create a new file on the external host.	**Select Yes** or **No.** The configured default is **No**.
Target Directory Name	Specifies the directory on the external system from which files are retrieved or sent. The directory name and path is preferred, otherwise, the path is relative to your home directory when you log on to the FTP server. The value can be a literal, regular expression (source), or pattern name (destination). For outbound FTP operations (destination), the directory is created if it does not already exist. See **"Target Directory Name Is Pattern" on page 45**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A directory name and path on the target external system.
Target Directory Name Is Pattern	Specifies whether the directory name is represented as literal, or as a regular expression or name pattern, as follows: ▪ **No**: indicates that the name entered is a literal, an exact match. ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. See **"Target Directory Name" on page 45**.	**Select Yes** or **No.** The configured default is **No**.

Table 9 Connectivity Map - BatchFTP - Target Location (Continued)

Name	Description	Required Value
Target File Name	Specifies the name of the remote FTP file to be retrieved or sent. The value can be a literal, regular expression (get), or pattern name (put). For MVS GDG systems, the target file name can be the version of the data set, for example: ▪ Target directory name = '**STC.SAMPLE.GDGSET**' ▪ Target file name = (**0**) to indicate the current version See **"Target File Name Is Pattern" on page 46**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	For inbound: a literal file name or a regular expression. For outbound: a literal file name or name pattern.
Target File Name Is Pattern	Specifies whether the target file name represents a literal, or as a regular expression or name pattern, as follows: ▪ **No**: indicates that the name entered is a literal, an exact match. ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. See **"Target File Name" on page 46**.	Select **Yes** or **No.** The configured default is **Yes**.

3.3.8 SSH Tunneling (BatchFTP Connectivity Map)

The **SSH Tunneling** section provides information for configuring the **SSH Tunneling** properties. If Secure FTP (FTP over SSH or FTP over SSL) is required, use the Secure FTP OTDs (**BatchFTPOverSSL**, **BatchSFTP**, and **BatchSCP**).

The **SSH Tunneling** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 10.

Table 10 Connectivity Map - BatchFTP - SSH Tunneling

Name	Description	Required Value
SSH Channel Established	Specifies whether the eWay needs to launch an SSH subprocess. Selecting **No** indicates that the SSH channel has not yet been established. The eWay spawns a subprocess internally then establishes the channel on your behalf. If you select **No**, you must set the following parameters: ▪ **SSH Command Line** ▪ **SSH Listen Port** (Environment property) If you select **No**, setting the following parameters is optional: ▪ **SSH User Name** (Environment property) ▪ **SSH Password** (Environment property) Selecting **Yes** indicates that an SSH channel has already been established. That is, the channel has already been started outside the eWay, and the eWay does not need to establish it. For example, you could have issued a command outside of eGate, or you could know that another Batch eWay instance has already established the channel by the time this eWay runs. If you select **Yes**, you must set the following parameters: ▪ **SSH Listen Host** (Environment property) ▪ **SSH Listen Port** (Environment property)	**Select Yes** or **No.** The configured default is **No**.

Table 10 Connectivity Map - BatchFTP - SSH Tunneling (Continued)

Name	Description	Required Value
SSH Command Line	Specifies the command line used to establish an SSH channel. This parameter is required only when you set the **SSH Channel Established** parameter to **No**. This entry must be the complete, correct command line required by the additional software application you are using to support SSH tunneling. This command line is executed as it is, so you must be sure of the following: ▪ It contains all the necessary arguments ▪ The syntax is correct ▪ It is compliant with your SSH-environment To verify these requirements, test this command line manually outside of eGate to make sure it works correctly. Execute the command line from the shell and ensure that it does not prompt for any additional user input. If it does, continue to add whatever additional parameters are required until it no longer prompts for additional input, then use that command line in the eWay's configuration. You can specify any other options that are based on your SSH-environment. However, if you do so, you must still be sure this command line is correct and complete. For example, port forwarding could be specified using the following command-line option: -L **ListenPort:FtpServerHost:FtpServerPort** In this example, ***ListenPort*** must be the same value as that given for the parameter **SSH Listen Port**. The value given for **FtpServerHost** overwrites the parameter setting for **Host Name** under the **FTP** parameters. The value given for **FtpServerPort** overwrites the parameter setting for **Server Port** under the **FTP** parameters. All other settings under the **FTP** parameters operate for the specified FTP server: **FtpServerHost:FtpServerPort**. If the SSH channel established by an SSH command line must be shared by other Batch eWay instances located on different eGate client hosts, you must configure SSH port forwarding to allow non-local connections from other hosts. For some SSH clients, you can use the option **-g**. ***Note:*** You can also specify port forwarding in your SSH configuration file. (*Continued on the next page*)	A valid SSH command line.

Table 10 Connectivity Map - BatchFTP - SSH Tunneling (Continued)

Name	Description	Required Value
SSH Command Line (continued)	(*Continued from last page*) The command-line syntax can differ, depending on the type of SSH client implementation you are using. See your SSH-tunneling support software user documentation for details. Examples: ssh -L 3456:ftp.sun.com:21 -o BatchMode=yes apple ssh -L 4567:apple:21 -o BatchMode=yes apple ssh -L 5678:orange:21 -o BatchMode=yes apple ssh -L 6789:orange:21 -g -o BatchMode=yes apple plink -L 4567:apple:21 apple plink -L 5678:orange:21 apple plink -L 6789:orange:21 -g apple	
SSH Tunneling Enabled	Specifies whether the FTP command connection is secured through an SSH tunnel. If you choose **No**, all other parameters in this section are ignored. ***Note:*** If you want to use the SSH port-forwarding feature, you may need to reconfigure your FTP server, depending on what kind of server you are using and how it is currently configured. See your SSH documentation for more information.	**Select Yes** or **No.** The configured default is **No**.

Additional SSH-supporting Software

The eWay's SSH tunneling (also known as port forwarding) feature utilizes additional existing SSH-supporting software applications, for example, Plink on Windows or OpenSSH on UNIX (see **"Additional Software Requirements" on page 190**

For different SSH client implementations, the command syntax and environment configuration may vary. See your SSH-supporting application's user guide for details.

Port-forwarding Configuration

SSH tunneling provides secure FTP command connections. This mechanism is based on an existing SSH port-forwarding configuration. You must configure SSH port forwarding on the *SSH listen host* before you configure the supporting eWay Connection.

For example, on the eGate client host **localhost**, you can issue a command, such as:

```
ssh -L 4567:apple:21 -o BatchMode=yes apple
```

Under the eWay's configuration for the previous example, you must specify:

- **localhost** for the Environment parameter **SSH Listen Host**
- **4567** for the Environment parameter **SSH Listen Port**

In this case, the eWay connects to the FTP server **apple:21** through an SSH tunnel. For more information on SSH tunneling, see **"SSH Tunneling Support" on page 189**.

Note: *It is possible to use SOCKS and SSH tunneling at the same time. However, this practice is not recommended.*

3.3.9 General Settings (BatchFTP Connectivity Map)

The **General Settings** section of the BatchFTP Connectivity Map properties contains the top-level parameters displayed in Table 11.

Table 11 Connectivity Map - BatchFTP - General Settings

Name	Description	Required Value
Synchronized	Specifically applies to legacy Batch eWay Projects. Provides backward compatibility to allow Projects that were created using the Batch eWay version 5.0.7 or earlier to be imported and deployed without a change in the eWays behavior. The selections are: ▪ **Yes**: Provides backward compatibility for legacy (pre-5.0.8 Batch eWay) Projects. The eWay run in synchronized mode, one instance of the Collaboration after the other. ▪ **No**: For use with new Batch eWay Projects. The eWay run in parallel, creating multiple instances of the Collaboration that run in parallel. All OTD instances used in a Project should have the same value for this property. ***Note:*** Synchronized must be set to "Yes" to use Sequence Numbering.	**Yes** or **No.** The default setting is **Yes, simulating Projects created with** Batch eWay version 5.0.7 or earlier.

3.4 BatchFTP eWay Environment Properties

This section describes the configuration properties for the **BatchFTP OTD accessed from the Environment Explorer tree**.

The BatchFTP Environment Explorer properties include the following sections:

Caution: *Several of these configuration options allow you to use regular expressions. This advanced feature is useful but must be used carefully. An improperly formed regular expression can cause the creation of undesired data or even the loss of data. You must have a clear understanding of regular-expression syntax and construction before attempting to use this feature. It is recommended that you test such configurations thoroughly before moving them to production.*

3.4.1 SOCKS (BatchFTP Environment)

This section provides information for configuring the **SOCKS** properties (accessed from the Environment Explorer). The BatchFTP eWay supports the negotiation methods, **No-authentication** and **User/password**. For more information on SOCKS, see **"SOCKS" on page 188**.

The **SOCKS** section of the BatchFTP Environment properties contains the top-level parameters displayed in Table 12.

Table 12 Environment - BatchFTP - SOCKS

Name	Description	Required Value
Socks Host Name	Specifies the SOCKS server (host) name. If you are communicating with a SOCKS server enter the SOCKS server name in this parameter.	The name of the SOCKS server.
Socks Server Port	Specifies the port number to use on the SOCKS server, when connecting to it.	An integer from **1** to **65,535**. The configured default is **1080**.
Socks User Name	Specifies the user name to use (together with the password specified under the **Socks Password** parameter) for authentication with a SOCKS5 server, if necessary. This parameter is used for the user/password negotiation method.	A valid SOCKS5 user name.

Table 12 Environment - BatchFTP - SOCKS (Continued)

Name	Description	Required Value
Socks Password	Specifies the password to use (together with the user name specified under the **Socks User Name** parameter) for authentication with a SOCKS5 server, if necessary. This parameter is used for the user/password negotiation method. ***Note:*** The corresponding Java accessors are **getSocksPassword()**, **setSocksPassword(java.lang.String p)** and **setSocksEncryptedPassword(java.lang.String p)**.	A valid SOCKS5 password.

3.4.2 FTP (BatchFTP Environment)

The **FTP** section of the BatchFTP Environment properties contains the top-level parameters displayed in Table 13.

Table 13 Environment - BatchFTP - FTP

Name	Description	Required Value
Host Name	Specifies the name of the external system that the eWay connects to. If the parameter **SSH Tunneling Enabled** under the **SSH Tunneling** configuration settings is set to **Yes**, the parameters **Host Name** and **Server Port,** under the FTP settings, are ignored. In this case, the FTP host name is determined by an SSH option, according to the following model: ssh -L **ListenPort:FtpServerHost:FtpServerPort SSHServer** In the previous example, the FTP feature communicates with the FTP server ***FtpServerHost:FtpServerPort*** using an existing SSH tunnel. See **"SSH Tunneling (BatchFTP Connectivity Map)" on page 47** for details. If the parameter **Socks Enabled** under the **SOCKS** configuration parameters is set to **Yes**, the host name under the **FTP** configuration could fail to resolve some names, for example, **localhost** or **127.0.0.1** correctly. Use real IP or machine names to represent the hosts. See **"SOCKS (BatchFTP Connectivity Map)" on page 35** for details.	The host name.
Server Port	Specifies the port number to use on the FTP server when connecting to it. If the parameter **SSH Tunneling Enabled** under the **SSH Tunneling** configuration is set to **Yes**, the parameters **Host Name** and **Server Port** under the **FTP** configuration are ignored. In this case, the FTP server port number is determined by an SSH option, according to the following model: **ssh -L ListenPort:FtpServerHost:FtpServerPort SSHServer** In the previous example, the FTP feature communicates with the FTP server ***FtpServerHost:FtpServerPort*** using an existing SSH tunnel. See **"SSH Tunneling (BatchFTP Connectivity Map)" on page 47** for details.	The server port number.

Table 13 Environment - BatchFTP - FTP (Continued)

Name	Description	Required Value
User Name	Specifies the user name used to log onto the external system, when required.	A user name that provides access to the external system.
Password	If a password is required to log on to an external system, enter the password that corresponds to the user name. The corresponding Java accessor methods are **getPassword()**, **setPassword()**, and **setEncryptedPassword()**.	The password.
User Defined Heuristics Configuration File	Specifies the name and location of the user defined FTP heuristics configuration file. The format of the files content must be in the same form as that of the FTPHeuristics.cfg file. See **"Creating a Custom Heuristics Configuration File" on page 132** for more details. This property works in conjunction with the property **"User Defined Directory Listing Style." on page 38** User Defined Directory Listing Style property ***Note:*** The BatchFTP OTD will generate an exception if a selected User Defined Directory Listing Style or the User Defined Heuristics Configuration File path is not defined correctly. If a User Defined Directory Listing Style is specified, a corresponding value must also be provided for the User Defined Heuristics Configuration File property.	The location and name of the user defined FTP heuristics configuration file on the local host.

3.4.3 General Settings (BatchFTP Environment)

The **General Settings** section of the BatchFTP Environment properties contains the top-level parameters displayed in Table 14.

Table 14 Environment - BatchFTP - General Settings

Name	Description	Required Value
State Persistence Base Location	Specifies a working directory for storing intermediary results. Options: ▪ Leave value blank: BatchFTP will use a default folder as the working directory. ▪ Specify a path to a local file system folder with read/write permissions. ***Note:*** See **"Deploying an EAR File" on page 24** if you are deploying the project to another application server.	A working directory with read/write permissions, or leave blank (no value) to accept a default directory.
Connection Mode	Specifies whether a physical connection is established when an external connection is instantiated. The options are: ▪ **Automatic**: Establishes a physical connection when an external connection is instantiated. ▪ **Manual**: Does not automatically establish a physical connection when an external connection is instantiated. If a physical connection is not automatically established, a physical connection must be established from the Collaboration (for example, by calling the **connect()** method).	Select **Automatic** or **Manual**. The configured default is Automatic.

3.4.4 SSH Tunneling (BatchFTP Environment)

This section provides information for configuring the **SSH Tunneling** properties (accessed from the Environment Explorer). If Secure FTP (FTP over SSH or FTP over SSL) is required, use the Secure FTP OTDs (**BatchFTPOverSSL**, **BatchSFTP**, and **BatchSCP**). SSH Tunneling is supported for compatibility purposes.

The **SSH Tunneling** section of the BatchFTP Environment properties contains the top-level parameters displayed in Table 15.

Table 15 Environment - BatchFTP - SSH Tunneling

Name	Description	Required Value
SSH Listen Host	Specifies the name of the host where the SSH support software runs, and to which the host listens. This parameter is required when and only when **SSH Channel Established** is set to **Yes**. The reason for this is, if you choose **No** this Listen Host will always be **localhost** because the SSH client will always be started from localhost. For optimum security, it is recommended that you use **localhost** as your choice. The connection to the corresponding port number on this host is forwarded to the FTP server through an SSH-secure channel. On the listen host, the SSH support software must be configured and started with the **Port-Forwarding** option. The FTP command connection is forwarded through the secure tunnel. The corresponding SSH command uses the following model: ssh -L **ListenPort:FtpServerHost:FtpServerPort** -o BatchMode=yes **SSHServer** If this host name is not **localhost**, the data transport between the local host and the SSH listen host is not secure. Also, your SSH support software should be configured to allow connections to other hosts (for some SSH clients, it is an option **-g**). For example, on an SSH listen host, you could issue a command, such as: ssh -L 4567:apple:21 -o BatchMode=yes apple or ssh -L 5678:orange: 21 -o BatchMode=yes apple Regardless, the transport between the SSH listen host and the FTP server is still secure.	The SSH listen host name. The configured default is **localhost.**

Table 15 Environment - BatchFTP - SSH Tunneling (Continued)

Name	Description	Required Value
SSH Listen Port	Specifies the port number that the SSH-tunneling support software uses to check for incoming connections. This port number can be any unused port number on the SSH listen host. The connection to this port is forwarded to the FTP server through an SSH-secure channel. This parameter is required and it must be exactly same as the **ListenPort** value in the SSH command you issue either inside or outside the eGate system. The corresponding SSH command line uses the following model: ssh -L **ListenPort:FtpServerHost:FtpServerPort** -o BatchMode=yes **SSHServer Required Values**	An integer from **1** to **65535**.
SSH User Name	Specifies an SSH user name. This parameter can be required when the setting for the **SSH Channel Established** parameter is **No**. This parameter is required only if the SSH support software is started from within the eWay (refer to the corresponding SSH command line). Even then, it is only required if your SSH implementation executes in an interactive way that requires you to enter a user name. Again, this requirement depends on how you specify the SSH command line and how your SSH environment is configured.	The SSH user name.
SSH Password	Specifies an SSH password corresponding to the user name entered under **SSH User Name**. This parameter can be required only when the setting for the **SSH Channel Established** parameter is **No**. For more information, see **SSH User Name**.	The SSH password.

3.4.5 Connection Pool Settings (BatchFTP Environment)

The **Connection Pool Settings** section of the BatchFTP Environment properties contains the top-level parameters displayed in Table 16.

Table 16 Environment - BatchFTP - Connection Pool Settings

Name	Description	Required Value
Steady pool size	Specifies the minimum number of physical EIS connections that the pool keeps available at all times.	An integer indicating the maximum number of connections available at all times. A value of **0** (zero) indicates that there are no physical connections in the pool and that new connections are created as needed. The configured default is **2**.
Maximum pool size	**Specifies the maximum number of physical EIS connections the pool contains.**	An integer indicating the maximum number of connections allowed. A value of **0 (zero) indicates that there is no maximum**. The default value is **10**.
Max Idle Timeout In Second	**Specifies the maximum idle timeout (in seconds)**. This is a hint to the server. A timer thread periodically removes unused connections. This parameter defines the interval at which this thread runs. This thread removes unused connections after the specified idle time expires. It allows the user to specify the maximum amount of time that a connection can remain in the pool. .	An integer indicating **the maximum idle timeout in seconds**. When this is set to a number greater than **0** (zero), the container removes or destroys any connections that are idle for the specified duration. A value of **0** specifies that idle connections can remain in the pool indefinitely. The configured default is **300** (5 minutes).

3.4.6 Connection Retry Settings (BatchFTP Environment)

The **Connection Retry Settings** section The **Connection Pool Settings** section of the BatchFTP Environment properties contains the top-level parameters displayed in Table 17.

Table 17 Environment - BatchFTP - Connection Retry Settings

Name	Description	Required Value
Connection Retries	Specifies the number of retries to establish a connection upon failure to acquire a connection.	An integer indicating the maximum number of retries to establish a connection upon failure to acquire a connection. The Configured default value is **0**.
Connection Retry Interval	Specifies the length of time (in milliseconds) between each reattempt to access the destination file. This is used in conjunction with the Connection Retries setting.	An integer indicating length of the pause (in milliseconds). The configured default value is **1000** (1 second).

3.5 BatchFTPOverSSL eWay Connectivity Map Properties

The BatchFTPOverSSL eWay Connectivity Map properties include the following sections:

3.5.1 Pre Transfer (BatchFTPOverSSL Connectivity Map)

The **Pre Transfer** section allows user to customize the behaviors of protection/backup/recovery. This section describes the operation that will be performed on remote end or locally before the real file transfer.

The **Pre Transfer** section of the BatchFTPOverSSL Connectivity Map properties contains the top-level parameters displayed in Table 18.

Table 18 Connectivity Map - BatchFTPOverSSL - Pre Transfer

Name	Description	Required Value
Remote Dir Name	Specifies the directory and path on the remote external system where file the is renamed or copied. This is only for **Rename** or **Copy** of the **Remote Pre Command**. The value can be a literal, regular expression (source), or pattern name (destination). When specifying a destination directory, the directory is created if it doesn't already exist. Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. For example, the pattern **%f** means the original working directory name. See **"Remote Dir Name Is Pattern" on page 62**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A directory name and path location on the target system. Special characters are allowed.

Table 18 Connectivity Map - BatchFTPOverSSL - Pre Transfer (Continued)

Name	Description	Required Value
Remote Dir Name Is Pattern	Specifies whether the pattern entered for the directory represents a literal, or a name pattern or regular expression, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: Indicates that the name value entered represents a literal, an exact match. No pattern matching or name expansion is done. See **"Remote Dir Name" on page 61**.	Select **Yes** or **No**. The configured default is **Yes**.
Remote File Name	Specifies the file name on the external system, to which a file is renamed or copied. The value represents the file name without the path. This setting is only for the **Rename** or **Copy** operations of **Pre Transfer Command** parameter. The value can be a literal, regular expression (get), or pattern name (put). Special characters are allowed, for example, the pattern **%f** indicates the original working file name. The expansion of any special characters is carried out each time this parameter is used. See **"Remote File Name Is Pattern" on page 62**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A remote file name.
Remote File Name Is Pattern	Specifies whether the pattern entered for the file name represents a literal, or a name pattern or regular expression, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Remote File Name" on page 62**.	Select **Yes** or **No**. The configured default is **Yes**.

Table 18 Connectivity Map - BatchFTPOverSSL - Pre Transfer (Continued)

Name	Description	Required Value
Remote Pre Command	Allows you to execute a desired action directly before the actual file transfer. For an inbound transfer, the file can be made unavailable to other clients polling the target system with the same directory and file pattern or name. For an outbound transfer, you can perform an automatic backup/clean-up of the existing files. The options are: ▪ **Rename**: Rename the target file for protection or recovery. ▪ **Copy**: Copy the target file for backup or recovery. ▪ **None**: Do nothing. To gain proper protection, backup, or recovery, you must choose the appropriate setting that serves your purpose. For example, to recover from failures on an outbound appending transfer, use the **Copy** setting. ***Caution:*** When you are using **Rename**, if the destination file exists, different FTP servers can behave differently. For example, on some UNIX FTP servers, the destination file is overwritten without question. That is, no error or warning message is given. On other FTP servers, a Windows XP server for example, the system generates an error that results in exceptions being thrown in the called OTD method. Be sure you are familiar with the native behavior of the corresponding FTP server. If you are in doubt, try the action at the command prompt. If the action displays an error message, it may result in an exception being thrown in the Collaboration.	Select **Rename**, **Copy**, or **None**. The configured default is **None**. ***Note:*** The **Copy** option could slow system performance, especially if you are copying a large file.

Table 18 Connectivity Map - BatchFTPOverSSL - Pre Transfer (Continued)

Name	Description	Required Value
Local Dir Name	Specifies the directory name (path) to be used by **Rename** or **Copy**. The value can be a literal, regular expression (source), or pattern name (destination). Special characters are allowed (name pattern). The expansion of any special characters is carried out each time this parameter is used. See **"Remote File Name Is Pattern" on page 62**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**. ***Note:*** When entering a path separator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir.	A directory name.
Local Dir Name Is Pattern	Specifies whether the Local Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local Dir Name" on page 64**.	Select **Yes** or **No**. The configured default is **No**.
Local File Name	Specifies the file name to be used by **Rename** or **Copy**. The value can be a literal, regular expression (get), or pattern name (put). Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**. See **"Local File Name Is Pattern" on page 65**.	A file name.

Table 18 Connectivity Map - BatchFTPOverSSL - Pre Transfer (Continued)

Name	Description	Required Value
Local File Name Is Pattern	Specifies whether the Local File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. See **"Local File Name" on page 64**.	Select **Yes** or **No**. The configured default is **No**.
Local Pre Command	Allows you to execute a desired action directly before the actual file transfer. For an inbound transfer, the file can be made unavailable to other clients polling the target system with the same directory and file pattern or name. For an outbound transfer, you can perform an automatic backup of the existing files. The options are: ▪ **Rename**: Rename the target file for protection or recovery. ▪ **Copy**: Copy the target file for backup or recovery. ▪ **None**: Do nothing. To gain proper protection, backup, or recovery, you must choose the appropriate setting that serves your purpose. For example, to recover from failures on an outbound appending transfer, use the **Copy** setting. ***Caution:*** Rename and Copy overwrite the file specified by the Local Dir Name and Local File Name properties, if they exist.	Select **Rename**, **Copy**, or **None**. The configured default is **None**. ***Note:*** The **Copy** option could slow system performance, especially if you are copying a large file.

3.5.2 FTP and SSL Settings (BatchFTPOverSSL Connectivity Map)

The **FTP and SSL Settings** section of the BatchFTPOverSSL Connectivity Map properties contains the top-level parameters displayed in Table 19.

Table 19 Connectivity Map - BatchFTPOverSSL - FTP and SSL Settings

Name	Description	Required Value
Secure Mode	Specifies the secure mode. Selections are: ▪ **None**: FTP is in clear text. ▪ **Implicit SSL**: The SSL handshake is started right after the socket connection is done. ▪ **Explicit SSL**: The SSL handshake is started by the client sending AUTH SSL/TLS FTP command.	Select **None**, **Implicit SSL**, or **Explicit SSL**. **None is the** configured default.
Directory Listing Style	Specifies the directory listing style of the FTP Server as **UNIX**, **NT**, or **MVS**. This provides a "hint" to the client side for parsing the directory listing response from the FTP Server.	Leave as **UNIX**. Currently the only supported option is UNIX. The configured default is **UNIX**.
Enabled Passive Mode	Specifies whether FTP passive mode is enabled.	**Select Yes** or **No**. **Yes** indicates that FTP passive mode is enabled. The configured default is **Yes**.
Transfer Mode	Specifies whether the transfer is binary code or ASCII text.	**Select BINARY or ASCII.** The configured default is **BINARY**.
Append	Specifies whether new data transferred to a remote server is appended to data that was previously transferred.	**Select Yes** or **No**. **Yes** indicates that data will be appended. The configured default is **No**.
Required Server Authentication	Specifies whether server authentication is required. The selections are: ▪ **Yes**: Indicates that server authentication is required, and that all parameters used for authentication (for example, **Key Store Location**, **Key Store Password**, **Key Store Type**, and so forth) must be set correctly so that the server certificate can be verified against the local trusted CA certificates. ▪ **No**: Indicates that server authentication is not required.	**Select Yes** or **No**. The configured default is **Yes**.
Distinguished Name for User	Specifies the distinguished name (DN) for the login user. This is imported from a CSR reply, and used to configure client authentication.	The Distingusihed Name, as in X.509.

Table 19 Connectivity Map - BatchFTPOverSSL - FTP and SSL Settings (Continued)

Name	Description	Required Value
Alias in Key Store	Specifies the alias for a key pair in a JKS type Key Store. This value is used to configure client authentication.	The alias.
Alias Password	Specifies the password that protects the key pair entry in the keystore, identified by the alias.	The alias password.
Remote Directory	Specifies a directory on the FTP server where data is sent or received. The accessibility of the directory usually depends on the login user. The value can be a literal, regular expression (source), or pattern name (destination). See **"Remote Directory Name is Pattern" on page 67**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The name of the remote directory.
Remote Directory Name is Pattern	Specifies whether the Remote Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. See **"Remote Directory" on page 67**.	Select **Yes** or **No.** The configured default is **No**.
Remote File	Specifies the file name on the remote server. ▪ The value can be a literal, regular expression (get), or pattern name (put). See **"Remote File Name is Pattern" on page 68**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The name of the remote file.

Table 19 Connectivity Map - BatchFTPOverSSL - FTP and SSL Settings (Continued)

Name	Description	Required Value
Remote File Name is Pattern	Specifies whether the Remote File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. See **"Remote File" on page 67**.	Select **Yes** or **No.** The configured default is **No**.
Local Directory	Specifies the local directory (path) for files that are sent to or received from a remote system. The value can be a literal, regular expression (source), or pattern name (destination). See **"Local Directory Name is Pattern" on page 68**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The local directory name.
Local Directory Name is Pattern	Specifies whether the Local Directory name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local Directory" on page 68**.	Select **Yes** or **No.** The configured default is **No**.
Local File	Specifies the local file name. The value can be a literal, regular expression (get), or pattern name (put). See **"Local File Name is Pattern" on page 69**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The local file name.

Table 19 Connectivity Map - BatchFTPOverSSL - FTP and SSL Settings (Continued)

Name	Description	Required Value
Local File Name is Pattern	Specifies whether the Local File name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local File" on page 68**.	Select **Yes** or **No.** The configured default is **No**.
Local File Overwrite	Specifies whether new data downloaded from the remote will overwrite existing data.	Select **Yes** or **No.** The configured default is **No**.

3.5.3 Post Transfer (BatchFTPOverSSL Connectivity Map)

The **Post Transfer** section of the BatchFTPOverSSL Connectivity Map properties contains the top-level parameters displayed in Table 20.

Table 20 Connectivity Map - BatchFTPOverSSL - Post Transfer

Name	Description	Required Value
Remote Dir Name	Specifies the directory name (path) on the remote external system where the file the will be renamed or copied. This is only for **Rename** or **Copy** of the Post Transfer Command. The value can be a literal, regular expression (source), or pattern name (destination). For outbound (destination), the directory is created if it doesn't already exist. Special characters are allowed. For example, the pattern **%f** means the original working directory name. The expansion of any special characters is carried out each time this parameter is used. See **"Remote Dir Name Is Pattern" on page 70**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A directory name and path on the external system. Special characters are allowed.
Remote Dir Name Is Pattern	Specifies whether the Remote Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Remote Dir Name" on page 70**.	Select **Yes** or **No**. The configured default is **Yes**.

Table 20 Connectivity Map - BatchFTPOverSSL - Post Transfer (Continued)

Name	Description	Required Value
Remote File Name	Specifies the file name on the external system, to which a file is renamed or copied. This setting is only for the **Rename** or **Copy** operations of **Post Transfer Command** parameter. The value can be a literal, regular expression (get), or pattern name (put). Special characters are allowed, for example, the pattern **%f** indicates the original working file name. The expansion of any special characters is carried out each time this parameter is used. See **"Remote File Name Is Pattern" on page 71**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The file name.
Remote File Name Is Pattern	Specifies whether the Remote File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. See **"Remote File Name" on page 71**.	Select **Yes** or **No**. The configured default is **Yes**.
Remote Post Command	Allows you to execute a desired action directly after the actual file transfer. For an inbound transfer, it can be applied to mark the transferred file as consumed by making an automatic backup (**Rename**) or by destroying it permanently (**Delete**). For an outbound transfer, it can be applied to make the transferred file available to other clients by renaming it. ▪ **Rename**: Rename the transferred file. ▪ **Delete**: Delete the transferred file. ▪ **None**: Do nothing. Note: For **Rename**, if the destination file exists, different FTP servers may behave differently. For example, on some UNIX FTP servers, the destination file will be overwritten without extra message. On an NT FTP server, this will fail and get an exception. It does not define unified behavior, rather, it follows the native behavior of the corresponding FTP server.	Select **Rename**, **Delete**, or **None**. The configured default is **None**.

Table 20 Connectivity Map - BatchFTPOverSSL - Post Transfer (Continued)

Name	Description	Required Value
Local Dir Name	Specifies the directory name (path) to be used by **Rename**. The value can be a literal, regular expression (source), or pattern name (destination). Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. ***Note:*** For path separator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir. See **"Local Dir Name Is Pattern" on page 72**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The local directory name.
Local Dir Name Is Pattern	Specifies whether the Local Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local Dir Name" on page 72**.	Select **Yes** or **No**. The configured default is **No**.
Local File Name	Specifies the file name to be used by **Rename**. The value can be a literal, regular expression (get), or pattern name (put). Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. ***Note:*** For path seperator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir. See **"Local File Name Is Pattern" on page 73**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A file name.

Table 20 Connectivity Map - BatchFTPOverSSL - Post Transfer (Continued)

Name	Description	Required Value
Local File Name Is Pattern	Specifies whether the Local File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local File Name" on page 72**.	Select **Yes** or **No**. The configured default is **No**.
Local Post Command	Allows you to execute a desired action directly after the actual file transfer. For an inbound transfer, the target file can be marked as **consumed** by making an automatic backup (**Rename**) or by destroying it permanently (**Delete**). For an outbound transfer the target file can be made available to other clients by renaming it. The options are: ▪ **Rename**: Rename the target file. ▪ **Delete**: Delete the target file (inbound transfers only). ▪ **None**: Do nothing. ***Note:*** Rename overwrites the file specified by the Local Dir Name and Local File Name properties, if they exist.	Select **Rename**, **Delete**, or **None**. The configured default is **None**.

3.5.4 Firewall Settings (BatchFTPOverSSL Connectivity Map)

The **Firewall Settings** section of the BatchFTPOverSSL Connectivity Map properties contains the top-level parameters displayed in Table 21.

Table 21 Connectivity Map - BatchFTPOverSSL - Firewall Settings

Name	Description	Required Value
Use Firewall	Specifies whether you are using a firewall. If a firewall is used, supports **SOCKS 4** and **5**.	**Yes** or **No**. **Yes** indicates that you are using a firewall. The configured default is **No**.
SOCKS version	Specifies the SOCKS version of the firewall. The supported options are **4** for SOCKS version 4, or **5** for SOCKS version **5.**	Select **4** for **SOCKS version 4**, or **5** for **SOCKS version 5**.

3.5.5 Synchronization (BatchFTPOverSSL Connectivity Map)

The **Synchronization** section of the BatchFTPOverSSL Connectivity Map properties contains the top-level parameters displayed in Table 22.

Table 22 Connectivity Map - BatchFTPOverSSL - Synchronization

Name	Description	Required Value
Synchronized	Specifies whether the eWay simulates the pre-version 5.1 eWay behavior in which the eWay runs synchronized or in parallel. The selections are: ▪ **Yes**: The eWay run in synchronized mode, one instance of the Collaboration after the other. ▪ **No**: The eWay run in parallel, creating multiple instances of the Collaboration that run in parallel. ***Note:*** All OTD instances used in a Project should have the same value for this property.	Select **Yes** or **No.** The configured default is **No**.

3.6 BatchFTPOverSSL eWay Environment Properties

This section describes the configuration properties for the **BatchFTPOverSSL OTD,** accessed from the Environment Explorer.

The BatchFTPOverSSL eWay Environment properties include the following sections:

3.6.1 FTP and SSL Settings (BatchFTPOverSSL Environment)

The **FTP and SSL Settings** section of the BatchFTPOverSSL Environment properties contains the top-level parameters displayed in Table 11.

Table 23 Environment - BatchFTPOverSSL - FTP and SSL Settings

Name	Description	Required Value
FTP Host	Specifies the **FTP server host name or IP address.**	The FTP server host name or IP address. The configured default is **localhost**.
Explicit port for FTP over SSL	Specifies the **FTP port for explicit SSL. default is 21 (data port 20).**	The FTP port number for explicit SSL. The configured default is **21** (data port 20).
Implicit port for FTP over SSL	Specifies the **FTP port for implicit SSL.**	The **FTP port for implicit SSL** **The configured default is 990 (data port 989).**
User ID	Specifies the user login for FTP server.	The user login name for FTP server.
Password	Specifies the password for the FTP server user login.	The password for the FTP server user login.
Key Store Location	Specifies the path to the keystore that contains the trusted CA certificates required for server authentication.	The fully qualified path to the keystore file**.**
Key Store Password	Specifies the password to access the keystore file.	**The password for the keystore.**

Table 23 Environment - BatchFTPOverSSL - FTP and SSL Settings (Continued)

Name	Description	Required Value
Key Store Type	Specifies the keystore format type. Selections include **JKS** or **other**. ***Note:*** JKS is currently the only supported keystore type. If "other" is selected as the value, an exception is thrown when the OTD is initialized, with the error message, "Unknown type key store".	Select JKS. **JKS** is currently the only supported keystore type, and the configured default.

3.6.2 Firewall Settings (BatchFTPOverSSL Environment)

The **Firewall Settings** section of the BatchFTPOverSSL Environment properties contains the top-level parameters displayed in Table 24.

Table 24 Environment - BatchFTPOverSSL - Firewall Settings

Name	Description	Required Value
Firewall Host	Specifies **the proxy server host name or IP.**	The proxy server host name or IP address.
Firewall Port	Specifies **the proxy server port.**	The proxy server port number.
User ID	Specifies **the user login on the proxy server.**	User login ID.
Password	Specifies **the password for the user login.**	The user password.

3.6.3 General Settings (BatchFTPOverSSL Environment)

The **General Settings** section of the BatchFTPOverSSL Environment properties contains the top-level parameters displayed in Table 25.

Table 25 Environment - BatchFTPOverSSL - General Settings

Name	Description	Required Value
Connection Mode	Specifies whether a physical connection is established when an external connection is instantiated. Options are: ▪ **Automatic**: Establishes a physical connection when an external connection is instantiated. ▪ **Manual**: Does not automatically establish a physical connection when an external connection is instantiated. If a physical connection is not automatically established, a physical connection must be established from the Collaboration (for example, by calling the **connect()** method). If BatchFTPOverSSL Connection Mode is set to **Automatic**, the following Environment parameters must be set with valid values: ♦ FTP and SSL Settings > **FTP Host** ♦ FTP and SSL Settings > **Explicit port for FTP over SSL** (this must be set if the BatchFTPOverSSL Connectivity Map property, **FTP and SSL Settings > Secure Mode** is set to **Explicit SSL**) ♦ FTP and SSL Settings > **Implicit port for FTP over SSL** (this must be set if the BatchFTPOverSSL Connectivity Map property, **FTP and SSL Settings > Secure Mode** is set to **Implicit SSL**) ♦ FTP and SSL Settings > **User ID** ♦ FTP and SSL Settings > **Password** ♦ FTP and SSL Settings > **Key Store Location** (this must be set if the BatchFTPOverSSL Connectivity Map property, **FTP and SSL Settings > Require Server Authentication** is set to **Yes**, or if the BatchFTPOverSSL Connectivity Map property, **FTP and SSL Settings > Distinguished Name for User** contains a value, indicating that client authentication is required) ♦ FTP and SSL Settings > **Key Store Password** (this must be set if **Key Store Location** is set) Also, if a firewall is enabled and the Connectivity Map property, **Firewall Settings > Use Firewall**, is set to **Yes**, the following Environment parameters must be set with valid values: ♦ Firewall Settings > **Firewall Host** ♦ Firewall Settings > **Firewall Port** ♦ Firewall Settings > **User** ♦ Firewall Settings > **Password**	Select **Automatic** or **Manual**. The default is **Automatic**.

Table 25 Environment - BatchFTPOverSSL - General Settings

Name	Description	Required Value
Temp Dir	Specifies a working directory for storing intermediary results. Options include: ▪ **Leave value blank**: BatchFTPOverSSL will use a default location as the temporary directory. ▪ Specify a path to a local file system folder with read/write permissions. ***Note:*** See **"Deploying an EAR File" on page 24** if you are deploying the project to another application server.	A working directory with read/write permissions, or leave blank (no value) to accept a default directory.

3.6.4 Connection Pool Settings (BatchFTPOverSSL Environment)

The **Connection Pool Settings** section of the BatchFTPOverSSL Environment properties contains the top-level parameters displayed in Table 26.

Table 26 Environment - BatchFTPOverSSL - Connection Pool Settings

Name	Description	Required Value
Steady pool size	Specifies the minimum number of physical connections the pool keeps available at all times.	An integer indicating the maximum number of connections available at all times. A value of **0** (zero) indicates that there are no physical connections in the pool and that new connections are created as needed. The configured default is **2**.
Maximum pool size	**Specifies the maximum number of physical connections the pool contains. 0 (zero) indicates that there is no maximum.**	An integer indicating the maximum number of connections allowed. A value of **0 (zero) indicates that there is no maximum**. The configured default is **10**.
Max Idle Timeout In Seconds	**Specifies the maximum idle timeout (in seconds)**. This is a hint to the server. A timer thread periodically removes unused connections. This parameter defines the interval at which this thread runs. This thread removes unused connections after the specified idle time expires. It allows the user to specify the amount of time a connection can remain idle in the pool.	An integer indicating **the maximum idle timeout in seconds**. When this is set to a number greater than **0**, the container removes or destroys any connections that are idle for the specified duration. A value of **0** specifies that idle connections can remain in the pool indefinitely. The configured default is **300** (5 minutes).

3.6.5 Connection Retry Settings (BatchFTPOverSSL Environment)

The **Connection Retry Settings** section of the BatchFTPOverSSL Environment Map properties contains the top-level parameters displayed in Table 27.

Table 27 Environment - BatchFTPOverSSL - Connection Retry Settings

Name	Description	Required Value
Connection Retries	Specifies the number of retries to establish a connection upon failure to acquire a connection.	An integer indicating the maximum number of retries to establish a connection upon failure to acquire a connection. The default value is **0**.
Connection Retry Interval	**Specifies the length of the pause (in milliseconds) between each reattempt to access the destination file. Used in conjunction with the Connection Retries setting.**	An integer indicating **length of the pause (in milliseconds)**. The default value is **1000** (1 second).

3.7 BatchSCP eWay Connectivity Map Properties

This section describes the configuration properties for the **BatchSCP OTD,** accessed from the Connectivity Map.

The BatchSCP eWay Connectivity Map properties include the following sections:

3.7.1 SCP Settings (BatchSCP Connectivity Map)

The **SCP Settings** section of the BatchSCP Connectivity Map properties contains the top-level parameters displayed in Table 28.

Table 28 Connectivity Map - BatchSCP - SCP Settings

Name	Description	Required Value
Authentication Type	Specifies the client authentication type. The options are: ▪ PASSWORD ▪ HOSTBASED ▪ PUBLICKEY Refer to your specific SSH server documentation for information regarding your authentication type.	Select **PASSWORD**, **HOST BASED**, or **PUBLICKEY**. The configured default is **PASSWORD**.
Do Host Key Verification	Specifies whether SSH server authentication by verification of the public key, is enabled.	Select **Yes** or **No. Yes** enables SSH server authentication by verifying the public key. The configured default is **Yes**.
Remote Directory	Specifies the directory on the SSH (with SFTP sub-system) server where data is sent or received. The accessibility of the directory is usually dependent upon the login user..	The remote directory.
Remote File	Specifies the name of a file on the remote server used to either receive published data, or hold data to be retrieved.	The remote file.
Local Directory	Specifies the local directory for files be sent to the remote server, or received from remote server.	A local directory.
Local File	Specifies the local file under local directory to be sent to remote, or receive data from remote.	The local file.

Table 28 Connectivity Map - BatchSCP - SCP Settings

Name	Description	Required Value
Is Copy Recursive	Specifies whether the copy is recursive (for example, copy all sub directories).	Select **Yes** or **No. Yes** indicates that the copy is recursive. The configured default is **No**.

3.7.2 Firewall Settings (BatchSCP Connectivity Map)

The **Firewall Settings** section of the BatchSCP Connectivity Map properties contains the top-level parameters displayed in Table 29.

Table 29 Connectivity Map - BatchSCP - Firewall Settings

Name	Description	Required Value
Use Firewall	Specifies whether a firewall is used.	**Yes** or **No**. **Yes** indicates that you are using a firewall. The configured default is **No**.
SOCKS Version	Specifies the SOCKS version required by the firewall. The supported options are **4** for SOCKS version **4**, or **5** for SOCKS version **5**	Select **4** for SOCKS version 4, or **5** for SOCKS version **5**. The configured default is **5**.

3.7.3 Synchronization (BatchSCP Connectivity Map)

The **Synchronization** section of the BatchSCP Connectivity Map properties contains the top-level parameters displayed in Table 30.

Table 30 Connectivity Map - BatchSCP - Synchronization

Name	Description	Required Value
Synchronized	Specifies whether the eWay simulates the pre-version 5.1 eWay behavior in which the eWay runs synchronized or in parallel. The selections are: ▪ **Yes**: The eWay run in synchronized mode, one instance of the Collaboration after the other. ▪ **No**: The eWay run in parallel, creating multiple instances of the Collaboration that run in parallel. ***Note:*** All OTD instances used in a Project should have the same value for this property.	**Yes** or **No.** The default setting is **No**.

3.8 BatchSCP eWay Environment Properties

This section describes the configuration properties for the **BatchSCP OTD,** accessed from the Environment Explorer.

The BatchSCP eWay Environment properties include the following sections:

3.8.1 SSH Settings (BatchSCP Environment)

The **SSH Settings** section of the BatchSCP Environment properties contains the top-level parameters displayed in Table 31.

Table 31 Environment - BatchSCP - SSH Settings

Name	Description	Required Value
SSH Host	Specifies the host name or IP address of the SSH server.	The host name or IP address of the SSH server.
SSH Port	Specifies the port number of the SSH server.	The port number of the SSH server.
User	Specifies the user login name for the SSH server.	A user login name for the SSH server.
Password	Specifies a login password for the user.	A login password.
Key File	Specifies the folder (path) that holds the private key for client authentication.	The folder (path) that holds the private key for client authentication.
Key File Password	Specifies the password used to protect the key file (key pair).	The Key File password.
Server Public Key	Specifies the folder (path) that holds the public key used by the SSH server. This key is generated on the server and sent to the client through a, safe channel, and stored on the client machine for host key verification. See your specific SSH server documentation for more information.	The the folder (path) that contains the server public key for host key verification.
Server Name For Host Key Verification	Specifies the full domain name of the SSH server used for host key verification.	The expected SSH host name when doing host key verification.

Table 31 Environment - BatchSCP - SSH Settings (Continued)

Name	Description	Required Value
Preferred Public Key Algorithm	Specifies the preferred public key algorithm for SSH authentication. The options are **DSA** and **RSA**.	**DSA** or **RSA**. The configured default is DSA.

3.8.2 Firewall Settings (BatchSCP Environment)

The **Firewall Settings** section of the BatchSCP Environment properties contains the top-level parameters displayed in Table 11.

Table 32 Environment - BatchSCP - Firewall Settings

Name	Description	Required Value
Firewall Host	Specifies **the proxy server host name or IP address.**	The proxy server host name or IP address. The configured default is **localhost**.
Firewall Port	Specifies **the proxy server port number.**	The proxy server port number. The configured default is **1080**.
Password	Specifies **a login password for the proxy server.**	A password for the proxy server login ID.
User	Specifies **a user login ID for the proxy server.**	A **user login ID for the proxy server.**

3.8.3 General Settings (BatchSCP Environment)

The **General Settings** section of the BatchSCP Environment properties contains the top-level parameters displayed in Table 33.

Table 33 Environment - BatchSCP - General Settings

Name	Description	Required Value
Connection Mode	Specifies whether a physical connection is established when an external connection is instantiated. Options are: ▪ **Automatic**: Establishes a physical connection when an external connection is instantiated. ▪ **Manual:** Does not automatically establish a physical connection when an external connection is instantiated. If a physical connection is not automatically established, a physical connection must be established from the Collaboration (for example, by calling the **connect()** method). If BatchSCP Connection Mode is set to **Automatic**, the following Environment parameters must be set with valid values: ♦ SCP Settings > **SSH Host** ♦ SCP Settings > **SSH Port** ♦ SCP Settings > **User** ♦ SCP Settings > **Password** ♦ SCP Settings > **Key File** (this must be set if the BatchSCP Connectivity Map property, **SCP Settings > Authentication Type** is set to **HOSTBASED** or **PUBLICKEY**) ♦ SCP Settings > **Key File Password** (required by the **Key File** property) ♦ SCP Settings > **Server Public Key** (this must be set if the BatchSCP Connectivity Map property, **SCP Settings > Do Host Key Verification** is set to **Yes**) ♦ SCP Settings > **Server Name for Host Key Verification** (this must be set if the BatchSCP Connectivity Map property, **SCP Settings > Do Host Key Verification** is set to **Yes**) Also, if a firewall is enabled and the Connectivity Map property, **Firewall Settings > Use Firewall**, is set to **Yes**, the following Environment parameters must be set with valid values: ♦ Firewall Settings > **Firewall Host** ♦ Firewall Settings > **Firewall Port** ♦ Firewall Settings > **User** ♦ Firewall Settings > **Password**	Select **Automatic** or **Manual**. The configured default is **Automatic**.

3.8.4 Connection Pool Settings (BatchSCP Environment)

The **Connection Pool Settings** section of the BatchSCP Environment properties contains the top-level parameters displayed in Table 34.

Table 34 Environment - BatchSCP - Connection Pool Settings

Name	Description	Required Value
Steady pool size	Specifies the minimum number of physical connections the pool keeps available at all times.	An integer indicating the maximum number of connections available at all times. A value of **0** (zero) indicates that there are no physical connections in the pool and that new connections are created as needed. The configured default is **2**.
Maximum pool size	Specifies the maximum number of physical connections the pool contains. 0 (zero) indicates that there is no maximum.	An integer indicating the maximum number of connections allowed. A value of **0 (zero) indicates that there is no maximum**. The configured default is **10**.
Max Idle Timeout In Seconds	Specifies the maximum idle timeout (in seconds). This is a hint to the server. A timer thread periodically removes unused connections. This parameter defines the interval at which this thread runs. This thread removes unused connections after the specified idle time expires. It allows the user to specify the amount of time a connection can remain idle in the pool.	An integer indicating **the maximum idle timeout in seconds**. When this is set to a number greater than **0**, the container removes or destroys any connections that are idle for the specified duration. A value of **0** specifies that idle connections can remain in the pool indefinitely. The configured default is **300** (5 minutes).

3.8.5 Connection Retry Settings (BatchSCP Environment)

The **Connection Retry Settings** section contains the following top level parameters:

The **General Settings** section of the BatchSCP Environment Map properties contains the top-level parameters displayed in Table 35.

Table 35 Environment - BatchSCP - Connection Retry Settings

Name	Description	Required Value
Connection Retries	Specifies the number of retries to establish a connection upon failure to acquire a connection.	An integer indicating the maximum number of retries to establish a connection upon failure to acquire a connection. The default value is **0**.
Connection Retry Interval	Specifies the length of the pause (in milliseconds) between each reattempt to access the destination file. Used in conjunction with the Connection Retries setting.	An integer indicating **length of the pause (in milliseconds)**. The default value is **1000** (1 second).

3.9 BatchSFTP eWay Connectivity Map Properties

This section describes the configuration properties for the **BatchSFTP OTD,** accessed from the Connectivity Map.

The BatchSFTP eWay Connectivity Map properties include the following sections:

3.9.1 Pre Transfer (BatchSFTP Connectivity Map)

The **Pre Transfer** section allows user to customize the behaviors of protection/backup/recovery. This section describes the operation that will be performed on remote end or locally before the real file transfer.

The **Pre Transfer** section of the BatchSFTP Connectivity Map properties contains the top-level parameters displayed in Table 18.

Table 36 Connectivity Map - BatchSFTP - Pre Transfer

Name	Description	Required Value
Remote Dir Name	Specifies the directory name (path) on the remote external system where the file the is renamed or copied. This is only for **Rename** or **Copy** of the Remote Pre Command. The value can be a literal, regular expression (source), or pattern name (destination). For outbound (destination), the directory is created if it doesn't already exist. Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. For example, the pattern **%f** means the original working directory name. See **"Remote Dir Name Is Pattern" on page 94**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A directory name.

Table 36 Connectivity Map - BatchSFTP - Pre Transfer (Continued)

Name	Description	Required Value
Remote Dir Name Is Pattern	Specifies whether the Remote Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Remote Dir Name" on page 93**.	Select **Yes** or **No**. The configured default is **Yes**.
Remote File Name	Specifies the file name on the external system, to which a file is renamed or copied. This setting is only for the **Rename** or **Copy** operations of **Pre Transfer Command** parameter. The value can be a literal, regular expression (get), or pattern name (put). Special characters are allowed, for example, the pattern **%f** indicates the original working file name. The expansion of any special characters is carried out each time this parameter is used. See **"Remote File Name Is Pattern" on page 94**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The file name.
Remote File Name Is Pattern	Specifies whether the Remote File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Remote File Name" on page 94**.	Select **Yes** or **No**. The configured default is **Yes**.

Table 36 Connectivity Map - BatchSFTP - Pre Transfer (Continued)

Name	Description	Required Value
Remote Pre Command	Allows you to execute a desired action directly before the actual file transfer. For an inbound transfer, the file can be made unavailable to other clients polling the target system with the same directory and file pattern or name. For an outbound transfer, you can perform an automatic backup/clean-up of the existing files. The options are: ▪ **Rename**: Rename the target file for protection or recovery. ▪ **Copy**: Copy the target file for backup or recovery. ▪ **None**: Do nothing. To gain proper protection, backup, or recovery, you must choose the appropriate setting that serves your purpose. For example, to recover from failures on an outbound appending transfer, use the **Copy** setting. ***Caution:*** When you are using **Rename**, if the destination file exists, different FTP servers can behave differently. For example, on some UNIX FTP servers, the destination file is overwritten without question. That is, no error or warning message is given. On other FTP servers, a Windows XP server for example, the system generates an error that results in exceptions being thrown in the called OTD method. Be sure you are familiar with the native behavior of the corresponding FTP server. If you are in doubt, try the action at the command prompt. If the action displays an error message, it may result in an exception being thrown in the Collaboration.	Select **Rename**, **Copy**, or **None**. The configured default is **None**. ***Note:*** The **Copy** option could slow system performanc e, especially if you are copying a large file.

Table 36 Connectivity Map - BatchSFTP - Pre Transfer (Continued)

Name	Description	Required Value
Local Dir Name	Specifies the local directory name (path) to be used by **Rename** or **Copy**. The value can be a literal, regular expression (source), or pattern name (destination). Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. See **"Local Dir Name Is Pattern" on page 96**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**. ***Note:*** When entering a path separator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir.	A directory name.
Local Dir Name Is Pattern	Specifies whether the Local Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. See **"Local Dir Name" on page 96**.	Select **Yes** or **No**. The configured default is **No**.
Local File Name	Specifies the local file name to be used by **Rename** or **Copy**. The value can be a literal, regular expression (get), or pattern name (put). Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. See **"Local File Name Is Pattern" on page 96**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A file name.
Local File Name Is Pattern	Specifies whether the Local File Name name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local File Name" on page 96**.	Select **Yes** or **No**. The configured default is **No**.

Table 36 Connectivity Map - BatchSFTP - Pre Transfer (Continued)

Name	Description	Required Value
Local Pre Command	Allows you to execute a desired action directly before the actual file transfer. For an inbound transfer, the file can be made unavailable to other clients polling the target system with the same directory and file pattern or name. For an outbound transfer, you can perform an automatic backup of the existing files. The options are: ▪ **Rename**: Rename the target file for protection or recovery. ▪ **Copy**: Copy the target file for backup or recovery. ▪ **None**: Do nothing. To gain proper protection, backup, or recovery, you must choose the appropriate setting that serves your purpose. For example, to recover from failures on an outbound appending transfer, use the **Copy** setting. ***Caution:*** Rename and Copy overwrite the file specified by the Local Dir Name and Local File Name properties, if they exist. See **"Using Name Patterns" on page 178**.	Select **Rename**, **Copy**, or **None**. The configured default is **None**. ***Note:*** The **Copy** option could slow system performanc e, especially if you are copying a large file.

3.9.2 SFTP Settings (BatchSFTP Connectivity Map)

The **SFTP Settings** section of the BatchSFTP Connectivity Map properties contains the top-level parameters displayed in Table 37.

Table 37 Connectivity Map - BatchSFTP - SFTP Settings

Name	Description	Required Value
Transfer Mode	Specifies whether the transfer is binary code or ASCII text.	**Select BINARY or ASCII.** The configured default is **BINARY**.
Remote EOL	Specifies the remote server - end of line. Options are **CR**, **LF**, **CRLF**.	Select **CR**, **LF**, or **CRLF**. **CRLF** is the configured default.
Transfer Block Size	Specifies the block size used when transferring files. Do not increase the default, as the remote server may not be able to support higher blocksizes.	An integer indicating the block size used when transferring files. The configured default is **32768**.
Local Read Buffer Size	Specifies the size (in bytes) of the buffer which is used to read from the local file system.	An integer indicating the size (in bytes) of the local read buffer. A value of **-1** indicates that the whole local file is read at once.
Authentication Type	Specifies the client authentication type. The options are as follows: ▪ PASSWORD ▪ HOSTBASED ▪ PUBLICKEY Refer to your specific SSH server documentation for information regarding your authentication type.	Select **PASSWORD**, **HOST BASED**, or **PUBLICKEY**. The configured default is **PASSWORD**.
Do Host Key Verification	Specifies whether SSH server authentication by verification of the public key, is enabled.	Select **Yes** or **No. Yes** enables SSH server authentication by verifying the public key. The configured default is **Yes**.

Table 37 Connectivity Map - BatchSFTP - SFTP Settings (Continued)

Name	Description	Required Value
Remote Directory	Specifies the directoryname (path) on the SSH (with SFTP sub-system) server where data is sent or received. The accessibility of the directory usually depends on the login user. The value can be a literal, regular expression (source), or pattern name (destination). See **"Remote Directory Name is Pattern" on page 99**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The remote directory name.
Remote Directory Name is Pattern	Specifies whether the Remote Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Remote Directory" on page 99**.	**Select Yes** or **No.** The configured default is **No**.
Remote File	Specifies the name of a file on the remote server. The value can be a literal, regular expression (get), or pattern name (put). See **"Remote File Name is Pattern" on page 99**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The remote file.
Remote File Name is Pattern	Specifies whether the Remote File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Remote File" on page 99**.	**Select Yes** or **No.** The configured default is **No.**

Table 37 Connectivity Map - BatchSFTP - SFTP Settings (Continued)

Name	Description	Required Value
Local Directory	Specifies the local directory name (path) for sending or recieve files on the remote server. The value can be a literal, regular expression (source), or pattern name (destination). See **"Local Directory Name is Pattern" on page 100**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A local directory.
Local Directory Name is Pattern	Specifies the meaning of the **Local Directory Name** property as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local Directory" on page 100**.	**Select Yes** or **No.** The configured default is **No**.
Local File	Specifies the local file to be sent or received on the remote server. The value can be a literal, regular expression (get), or pattern name (put). See **"Local File Name is Pattern" on page 100**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The local file.
Local File Name is Pattern	Specifies whether the local file name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local File" on page 100**.	**Select Yes** or **No.** The configured default is **No**.

3.9.3 Post Transfer (BatchSFTP Connectivity Map)

The **Post Transfer** section of the BatchSFTP Connectivity Map properties contains the top-level parameters displayed in Table 38.

Table 38 Connectivity Map - BatchSFTP - Post Transfer

Name	Description	Required Value
Remote Dir Name	Specifies the directory name (path) on the remote system where the file the will be renamed or copied. T This is only for **Rename** or **Copy** of the Post Transfer Command. The value can be a literal, regular expression (source), or pattern name (destination). For outbound (destination), the directory is created if it doesn't already exist. Special characters are allowed. For example, the pattern **%f** means the original working directory name. The expansion of any special characters is carried out each time this parameter is used. See **"Remote Dir Name Is Pattern" on page 101**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A directory name.
Remote Dir Name Is Pattern	Specifies whether the Remote Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Remote Dir Name Is Pattern" on page 101**.	Select **Yes** or **No**. The configured default is **Yes**.
Remote File Name	Specifies the file name on the external system. This setting is only for the **Rename** or **Copy** operations of **Post Transfer Command** parameter. The value can be a literal, regular expression (get), or pattern name (put). Special characters are allowed, for example, the pattern **%f** indicates the original working file name. The expansion of any special characters is carried out each time this parameter is used. See **"Remote File Name Is Pattern" on page 102**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A file name.

Table 38 Connectivity Map - BatchSFTP - Post Transfer (Continued)

Name	Description	Required Value
Remote File Name Is Pattern	Specifies whether the Remote File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Remote File Name" on page 101**.	Select **Yes** or **No**. The configured default is **Yes**.
Remote Post Command	Allows you to execute a desired action directly after the actual file transfer. For an inbound transfer, it can be applied to mark the transferred file as consumed by making an automatic backup (**Rename**) or by destroying it permanently (**Delete**). For an outbound transfer, it can be applied to make the transferred file available to other clients by renaming it. ▪ **Rename**: Rename the transferred file. ▪ **Delete**: Delete the transferred file. ▪ **None**: Do nothing. ***Note:*** For **Rename**, if the destination file exists, different FTP servers may behave differently. For example, on some UNIX FTP servers, the destination file will be overwritten without extra message. On an NT FTP server, this will fail and get an exception. It does not define unified behavior, rather, it follows the native behavior of the corresponding FTP server.	Select **Rename**, **Delete**, or **None**. The configured default is **None**.
Local Dir Name	Specifies the local directory name (path) to be used by **Rename**. The value can be a literal, regular expression (source), or pattern name (destination). Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. See **"Local Dir Name Is Pattern" on page 103**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**. When entering a path separator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir.	A directory name.

Table 38 Connectivity Map - BatchSFTP - Post Transfer (Continued)

Name	Description	Required Value
Local Dir Name Is Pattern	Specifies whether the Local Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local Dir Name" on page 102**.	Select **Yes** or **No**.The configured default is **No**.
Local File Name	Specifies the local file name to be used by **Rename**. The value can be a literal, regular expression (get), or pattern name (put). Special characters are allowed. The expansion of any special characters is carried out each time this parameter is used. See **"Local File Name Is Pattern" on page 103**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	A file name.
Local File Name Is Pattern	Specifies whether the Local File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Local File Name" on page 103**.	Select **Yes** or **No**. The configured default is **No**.

Table 38 Connectivity Map - BatchSFTP - Post Transfer (Continued)

Name	Description	Required Value
Local Post Command	Allows you to execute a desired action directly after the actual file transfer. For an inbound transfer, the target file can be marked as consumed by making an automatic backup (**Rename**) or by destroying it permanently (**Delete**). For an outbound transfer the target file can be made available to other clients by renaming it. The options are: ▪ **Rename**: Rename the target file. ▪ **Delete**: Delete the target file (inbound transfers only). ▪ **None**: Do nothing. ***Note:*** Rename overwrites the file specified by the Local Dir Name and Local File Name properties, if they exist.	Select **Rename**, **Delete**, or **None**. The configured default is **None**.

3.9.4 Firewall Settings (BatchSFTP Connectivity Map)

The **Firewall Settings** section of the BatchSFTP Connectivity Map properties contains the top-level parameters displayed in Table 39.

Table 39 Connectivity Map - BatchSFTP - Firewall Settings

Name	Description	Required Value
Use Firewall	Specifies whether a firewall is used.	Select **Yes** or **No**. **Yes** indicates that you are using a firewall. The configured default is **No**.
SOCKS Version	Specifies the SOCKS version of the firewall. The supported options are **4** for SOCKS version **4**, or **5** for SOCKS version **5**	Select **4** for SOCKS version 4, or **5** for SOCKS version **5**. The configured default is **5**

3.9.5 Synchronization (BatchSFTP Connectivity Map)

The **Synchronization** section of the BatchSFTP Connectivity Map properties contains the top-level parameters displayed in Table 40.

Table 40 Connectivity Map - BatchSFTP - Synchronization

Name	Description	Required Value
Synchronized	Specifies whether the eWay simulates the pre-version 5.1 eWay behavior in which the eWay runs synchronized or in parallel. The selections are: ▪ **Yes**: The eWay run in synchronized mode, one instance of the Collaboration after the other. ▪ **No**: The eWay run in parallel, creating multiple instances of the Collaboration that run in parallel. ***Note:*** All OTD instances used in a Project should have the same value for this property.	**Select Yes** or **No.** The default setting is **No**.

3.10 BatchSFTP eWay Environment Properties

This section describes the configuration properties for the **BatchSFTP OTD,** accessed from the Environment Explorer.

The BatchSFTP eWay Environment properties include the following sections:

3.10.1 SFTP Settings (BatchSFTP Environment)

The **SFTP Settings** section of the BatchSFTP Environment properties contains the top-level parameters displayed in Table 41.

Table 41 Environment - BatchSFTP - SFTP Settings

Name	Description	Required Value
SSH Host	Specifies the host name or IP address of the SSH server.	The host name or IP address of the SSH server.
SSH Port	Specifies the port number of the SSH server.	The port number of the SSH server.
User ID	Specifies the user login name for the SSH server.	A user login name for the SSH server.
Password	Specifies a login password for the user.	A password.
Key File	Specifies the folder (path) that holds the private key for client authentication of type PUBLIC KEY or HOST BASED..	The folder (path) that holds the private key for client authentication.
Key File Password	Specifies the password used to protect the key file.	The Key File password.
Server Public Key	Specifies the folder (path) that holds the public key used by the SSH server. This key is generated on the server and sent to the client through a, safe channel, and stored on the client machine for host key verification.	The the folder (path) that contains the server public key for host key verification.
Server Name For Host Key Verification	Specifies the full domain name of the SSH server used for host key verification.	The expected SSH host name when doing host key verification.
Preferred Public Key Algorithm	Specifies the preferred public key algorithm for SSH authentication. The options are **DSA** and **RSA**	**DSA** or **RSA**. The configured default is DSA.

3.10.2 Firewall Settings (BatchSFTP Environment)

The **Firewall Settings** section of the BatchSFTP Environment properties contains the top-level parameters displayed in Table 42.

Table 42 Environment - BatchSFTP - Firewall Settings

Name	Description	Required Value
Firewall Host	Specifies **the proxy server host name or IP address.**	The proxy server host name or IP address. The configured default is **localhost**.
Firewall Port	Specifies **the proxy server port number.**	The proxy server port number. The configured default is **1080**.
User ID	Specifies **a user login ID for the proxy server.**	A **user login ID for the proxy server.**
Password	Specifies **a login password for the proxy server.**	A password for the proxy server login ID.

3.10.3 General Settings (BatchSFTP Environment)

The **General Settings** section of the BatchSFTP Environment properties contains the top-level parameters displayed in Table 43.

Table 43 Environment - BatchSFTP - General Settings

Name	Description	Required Value
Connection Mode	Specifies whether a physical connection is established when an external connection is instantiated. Options: ▪ **Automatic:** Establishes a physical connection when an external connection is instantiated. ▪ **Manual:** Does not automatically establish a physical connection when an external connection is instantiated. If a physical connection is not automatically established, a physical connection must be established from the Collaboration (for example, by calling the **connect()** method). If BatchSFTP Connection Mode is set to **Automatic**, the following Environment parameters must be set with valid values: ♦ SFTP Settings > **SSH Host** ♦ SFTP Settings > **SSH Port** ♦ SFTP Settings > **User** ♦ SFTP Settings > **Password** ♦ SFTP Settings > **Key File** (this must be set if the BatchSFTP Connectivity Map property, **SFTP Settings > Authentication Type**, is set to **HOSTBASED** or **PUBLICKEY**) ♦ SFTP Settings > **Key File Password** (required by the **Key File** property) ♦ SFTP Settings > **Server Public Key** (this must be set if the BatchSFTP Connectivity Map property, **SFTP Settings > Do Host Key Verification** is set to **Yes**) ♦ SFTP Settings > **Server Name for Host Key Verification** (this must be set if the BatchSFTP Connectivity Map property, **SFTP Settings > Do Host Key Verification** is set to **Yes**) Also, if a firewall is enabled and the Connectivity Map property, **Firewall Settings > Use Firewall**, is set to **Yes**, the following Environment parameters must be set with valid values: ♦ Firewall Settings > **Firewall Host** ♦ Firewall Settings > **Firewall Port** ♦ Firewall Settings > **User** ♦ Firewall Settings > **Password**	Select **Automatic** or **Manual**. The configured default is **Automatic**.

3.10.4 Connection Pool Settings (BatchSFTP Environment)

The **Connection Pool Settings** section of the BatchSFTP Environment properties contains the top-level parameters displayed in Table 34.

Table 44 Environment - BatchSFTP - Connection Pool Settings

Name	Description	Required Value
Steady pool size	Specifies the minimum number of physical connections the pool keeps available at all times.	An integer indicating the maximum number of connections available at all times. A value of **0** (zero) indicates that there are no physical connections in the pool and that new connections are created as needed. The configured default is **2**.
Maximum pool size	**Specifies the maximum number of physical connections the pool contains. 0 (zero) indicates that there is no maximum.**	An integer indicating the maximum number of connections allowed. A value of **0 (zero) indicates that there is no maximum**. The configured default is **10**.
Max Idle Timeout In Seconds	**Specifies the maximum idle timeout (in seconds)**. This is a hint to the server. A timer thread periodically removes unused connections. This parameter defines the interval at which this thread runs. This thread removes unused connections after the specified idle time expires. It allows the user to specify the amount of time a connection can remain idle in the pool.	An integer indicating **the maximum idle timeout in seconds**. When this is set to a number greater than **0**, the container removes or destroys any connections that are idle for the specified duration. A value of **0** specifies that idle connections can remain in the pool indefinitely. The configured default is **300** (5 minutes).

3.10.5 Connection Retry Settings (BatchSFTP Environment)

The **Connection Retry Settings** section contains the following top level parameters:

The **General Settings** section of the BatchSFTP Environment Map properties contains the top-level parameters displayed in Table 45.

Table 45 Environment - BatchSFTP - Connection Retry Settings

Name	Description	Required Value
Connection Retries	Specifies the number of retries to establish a connection upon failure to acquire a connection.	An integer indicating the maximum number of retries to establish a connection upon failure to acquire a connection. The default value is **0**.
Connection Retry Interval	**Specifies the length of the pause (in milliseconds) between each reattempt to access the destination file. Used in conjunction with the Connection Retries setting.**	An integer indicating **length of the pause (in milliseconds)**. The default value is **1000** (1 second).

3.11 BatchLocalFile Connectivity Map Properties

This section explains the properties for the **BatchLocalFile OTD, accessed from the Connectivity Map**.

The BatchLocalFile properties include the following sections:

Caution: *Several of these configuration options allow regular expressions to be used. This advanced feature is useful but must be used carefully. An improperly formed regular expression can cause undesired data or loss of data. You must have a clear understanding of regular-expression syntax and construction before attempting to use this feature. It is recommended that you test such configurations thoroughly before moving them to production.*

3.11.1 Pre Transfer (BatchLocalFile Connectivity Map)

This section provides information about configuring the **Pre Transfer** parameters. Pre-transfer operations are those operations executed right before the actual data transfer.

The **Pre Transfer** section of the BatchLocalFile Connectivity Map properties contains the top-level parameters displayed in Table 46.

Table 46 Connectivity Map - BatchLocalFile - Pre Transfer

Name	Description	Required Value
Pre Directory Name	Specifies the directory name (path) on the external system in which a file is renamed or copied. This setting is only for the **Rename** or **Copy** operations of **Pre Transfer Command** parameter. The value can be a literal, or pattern name. For outbound transfers, the directory is created if it does not already exist. Special characters are allowed. For example, the pattern **%f** indicates the original working directory name. The expansion of any special characters is carried out each time this parameter is used. See ***Note:*** For path separator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir. See **"Pre Directory Name Is Pattern" on page 113**. See **"Using Name Patterns" on page 178**.	Enter the exact name of the directory (with the path), enter a pattern name, or select one of the following values: ▪ %f ▪ %f.%y%y%y%y%M%M %d%d.%h%h%m%m %s%s%S%S%S ▪ %f.copy ▪ %f.rename
Pre Directory Name Is Pattern	Specifies whether the Pre Directory Name represents a literal or a name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Pre Directory Name Is Pattern" on page 113**.	Select **Yes** or **No.** The configured default is **No**.

Table 46 Connectivity Map - BatchLocalFile - Pre Transfer (Continued)

Name	Description	Required Value
Pre File Name	Specifies the file name on the external system, to which a file is renamed or copied. This setting is only for the **Rename** or **Copy** operations of **Pre Transfer Command** parameter. The value can be a literal or pattern name. Special characters are allowed, for example, the pattern **%f** indicates the original working file name. The expansion of any special characters is carried out each time this parameter is used. See **"Pre File Name Is Pattern" on page 114**. See **"Using Name Patterns" on page 178**	Enter the exact name of the file, enter a pattern name, or select one of the following values: ▪ %f ▪ %f%# ▪ %f.%y%y%y%y%M%M %d%d.%h%h%m%m %s%s%S%S%S ▪ %f.copy ▪ %f.rename
Pre File Name Is Pattern	Specifies whether the Pre File Name represents a literal or a name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Pre File Name" on page 114**.	Select **Yes** or **No.** The configured default is **No**.
Pre Transfer Command	Allows you to determine the action executed directly before the actual file transfer. In the case of an inbound file transfer, you can make the file unavailable to other clients polling the target system via the same directory and file pattern or name. In the case of an outbound transfer, you can make an automatic backup of the existing file. The options are as follows: ▪ **Rename**: Rename the target file. ▪ **Copy**: Copy the target file. ▪ **None**: Do nothing.	Select **Rename**, **Copy**, or **None**; the default is **None**. ***Note:*** **Rename** and **Copy** overwrite the file or directory specified by the **Pre Directory Name** and **Pre Transfer Name** parameter, if it exists.

Note: *For more information on this feature, see* **"Pre/Post File Transfer Commands" on page 165**.

3.11.2 Sequence Numbering (BatchLocalFile Connectivity Map)

The **Sequence Numbering** section of the BatchLocalFile Connectivity Map properties contains the top-level parameters displayed in Table 47.

Table 47 Connectivity Map - BatchLocalFile - Sequence Numbering

Name	Description	Required Value
Max Sequence Number	Use this parameter when you have set up the target file name to contain a sequence number. It tells the eWay that when this value (the **Max Sequence Number**) is reached, to reset the sequence number to the **Starting Sequence Number** value. This parameter is used for the name pattern %#. See **"Using Name Patterns" on page 178**.	An integer from **1** to **2147483647**. The value of **Max Sequence Number** *must* be greater than that of **Starting Sequence Number**. The configured default value is **999999**.
Starting Sequence Number	Use this parameter when you have set up the target file name to contain a sequence number. It tells the eWay which value to start with in the absence of a sequence number from a previous run. Also, when the **Max Sequence Number** value is reached, the sequence number rolls over to the **Starting Sequence Number** value. This parameter is used for the name pattern **%#**. See **"Using Name Patterns" on page 178**.	An integer from **0** to **2147483647**. The value of the **Starting Sequence Number** *must* be less than the **Max Sequence Number**. The configured default value is **1**.

Note: *The Synchronized property, under General Settings, must be set to "Yes" to use Sequence Numbering.*

3.11.3 Post Transfer (BatchLocalFile Connectivity Map)

Post-transfer operations are those performed after the data transfer.

The **Post Transfer** section of the BatchLocalFile Connectivity Map properties contains the top-level parameters displayed in Table 48.

Table 48 Connectivity Map - BatchLocalFile - Post Transfer

Name	Description	Required Value
Post Directory Name	Specifies the directory name (path) on the external system in which a file is renamed. This setting is only for the **Rename** operation of the **Post Transfer Command** parameter. The value can be a literal or pattern name. For outbound transfers, the directory is created if it does not already exist. Special characters are allowed, for example, the pattern **%f** indicates the original working directory name. The expansion of any special characters is carried out each time this parameter is used. ***Note:*** For path separator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir. See **"Post Directory Name Is Pattern" on page 116**. See **"Using Name Patterns" on page 178**.	Enter the exact name of the directory (with the path), enter a pattern name, or select one of the following values: ▪ %f ▪ %f%# ▪ %f.%y%y%y%y%M%M %d%d.%h%h%m%m %s%s%S%S%S ▪ %f.copy ▪ %f.rename
Post Directory Name Is Pattern	Specifies whether the Post Directory Name represents a literal or a name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Post Directory Name" on page 116**.	Select **Yes** or **No.** The configured default is **No**.

Table 48 Connectivity Map - BatchLocalFile - Post Transfer

Name	Description	Required Value
Post File Name	Specifies either the name of the file that the transferred file is renamed to (Rename) or the directory it is moved to (Move), depending on the setting in the parameter **Post Transfer Command**. The value can be a literal or pattern name. Special characters are allowed. The expansion of any special characters are carried out each time this parameter is used. See **"Post File Name Is Pattern" on page 117**. See **"Using Name Patterns" on page 178**.	Enter the exact name of the file, enter a pattern name, or select one of the following values:Select one of the following values: ▪ %f ▪ %f.%y%y%y%y%M%M%d%d.%h%h%m%m%s%s%S%S%S ▪ %f.copy ▪ %f.rename
Post File Name Is Pattern	SSpecifies whether the Post File Name represents a literal or a name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Post File Name" on page 117**.	Select **Yes** or **No.** The configured default is **No**.
Post Transfer Command	Allows you to execute a desired action directly after the actual file transfer. For an inbound transfer, you can mark the transferred file as "consumed" by making an automatic backup (**Rename**) or by destroying it permanently (**Delete**). For an outbound transfer, you can make the transferred file available to other clients by renaming it. The options are as follows: ▪ **Rename**: Rename the target file. **Rename** overwrites the file specified by the **Post File Name** and **Post Directory Name**, if it exists ▪ **Copy**: Copy the target file. ▪ **Delete**: Delete the target file (inbound transfers only). ▪ **None**: Do nothing.	Select **Rename**, **Copy**, **Delete**, or **None**. The configured default is **None**.

3.11.4 General Settings (BatchLocalFile Connectivity Map)

The **General Settings** section of the BatchLocalFile Connectivity Map properties contains the top-level parameters displayed in Table 11.

Table 49 Connectivity Map - BatchLocalFile - General Settings

Name	Description	Required Value
Resume Reading Enabled	Specifies whether the OTD handles the Resume Reading feature as follows: ▪ **Yes**: Enables the OTD to store any state information necessary to resume reading from the current file in a subsequent execution of the Collaboration Rule. ▪ **No**: Indicates that the file is considered "consumed" even if the streaming consumer did *not* read until the end of file.	**Yes** or **No** The configured default is **No**. ***Note:*** Synchronized must be set to "Yes" to use Resume Reading.
Synchronized	Specifically applies to legacy Batch eWay Projects. Provides backward compatibility to allow Projects that were created using the Batch eWay version 5.0.7 or earlier to be imported and deployed without a change in the eWays behavior. The selections are: ▪ **Yes**: Provides backward compatibility for legacy (pre-5.0.8 Batch eWay) Projects. The eWay run in synchronized mode, one instance of the Collaboration after the other. ▪ **No**: For use with new Batch eWay Projects. The eWay run in parallel, creating multiple instances of the Collaboration that run in parallel. ***Note:*** All OTD instances used in a Project should have the same value for this property.	**Yes** or **No.** The default setting is **Yes, simulating Projects created with** Batch eWay version 5.0.7 or earlier. ***Note:*** Synchronized must be set to "Yes" to use Sequence Numbering or Resume Reading.

3.11.5 Target Location (BatchLocalFile Connectivity Map)

The **Target Location** section of the BatchLocalFile Connectivity Map properties contains the top-level parameters displayed in Table 50.

Table 50 Connectivity Map - BatchLocalFile - Target Location

Name	Description	Required Value
Append	Specifies whether to overwrite or append the data to the existing file. Use this parameter for outbound file transfers only. Choose the appropriate setting as follows: ▪ **Yes:** If the target file already exists, the data is appended to the existing file. ▪ **No:** The eWay overwrites the existing file on the remote system. If a file with the same name does not exist, both **Yes** and **No** create a new file on the external host.	Select **Yes** or **No** The configured default is **No**.
Target Directory Name	Specifies the directory name (path) on the local system from which files are retrieved or where they are sent. The value can be a literal, regular expression (source), or pattern name (destination). For outbound transfer (destination), the directory is created if it does not already exist. ***Note:*** For path separator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir. See **"Target Directory Name Is Pattern" on page 119**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**.	The directory name.
Target Directory Name Is Pattern	Specifies whether the Target Directory Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Target Directory Name" on page 119**.	Select **Yes** or **No** The configured default is **Yes**.

Table 50 Connectivity Map - BatchLocalFile - Target Location

Name	Description	Required Value
Target File Name	Specifies the name of the file on the local system either to be retrieved or sent. The value can be a literal, regular expression (get), or pattern name (put). See **"Target File Name Is Pattern" on page 120**. See **"Using Regular Expressions" on page 175** or **"Using Name Patterns" on page 178**	A file name.
Target File Name Is Pattern	Specifies whether the Target File Name represents a literal, or a regular expression or name pattern, as follows: ▪ **Yes**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **No**: indicates that the name value entered is a literal, an exact match. See **"Target File Name Is Pattern" on page 120**.	Select **Yes** or **No** The configured default is **No**.

3.12 BatchLocalFile Environment Properties

This section explains the properties for the **BatchLocalFile OTD, accessed from the Environment Explorer**.

The BatchLocalFile properties include the following sections:

3.12.1 General Settings (BatchLocalFile Environment)

The **General Settings** section of the BatchLocalFile Environment properties contains the top-level parameters displayed in Table 51.

Table 51 Environment - BatchLocalFile - General Settings

Name	Description	Required Value
State Persistence Base Location	Specifies a working directory for storing intermediary results. Options: ▪ Leave value blank: BatchLocalFile will use a default folder as the working directory. ▪ Specify a path to a local file system folder with read/write permissions. ***Note:*** See **"Deploying an EAR File" on page 24** if you are deploying the project to another application server.	A working directory with read/write permissions, or leave blank (no value) to accept a default directory.

3.12.2 Connection Pool Settings (BatchLocalFile Environment)

The **Connection Pool Settings** section of the BatchLocalFile Environment properties contains the top-level parameters displayed in Table 52.

Table 52 Environment - BatchLocalFile - Connection Pool Settings

Name	Description	Required Value
Steady pool size	Specifies the minimum number of physical connections the pool keeps available at all times.	An integer indicating the maximum number of connections available at all times. A value of **0** (zero) indicates that there are no physical connections in the pool and that new connections are created as needed. The configured default is **2**.
Maximum pool size	**Specifies the maximum number of physical connections the pool contains. 0 (zero) indicates that there is no maximum.**	An integer indicating the maximum number of connections allowed. A value of **0 (zero) indicates that there is no maximum**. The configured default is **10**.
Max Idle Timeout In Seconds	**Specifies the maximum idle timeout (in seconds)**. This is a hint to the server. A timer thread periodically removes unused connections. This parameter defines the interval at which this thread runs. This thread removes unused connections after the specified idle time expires. It allows the user to specify the amount of time a connection can remain idle in the pool.	An integer indicating **the maximum idle timeout in seconds**. When this is set to a number greater than **0**, the container removes or destroys any connections that are idle for the specified duration. A value of **0** specifies that idle connections can remain in the pool indefinitely. The configured default is **300** (5 minutes).

3.13 BatchRecord Connectivity Map Properties

This section explains the properties for the record-processing **BatchRecordOTD**. The BatchRecord properties include the following sections:

3.13.1 General Settings (BatchRecord Connectivity Map)

The **General Settings** section of the BatchRecord Connectivity Map properties contains the top-level parameters displayed in Table 53.

Table 53 Connectivity Map - BatchRecord - General Settings

Name	Description	Required Value
Parse or Create Mode	Specifies how this eWay Connection for the record-processing OTD is used. Set this parameter as follows: ▪ **Parse**: To use the OTD for parsing an inbound payload. ▪ **Create**: To use the OTD for creating an outbound payload. An instance of the OTD can be used for parsing an inbound payload (only) or for creating an outbound payload (only). A single OTD cannot be used for both purposes at the same time in the same Collaboration.	Select **Create** or **Parse**. The configured default is **Parse**.
Synchronized	Specifically applies to legacy Batch eWay Projects. Provides backward compatibility to allow Projects that were created using the Batch eWay version 5.0.7 or earlier to be imported and deployed without a change in the eWays behavior. The selections are: ▪ **Yes**: Provides backward compatibility for legacy (pre-5.0.8 Batch eWay) Projects. The eWay run in synchronized mode, one instance of the Collaboration after the other. ▪ **No**: For use with new Batch eWay Projects. The eWay run in parallel, creating multiple instances of the Collaboration that run in parallel. ***Note:*** All OTD instances used in a Project should have the same value for this property.	Select **Yes** or **No.** The default setting is **Yes, simulating Projects created with** Batch eWay version 5.0.7 or earlier.

3.13.2 Record (BatchRecord Connectivity Map)

This section allows you to configure the **Record** parameters, to specify the record characteristics you want the eWay to recognize.

The **Record** section of the BatchRecord Connectivity Map properties contains the top-level parameters displayed in Table 54.

Table 54 Connectivity Map - BatchRecord - Record

Name	Description	Required Value
Delimiter on Last Record	Allows you to supply the delimiter to be used with the final record. Use this parameter only when the **Record Type** is set to **Delimited**. Some message formats insist that the final message in a record set has no trailing delimiter. However, in most cases, you can safely leave this parameter set to **Yes**.	Select **Yes** or **No.** The configured default setting is **Yes**.
Record Delimiter	Specifies the delimiter to be used for records. Use this parameter when the **Record Type** is set to **Delimited**. The value entered is interpreted as a sequence of one or more bytes. If there are multiple bytes in the delimiter, each must be separated by a comma. You can enter the delimiters in the following formats: ▪ **ASCII Characters**: The eWay supports all ASCII characters. **Example**: *,*,* (records separated by ***) **Example**: | (records separated by a |) ▪ **Escaped ASCII**: The eWay supports \r, \n, \t, and \f. **Example**: \r,\n (records separated by CR NL) **Example**: \n (records separated by NL only) ▪ **Hex**: The eWay supports 0x00 to 0x7E **Example**: \0x0D,\0x0A (records separated by CR NL) ▪ **Octal**: The eWay supports 000 to 0177. **Example**: \015,\012 (same as \0x0D,\0x0A) ***Note:*** When using character delimiters with DBCS data, use single byte character(s), or equivalent hex values that do not coincide with either byte of the double byte characters. When using escaped ASCII, Hex, or Octal, the "\" character is required.	A valid character to use as data record delimiter.

Table 54 Connectivity Map - BatchRecord - Record (Continued)

Name	Description	Required Value
Record Size	Specifies a number indicating the record size (byte count). Use this parameter when the **Record Type** is set to **Fixed**, and a number indicating length must be supplied.	A number from **1** to **2,147,483,647** indicating the record size (byte count).
Record Type	Specifies the format of the records in the data payload in the Collaboration. Each payload contains zero or more records. Using this and related parameters, you can pass records individually to another component within eGate. Available options: ▪ **Delimited**: The records are separated by the delimiter specified under the **Record Delimiter** parameter. ▪ **Fixed**: The records are all of a given size, specified by the **Record Size** parameter. ▪ **Single Record**: If the payload is to be processed "as-is," select this option. ▪ **User Defined**: This option is not supported.	**Delimited**, **Fixed**, **or Single Record**. The configured default is **Delimited.**

3.14 BatchRecord Environment Properties

This section explains the properties for **BatchRecord, accessed from the Environment Explorer**.

The BatchRecord properties include the following:

3.14.1 Connection Pool Settings (BatchRecord Environment)

The **Connection Pool Settings** section of the BatchRecord Environment properties contains the top-level parameters displayed in Table 55.

Table 55 Environment - BatchRecord - Connection Pool Settings

Name	Description	Required Value
Steady pool size	Specifies the minimum number of physical connections the pool keeps available at all times.	An integer indicating the maximum number of connections available at all times. A value of **0** (zero) indicates that there are no physical connections in the pool and that new connections are created as needed. The configured default is **2**.
Maximum pool size	**Specifies the maximum number of physical connections the pool contains. 0 (zero) indicates that there is no maximum.**	An integer indicating the maximum number of connections allowed. A value of **0 (zero) indicates that there is no maximum**. The configured default is **10**.
Max Idle Timeout In Seconds	**Specifies the maximum idle timeout (in seconds)**. This is a hint to the server. A timer thread periodically removes unused connections. This parameter defines the interval at which this thread runs. This thread removes unused connections after the specified idle time expires. It allows the user to specify the amount of time a connection can remain idle in the pool.	An integer indicating **the maximum idle timeout in seconds**. When this is set to a number greater than **0**, the container removes or destroys any connections that are idle for the specified duration. A value of **0** specifies that idle connections can remain in the pool indefinitely. The configured default is **300** (5 minutes).

3.15 BatchInbound Connectivity Map Properties

This section explains the configuration parameters for the **BatchInbound eWay (OTD), accessed from the Connectivity Map** (there are no Environment properties for BatchInbound).

The BatchInbound properties include the following sections:

3.15.1 Settings (BatchInbound Connectivity Map)

The **Settings** section of the BatchInbound Connectivity Map properties contains the top-level parameters displayed in Table 56.

Table 56 Connectivity Map - BatchInbound- Settings

Name	Description	Required Value
Directory Name	Specifies the input directory name (path). It indicates the directory that the BatchInbound eWay polls for trigger or data files. The value can be a literal or a regular expression. ***Note:*** For path separator, use the forward slash "/" instead of the back slash "\". The eWay interprets the back slash as a special character. For example, use c:/temp/dir. See **"Directory Name is Pattern" on page 127**. See **"Using Regular Expressions" on page 175** .	The directory name.
Directory Name is Pattern	Specifies whether the Directory Name represents a literal, or a regular expression, as follows: ▪ **True**: indicates that the name value you enter is assumed to be a regular expression ▪ **False**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"Directory Name" on page 127**. ***Caution:*** Improper use may cause recursive matching.	Select **True** or **False.** The configured default is **False.**
File Name	Specifies the input filename. The value can be a literal or a regular expression. See **"File Name is Pattern" on page 128**. See **"Using Regular Expressions" on page 175** .	A file name.

Table 56 Connectivity Map - BatchInbound- Settings

Name	Description	Required Value
File Name is Pattern	Specifies whether the target file name represents a literal or a regular expression, as follows: ▪ **True**: indicates that the name value you enter is assumed to be a name pattern or regular expression. ▪ **False**: indicates that the name value entered is a literal, an exact match. No pattern matching or name expansion is done. See **"File Name" on page 127**.	Select **True** or **False.** The configured default is True.
Schedule Interval	Specifies the polling interval, or number of seconds between each poll of the input directory by the eWay for input files.	A number indicating the length of time in Milliseconds between eWay polls of the directory. The configured default is **5000** (or 5 seconds).

3.16 BatchInbound Environment Properties

This section explains the configuration parameters for the **BatchInbound eWay (OTD), accessed from the Environment Explorer**.

The BatchInbound properties include the following sections:

3.16.1 MDB Settings (BatchInbound Environment)

The **MDB Settings** section of the BatchInbound Environment properties contains the top-level parameters displayed in Table 57.

Table 57 Connectivity Map - BatchInbound- Settings

Name	Description	Required Value
Max Pool Size	Specifies the maximum size of the MDB (Message Driven Bean) pool.	**An integer indicating the maximum MDB pool size.** The configured default is **1000**.

3.17 Using FTP Heuristics

Due to the diversified nature of FTP server directory listing styles, that is, the format in which the list of directory entries are returned to the client as the result of a LIST command, the directory listing parser requires FTP server information regarding its specific format, to parse the result. These server specific parameters that provide format information are called FTP heuristics. When these heuristic parameters are set for a specific platform, it is called a directory listing style.

This section provides a general explanation of how the FTP Heuristics feature of the eWay operates, as well as some basic information on how to use it. It also explains the FTP Heuristics configuration parameters for the eWay.

FTP Heuristics

FTP heuristics are a set of parameters that the eWay uses to interact with external FTP daemons on a platform-specific level. The FTP heuristics create and parse both path and file names in the style required by the external system's platform (operating system).

Platform Selection

The BatchFTP OTD provides a number of built-in heuristics configuration styles. The style is selected from the BatchFTP eWay Connectivity Map property, **FTP > Directory Listing Style**. To select a style for your specific target platform do the following:

1. From the Connectivity Map, double-click the BatchFTP eWay. The FTPBatch eWay Properties Editor appears.
2. From the Editor's left pane, under the configuration tree, select **FTP**. The **FTP** parameters are now displayed in the Properties pane.
3. Click the **Directory Listing Style** field and select a system platform from the drop-down menu. The properties available from the drop-down menu, reflect the styles listed in the FtpHeuristics.cfg file.

The eWay's FTP Heuristics support the following platform types:

- UNIX
- AS400
- AS400-UNIX
- HCLFTPD 6.0.1.3
- HCLFTPD 5.1
- HP NonStop/Tandem
- MPE
- MSFTPD 2.0
- MSP PDS (Fujitsu)

- MSP PS (Fujitsu)
- MVS GDG (Generation Data Group)
- MVS PDS (Partitioned Data Sets)
- MVS Sequential
- NetWare 4.11
- Windows NT 3.5
- Windows NT 4.0
- UNIX
- UNIX (EUC-JP)
- UNIX (SJIS)
- User Defined
- User Defined1-10 (See **"Creating User Defined Heuristic Directory Listing Styles" on page 132**)
- VM/ESA
- VMS
- VOS3 PDS (Hitachi)
- VOS3 PS (Hitachi)
- VOSK (Hitachi)
- *User defined name and heuristic configuration (See* **"Creating User Defined Heuristic Directory Listing Styles" on page 132**

Note: *When using FTP with an AS400 UNIX (UFS) system, some specific FTP configuration settings are required (see* **"FTP Configuration Requirements for AS400 UNIX (UFS)" on page 144***).*

The following systems support Japanese mainframes:

- MSP PDS (Partitioned Data Sets)
- MSP PS (Physical Sequential)
- VOS3 PDS (Partition Data Sets)
- VOS3 PS (Physical Sequential)
- VOSK

IBM IP stack required for MVS Sequential, MVS GDG, and MVS PD

The FTP Heuristic methods used to interface with **MVS Sequential**, **MVS GDG**, and **MVS PDS,** are designed for FTP servers (at the mainframe) that use the **IBM IP** stack. Therefore, when you use FTP with an **MVS Sequential**, **MVS GDG**, or **MVS PDS** file system on a mainframe computer, you need to make sure that the FTP server is using an **IBM IP** stack. If any other IP stack is in place, the FTP heuristics feature will not work or may require modification.

3.17.1 Creating User Defined Heuristic Directory Listing Styles

You can create "user defined" heuristic configurations that allow you to interface with other platforms that are not listed in the Directory Listing Styles. The Batch eWay includes a mechanism that allows you to configure a set of heuristic properties so that the underlying parser can parse the LIST command result correctly. These properties are described under **"FTP Heuristics Configuration Parameters" on page 135**.

There are two methods for creating custom user defined directory listing styles:

- **Create a Custom Heuristics Configuration File**: You can create a custom user defined heuristics configuration file, listing the style names and parameters in the same format as the FtpHeuristics.cfg file. This file is then located on the logical host. The configuration file location and the style name are then specified in the BatchFTP configuration properties (see **"Creating a Custom Heuristics Configuration File" on page 132**).
- **Modify the FTP Heuristics Configuration File**: You can open FtpHeuristics.cfg file, add your user-defined style, and repackage the file. This method requires you to unzip a JAR file, add your custom style, and repackage the files (see **"Modifying the FTP Heuristics Configuration File" on page 133**). In many cases, this method may be more intrusive and cumbersome than the method listed above.

Creating a Custom Heuristics Configuration File

To create your own custom heuristics configuration file, do the following:

1. Using a text editor, create a user defined configuration file containing the property settings required to interface with your target platform. You can do this by copying a section (style) from the FtpHeuristics.cfg file that is similar to the style (platform parameter settings) that you are creating, or you can copy the format provided under **"Heuristics Configuration File Format" on page 134**.
2. Save your user defined configuration, as a CFG file, to a safe location on the logical host.
3. From the BatchFTP Environment properties, select the **FTP > User Defined Heuristics Configuration File** property, and enter the location and name of your user defined heuristics configuration file (for example **C:\USER_DEFINED_HEURISTICS\UDH.cfg**).
4. From the BatchFTP Connectivity Map properties, select **FTP > User Defined Directory Listing Style**, and enter the name of your user-named style (for example **MY AS400-UNIX**). You are allowed to list one user-named style. This style is now the configured Directory Listing Style, superseding the value of the Directory Listing Style property.

You can use this method to create multiple user-named styles by adding the styles to your user defined configuration file, and entering the different user defined style names in the Connectivity Map properties for each of your various FTPBatch component eWays.

You can also create multiple user defined configuration files if necessary, but this requires the creation of additional BatchFTP External Systems in the Environment. If

you chose this method, you must copy your Environment components (drag-and-drop) to the correct BatchFTP External System before applying Automap.

Considerations

If you decide to use this method for creating custom user defined heuristic configurations, take note of the following:

- The BatchFTP Connectivity Map property, **User Defined Directory Listing Style,** supersedes the **Directory Listing Style** property. When a **User Defined Directory Listing Style** is specified, it is used as the heuristic configuration for the corresponding BatchFTP eWay (OTD). To use the **Directory Listing Style** property value as the applied heuristic style, the **User Defined Listing Style** property value must be left blank.
- Setting the **User Defined Directory Listing Style** property value to blank (no value) makes the selected **Directory Listing Style** property value (built-in heuristic configuration) the current enabled style.
- At runtime, the user defined heuristics configuration file must exist on the logical host, and possess appropriate permission settings to allow the heuristic configuration parameters to be accessed by the deployed application.
- An error message is generated by the BatchFTP OTD when a **User Defined Directory Listing Style** is specified, but the **User Defined Heuristics Configuration File** property value is blank, or associated the user defined heuristics configuration file is not accessible or does not contain a corresponding style configuration.
- Setting the value of the **User Defined Directory Listing Style** triggers the loading of the corresponding heuristics configuration file specified by the **User Defined Heuristics Configuration File** property. If you make changes to the heuristics configuration file, set the **User Defined Heuristics Configuration File** property before setting the **User Defined Directory Listing Style**.

Modifying the FTP Heuristics Configuration File

To modify the **FtpHeuristics.cfg** file to include your user defined heuristic configuration styles, do the following:

1. The **FtpHeuristics.cfg** file is contained by the **stcbatch.jar** file, which is found in the following location:

   ```
   <JavaCAPS51>\edesigner\usrdir\modules\ext\batcheway\
   stcbatch.jar
   ```

   where *<JavaCAPS51>* is the Sun Java Composite Application Platform Suite install directory.
2. Unzip **stcbatch.jar** and locate the **FtpHeuristics.cfg** file.
3. Open **FtpHeuristics.cfg** with a text editor and add your user defined heuristic configuration styles.

Add User Defined Heuristic Configuration Styles

4 Copy the **User Defined** section (or any other section), and paste it to the bottom of FtpHeuristics.cfg.

5 Rename the section and each property name with your user-defined name or one of the available listings (User Defined1, User Defined2, and so forth). See the example provided under **"Heuristics Configuration File Format" on page 134**. In this example, the user defined name is MY AS400-UNIX). Only one style with a user-defined name can be specified, but 10 configuration styles can be named as User Defined1-10.

6 Modify the new section's properties for your target platform. See **"FTP Heuristics Configuration Parameters" on page 135** for property descriptions.

7 Repeat steps 2-4 above to create additional User Defined configurations.

Repackage the FtpHeuristics.cfg File.

8 Zip the **stcbatch.jar** file (including the updated **FtpHeuristics.cfg** file) and copy **stcbatch.jar** back to it's original location.

9 From the BatchFTP Configuration Map properties, select **FTP > User Defined > Directory Listing Style**, and enter the name of your user-named style (for example **MY AS400-UNIX**), or you can select any one of the 10 User Defined properties from the Directory Listing Style dropdown list (see **"Creating User Defined Heuristic Directory Listing Styles" on page 132**).

Your configuration changes will be applied to any Projects that are built and deployed with this Enterprise Designer.

Heuristics Configuration File Format

This example includes two user-named styles (MY AS400-UNIX, and UDH NT 4.0).

```
#
# ------------------------------------------------------------------------
#   Section:    MY AS400-UNIX
# ------------------------------------------------------------------------
#
MY AS400-UNIX!Commands Supported By FTP
Server!value=APPE%CWD%DELE%LIST%MKD%NOOP%PASS%QUIT%RETR%RNFR%RNTO%SITE%STOR%TYPE%USER!set=APPE%CWD
%DELE%LIST%MKD%NOOP%PASS%QUIT%RETR%RNFR%RNTO%SITE%STOR%TYPE%USER
MY AS400-UNIX!Header Lines To Skip!value=0!set=0
MY AS400-UNIX!Header Indication Regex Expression!value=!set=
MY AS400-UNIX!Trailer Lines To Skip!value=0!set=0
MY AS400-UNIX!Trailer Indication Regex Expression!value=!set=
MY AS400-UNIX!Directory Indication Regex Expression!value=!set=
MY AS400-UNIX!File Link Real Data Available!value=No!set=No%Yes
MY AS400-UNIX!File Link Indication Regex Expression!value=!set=
MY AS400-UNIX!File Link Symbol Regex Expression!value=!set=
MY AS400-UNIX!List Line Format!value=Fixed!set=Blank Delimited%Fixed
MY AS400-UNIX!Valid File Line Minimum Position!value=52!set=52
MY AS400-UNIX!File Name Is Last Entity!value=Yes!set=No%Yes
MY AS400-UNIX!File Name Position!value=52!set=52
MY AS400-UNIX!File Name Length!value=0!set=0
MY AS400-UNIX!File Extension Position!value=0!set=0
MY AS400-UNIX!File Extension Length!value=0!set=0
MY AS400-UNIX!File Size Verifiable!value=No!set=No%Yes
MY AS400-UNIX!File Size Position!value=0!set=0
MY AS400-UNIX!File Size Length!value=0!set=0
MY AS400-UNIX!Special Envelope For Absolute Pathname!value=!set=''
MY AS400-UNIX!Listing Directory Yields Absolute Pathnames!value=No!set=No%Yes
MY AS400-UNIX!Absolute Pathname Delimiter Set!value=///!set=///
MY AS400-UNIX!Change Directory Before Listing!value=Yes!set=No%Yes
MY AS400-UNIX!Directory Name Requires Terminator!value=No!set=No%Yes
#
#
# ------------------------------------------------------------------------
#   Section:   UDH NT 4.0
# ------------------------------------------------------------------------
```

```
#
UDH NT 4.0!Commands Supported By FTP
Server!value=APPE%CWD%DELE%LIST%MKD%NOOP%PASS%QUIT%RETR%RNFR%RNTO%SITE%STOR%TYPE%USER!set=APPE%CWD
%DELE%LIST%MKD%NOOP%PASS%QUIT%RETR%RNFR%RNTO%SITE%STOR%TYPE%USER
UDH NT 4.0!Header Lines To Skip!value=0!set=0
UDH NT 4.0!Header Indication Regex Expression!value=!set=
UDH NT 4.0!Trailer Lines To Skip!value=0!set=0
UDH NT 4.0!Trailer Indication Regex Expression!value=!set=
UDH NT 4.0!Directory Indication Regex Expression!value=<DIR>!set=<DIR>
UDH NT 4.0!File Link Real Data Available!value=No!set=No%Yes
UDH NT 4.0!File Link Indication Regex Expression!value=\.lnk$!set=\.lnk$
UDH NT 4.0!File Link Symbol Regex Expression!value=!set=
UDH NT 4.0!List Line Format!value=Blank Delimited!set=Blank Delimited%Fixed
UDH NT 4.0!Valid File Line Minimum Position!value=4!set=4
UDH NT 4.0!File Name Is Last Entity!value=Yes!set=No%Yes
UDH NT 4.0!File Name Position!value=4!set=4
UDH NT 4.0!File Name Length!value=0!set=0
UDH NT 4.0!File Extension Position!value=0!set=0
UDH NT 4.0!File Extension Length!value=0!set=0
UDH NT 4.0!File Size Verifiable!value=Yes!set=No%Yes
UDH NT 4.0!File Size Position!value=3!set=3
UDH NT 4.0!File Size Length!value=0!set=0
UDH NT 4.0!Special Envelope For Absolute Pathname!value=!set=
UDH NT 4.0!Listing Directory Yields Absolute Pathnames!value=No!set=No%Yes
UDH NT 4.0!Absolute Pathname Delimiter Set!value=\\\\\\!set=\\\\\\
UDH NT 4.0!Change Directory Before Listing!value=No!set=No%Yes
UDH NT 4.0!Directory Name Requires Terminator!value=No!set=No%Yes
```

3.17.2 FTP Heuristics Configuration Parameters

This section describes the configuration parameters for the **Batch FTP Heuristics** located in the FtpHeuristics.cfg file. The Batch FTP Heuristics configuration file (FtpHeuristics.cfg) contains the full set of parameters for each of the platforms listed under **"Platform Selection" on page 130**.

The FTP Heuristics configuration parameters are as follows:

Commands Supported by FTP Server

Description

Specifies the commands that the FTP server on the given host supports.

Required Values

One or more FTP commands as selected from the list.

Header Lines To Skip

Description

Specifies the number of beginning lines from a **LIST** command to be considered as a potential header (subject to the **Header Indication Regex Expression** configuration parameter, discussed below) and skipped.

Required Values

A non-negative integer. Enter zero if there are no headers.

Additional Information

In the example below, the line "total 6" comprises a one-line header.

```
total 6
-rw-r-----   1 ed        usr           110 Apr 15 13:43 AAA
-rw-r--r--   1 ed        usr           110 Apr 15 13:33 aaa
```

Header Indication Regex Expression

Description

Specifies a regular expression used to help identify lines which comprise the header in the output of a LIST command. All the declared lines of the header (see **Header Lines To Skip**, above) must match the regular expression.

Required Values

A regular expression. The default varies based on the FTP server's operating system. If there is no reliable way of identifying the header lines in the **LIST** command's output, leave this parameter undefined.

Additional Information

The regular expression "^ *total" indicates that each line in the header starts with "total," possibly preceded by blanks, for example:

```
total 6
-rw-r-----   1 ed        usr           110 Apr 15 13:43 AAA
-rw-r--r--   1 ed        usr           110 Apr 15 13:33 aaa
```

If the regular expression is undefined, then the header is solely determined by the value of the configuration parameter **Header Lines To Skip**.

Trailer Lines To Skip

Definition

Specifies the number of ending lines from a **LIST** command that are to be considered as a potential Trailer (subject to the **Trailer Indication Regex Expression**) and skipped.

Required Values

A non-negative integer. Enter zero if there are no trailers.

Trailer Indication Regex Expression

Definition

Specifies the regular expression used to help identify lines which comprise the trailer in the output of a **LIST** command. All the declared lines of the trailer (see **Trailer Lines To Skip**) must match the regular expression.

Required Values

A regular expression. If there is no reliable way of identifying the trailer lines in the **LIST** output, then leave this parameter undefined.

Additional Information

If the regular expression is undefined, then the header is determined solely by the value of the **Trailer Lines To Skip** configuration parameter.

Directory Indication Regex Expression

Definition

Specifies a regular expression used to identify external directories in the output of a **LIST** command. Directories cannot be retrieved and must be filtered out of the file list.

Required Values

A regular expression. If there is no reliable way of identifying the directory in the **LIST** output, then leave this parameter undefined.

Additional Information

The regular expression "^ *d" specifies that a directory is indicated by a line starting with the lowercase 'd,' possibly preceded by blanks, for example:

```
drwxr-xr-x   2 ed    usr     2048 Apr 17 17:43 public_html
```

File Link Real Data Available

Definition

Specifies whether a file may be a file link (a pointer to a file) on those operating systems whereon an FTP server will return the data for the real file as opposed to the content of the link itself.

Required Values

Yes or **No**.

File Link Indication Regex Expression

Definition

Specifies a regular expression that identifies external file links in the output of a **LIST** command. File links are pointers to the real file and usually have some visual symbol, such as ->, mixed in with the file name in the output of the **LIST** command. Only the link name is desired within the returned list.

Required Values

A regular expression. If there is no reliable way of identifying a file link within a **LIST** output, then leave this parameter undefined.

Additional Information

The regular expression "^ *l" specifies that a file link is indicated by a line starting with the lowercase "l," preceded possibly by blanks, for example:

```
lrwxr-xr-x   2 ed        usr   2048 Apr 17 17:43 p ->
public_html
```

File Link Symbol Regex Expression

Definition

Specifies a regular expression that parses the external file link name in the output of a **LIST** command. Only the link name is required for the file list to be returned.

Required Values

A regular expression. If there is no reliable way of identifying a file link within a **LIST** output, then leave this parameter undefined.

Additional Information

The regular expression "[ ] ->[ ]" defines that a file link symbol is represented by an arrow surrounded by spaces (" -> "). When parsed, only the file name to the right of the symbol is used.

In the following example, only the **public_html** would be used, not the "p" character:

```
lrwxrwxrwx   2 ed        usr  4 Apr 17 17:43 p -> public_html
```

List Line Format

Definition

Specifies whether fields in each line are blank delimited or fixed, that is, whether information always appears at certain columns.

Required Values

Blank Delimited or **Fixed**.

Additional Information

Even though some lines appear to be blank delimited, be wary of certain fields continuing their maximum value when juxtaposed with the next field without any separating blank. In such a case, we recommend you declare the line as "Fixed," for example:

```
-rw-r--r--   1 ed        usr           110 Apr 15 13:33 aaa
^^^^^^^^^^   ^ ^^        ^^^           ^^^ ^^^ ^^ ^^^^^ ^^^
     1       2  3         4             5   6  7    8    9
```

Valid File Line Minimum Position

Definition

Specifies the minimum number of positions (inclusive) a listing line must have in order to be considered as a possible valid file name line.

Required Values

For a **Fixed** list line format, enter a value equal to the number of columns, counting the first column at the far left as column 1. For a **Blank Delimited** list line format, enter a value equal to the number of fields, counting the first field on the far left as field 1.

For either case, if no minimum can be determined, set this value to zero (0).

Additional Information

For example, in the **Blank Delimited** line below, the minimum number of fields is 9:

```
-rw-r--r--   1 ed        usr           110 Apr 15 13:33 aaa
^^^^^^^^^^   ^ ^^        ^^^           ^^^ ^^^ ^^ ^^^^^ ^^^
     1       2  3         4             5   6  7    8    9
                                                File Name
```

Note: *The URL FTP Proxy will fail on ascertaining file names that have leading blanks, trailing blanks, or both.*

File Name Is Last Entity

Definition

Specifies whether the file name is the last entity on each line. This allows the file name to have imbedded blanks (however, leading or trailing blanks are not supported).

Required Values

Yes or **No**.

File Name Position

Definition

Specifies the starting position (inclusive) of a file name.

Required Values

For **Fixed** list line format, enter the column number, counting the first column on the far left as column 1. For **Blank Delimited** list line format, enter the field number, counting the first field on the extreme left as field 1.

Additional Information

For **Blank Delimited** List Line Format only, if the file name has imbedded blanks, then it can span over several fields, for example:

```
-rw-r--r--   1 ed        usr           110 Apr 15 13:33 aaa
^^^^^^^^^^   ^ ^^        ^^^           ^^^ ^^^ ^^ ^^^^^ ^^^
    1        2  3         4             5   6   7    8    9
                                                  File Name
```

File Name Length

Definition

Represents the maximum width of a file name; valid only for **Fixed** list line format.

Required Values

Enter one of the following:

- **An Integer:** Positive lengths imply that the file name is right-justified within the maximum field width, and thus leading-blanks are discarded.
- **Negative Lengths:** That is, compared to the absolute length, imply that the file name is left-justified and trailing-blanks are discarded.
- **Zero (0) Value Length:** If the file name is at the end of a file listing line, this value implies that the file name field extends to the end of the line.

Note: *For Blank Delimited list line format, this value is usually zero (0). However, if the File Name Length parameter is supplied even though a Blank Delimited list line format is specified, this implies that if the file name field exceeds the given length, then the rest of the List Line data occurs on the following line.*

File Extension Position

Definition

Specifies the left-most position of the file extension for those operating systems that present the file name extension separated from the main file name.

Required Values

For **Fixed** list line format, enter the column number, counting the first column at the extreme left as column 1. For **Blank Delimited** list line format, enter the field number, counting the first field at the far left as field 1. If there is no file extension (as on UNIX systems) set the value to zero (0).

File Extension Length

Definition

Specifies the maximum width of the file extension; valid only for **Fixed** list line format.

Required Values

Enter one of the following:

- **An Integer**
- **Positive Lengths:** Imply that the file extension is right-justified within the maximum field width and therefore leading-blanks are discarded.
- **Negative Lengths:** Imply that the file extension is left-justified and trailing-blanks are discarded (the absolute length is used).
- **Value of Zero (0):** *Always* for the **Blank Delimited** list line format.

File Size Verifiable

Definition

Specifies whether the file size is verifiable, significant, and accurate within a directory listing.

Required Values

Yes or **No**. The **File Size Stability Check** configurable parameter must also be enabled.

Additional Information

Even if the file size field of a listing line is not significant (that is, it is there but only represents an approximate value), the value of this parameter must be **No**. However, the file size location must still be declared in the **File Size Position** parameter below to assist determining which line of listing represents a valid file name, for example:

```
-rw-r--r--   1 ed        usr           110 Apr 15 13:33 aaa
                                       ^^^
                                       File Size
```

Note: *Use of this parameter does not guarantee that the file is actually stable. As this feature is intended only for backward compatibility with previous FTP implementations, we do not recommend that you rely on this functionality for critical data.*

File Size Position

Definition

Specifies the left-most position in the listing line that represents the size of the file. Even though for some operating systems the value shown might not truly reflect the file size, this position is still important in ascertaining that the line contains a valid file name.

Required Values

A non-negative integer. For **Fixed** list line format, the position value is the column number (starting with one (1) on the far left). For **Blank Delimited**, this value represents the field number (starting with one (1) on the far left). If the **LIST** line does not have a size field, set this parameter to zero (0).

Example

```
 -rw-r--r--   1 ed        usr           110 Apr 15 13:33 aaa
 ^^^^^^^^^^   ^ ^^        ^^^           ^^^ ^^^ ^^ ^^^^^ ^^^
      1        2  3         4             5   6  7    8    9
                                      File
                                         Size
```

The following text represents valid number representations of file sizes:

```
1234 or 1,234,567 or -12345 or +12345 or '  1234  ' or 12/34
or 1,234/56
```

The following text represents invalid number representations of file sizes (the ^ indicates where the error occurs):

```
'12 34' or 123,45,678 or 123-456-789 or --123 or 123-
   ^         ^                  ^        ^         ^
or 12345678901 or any number > 4294967295 or < -2147483647
   ^ (too large)
or 123.45 or 12AB34 or 0x45 or ,123,456 or 12//34
      ^        ^        ^      ^               ^
or /123 or 123/ or 12,3/45
   ^          ^        ^
```

File Size Length

Definition

Specifies the maximum width (number of columns) of the file size field, only valid for **Fixed** List Line Format.

Required Values

A non-negative integer. For **Blank Delimited** list line format, set this value to zero (0).

Special Envelope For Absolute Path Name

Definition

Specifies special enveloping characters required to surround an absolute path name (for example, single quotes are used in MVS). Only use a single quote at the start of the directory name.

Required Values

A pair of enveloping characters. Even if the leading and trailing character is identical, enter it twice.

If no enveloping characters are required for an operating system, leave this parameter undefined.

Note: *On UNIX, this parameter is always undefined.*

Listing Directory Yields Absolute Path Names

Definition

Specifies whether, when the **DIR** command is used on a directory name, the resulting file names are absolute.

Required Values

Yes or **No**.

Note: *On UNIX, this character is always set to No.*

Absolute Path Name Delimiter Set

Definition

Specifies any absolute path requiring certain delimiters to separate directory names (or their equivalent) from each other and from the file name.

Required Values

Enter the delimiters for the absolute path, starting from the left, for:

- Initial (left-most) directory delimiter
- Intermediate directory delimiters
- Initial (left-most) file name delimiter
- Optionally, the ending (right-most) file name delimiter

Wherever there is no specific delimiter, use "\0" (backslash zero) to act as a placeholder. Delimiters that are backslashes need to be escaped with another backslash (see Table 58).

Table 58 Delimiters and Path Naming by Platform

OS	Path Name Format	Delimiter Set				
		1	2	3	4	Enter
UNIX	/dir1/dir2/file.ext	/	/	/		///
Windows	C:\dir1\dir2\file.ext	\\	\\	\\		\\\\\\
VMS	disk1:[dir1.dir2]file.ext;1	[	.	]	;	[.];
MVS PDS	dir1.dir2(member)	\0	.	(	)	\0.()

Table 58 Delimiters and Path Naming by Platform (Continued)

OS	Path Name Format	Delimiter Set				
		1	2	3	4	Enter
MVS Sequential	dir1.dir2.filename	\0	.	.		\0..
MVS GDG	dir1.dir2.file(version#) (see Note)	\0	.	.		\0..
AS400	dir1/file.ext	\0	/	.		\0/.

Above, version # = 0 for current, +1 for new, -1 (-2, -3, etc.) for previous generations.

Change Directory Before Listing

Definition

Determines whether a change directory (**cd**) command needs to be done before issuing the **DIR** command to get a listing of files under the desired directory.

Required Values

Yes or **No**.

Note: *The current Batch eWay implementation does not rely on this parameter.*

Directory Name Requires Terminator

Definition

Determines whether a directory name that is not followed immediately by a file name requires the ending directory delimiter as a terminator (for example, as on VMS).

Required Values

Yes or **No**.

3.18 FTP Configuration Requirements for AS400 UNIX (UFS)

When using FTP with an **AS400 UNIX (UFS)** system, the following FTP configuration settings are required:

- FTP - Use PASV: **No** (see **"Use PASV" on page 38**)
- FTP Raw Commands - Pre Transfer Raw Commands: **site namefmt 1** (see **"Pre Transfer Raw Commands" on page 40**)

3.19 Dynamic Configuration

The **BatchFTP**, **BatchFTPOverSSL**, **BatchSCP** and **BatchSFTP** OTDs support automatic connection during initialization. Each of these OTDs require a number of properties to be set with valid values when Connection Mode is set to Automatic. This includes, but is not limited to, the following:

Environment Properties:

- Host Name
- Server Port
- User Name
- Password
- Any additional properties that are required for a successful connection.

These parameters must be set to valid values prior to using the BatchFTP OTD to allow the eWay to initialize successfully. After the initialization is successful, the parameters can be reconfigured from within the Collaboration Rule.

Dynamic configuration allows you to change configuration settings (based on the data input or Collaboration Rule logic) on the fly. Changes are made to the Collaboration using the Collaboration Editor. Make any necessary changes to the configuration settings and perform the **put** or **get**. The Project disconnects, reconnects with the new configuration settings, and performs the transfer.

Dynamic Configuration Sample

The following sample code demonstrates how to dynamically configure the eWay and perform a simple file transfer.

1. From BatchLocalFile:
   - Set the TargetDirectoryName

```
//@map:Copy "InDir" to TargetDirectoryName
BatchLocalFile_1.getConfiguration().setTargetDirectoryName( "InDir" );
```

2. From BatchFTP:
   - Disconnect the eWay

```
//@map:Client.disconnect
BatchFTP_1.getClient().disconnect();
```

   - Set the TargetDirectoryName

```
//@map:Copy "OutDir" to TargetDirectoryName
BatchFTP_1.getConfiguration().setTargetDirectoryName( "OutDir" );
```

   - Set the HostName

```
//@map:Copy "myftphostname" to HostName
BatchFTP_1.getConfiguration().setHostName( "myftphostname" );
```

   - Connect the eWay

```
//@map:Client.connect
BatchFTP_1.getClient().connect();
```

3. Perform a simple file transfer:
   - Get a local file

```
//@map:
BatchLocalFile_1.getClient().get();
```

- Assign the Payload

```
//@map:Copy Payload to Payload
BatchFTP_1.getClient().setPayload(BatchLocalFile_1.getClient().getPay
load() );
```

- Put a file on the FTP server

```
//@map:Client.put
BatchFTP_1.getClient().put();
```

To view the Collaboration Editor's Java Source Editor, click the **Advance mode** or **Source Code mode** icon, available on the Collaboration Editor toolbar.

3.19.1 Dynamic Configurable Parameters for Secure FTP OTDs

The secure Batch FTP OTDs contain several dynamic configurable parameters, which include (but are not limited to) the following:

BatchFTPOverSSL:

- CM (Connectivity Map) Link configuration > FTP and SSL Settings > Remote Directory
- CM Link configuration > FTP and SSL Settings > Remote Directory Name Is Pattern
- CM Link configuration > FTP and SSL Settings > Remote File
- CM Link configuration > FTP and SSL Settings > Remote File Name Is Pattern
- CM Link configuration > FTP and SSL Settings > Local Directory
- CM Link configuration > FTP and SSL Settings > Local Directory Name Is Pattern
- CM Link configuration > FTP and SSL Settings > Local File
- CM Link configuration > FTP and SSL Settings > Local File Name Is Pattern
- CM Link configuration > FTP and SSL Settings > Transfer Mode
- CM Link configuration > FTP and SSL Settings > Append
- CM Link configuration > FTP and SSL Settings > Local File Overwrite

BatchSCP:

- CM Link configuration > SCP Settings > Remote Directory
- CM Link configuration > SCP Settings > Remote File
- CM Link configuration > SCP Settings > Local Directory
- CM Link configuration > SCP Settings > Local File
- CM Link configuration > SCP Settings > Transfer Mode
- CM Link configuration > SCP Settings > Copy Recursive

BatchSFTP:

- CM Link configuration > SFTP Settings > Remote Directory
- CM Link configuration > SFTP Settings > Remote Directory Name Is Pattern

- CM Link configuration > SFTP Settings > Remote File
- CM Link configuration > SFTP Settings > Remote File Name Is Pattern
- CM Link configuration > SFTP Settings > Local Directory
- CM Link configuration > SFTP Settings > Local Directory Name Is Pattern
- CM Link configuration > SFTP Settings > Local File
- CM Link configuration > SFTP Settings > Local File Name Is Pattern
- CM Link configuration > SFTP Settings > Transfer Mode

Configuration Parameters that Accept Integer Values

The configuration parameters listed below can be configured from the Collaboration Editor by entering the specified integer values in the method parameters. The classes, **com.stc.connect.ssl.FTPSSLConstants** and **com.stc.connect.ssh.SSHConstants**, do not allow "incremental completion," that is, you must enter the value using the fully qualified name to access the constant.

For example, to set the BatchFTPOverSSL **Secure Mode** to **Explicit SSL**, from the Collaboration Editor, do the following:

1. From the Collaboration Editor toolbar, click **Source Code Mode**. The Collaboration Editor's **Java Source Editor** opens.
2. From the Business Rules tree (Business Rules pane) select the rule that contains the parameter or method that you want to configure. Selecting the rule highlights the corresponding code in the Java Source Editor. Find the code you wish to modify.
3. From the Java Source Editor, enter the value for the setting you require. For example, to set the BatchFTPOverSSL **SecureType** method to **Explicit SSL**, type **com.stc.connector.ssl.FTPSSLConstants.FTP_SECURE_TYPE_SSL_EXPLICIT** as the parameter value (see example below):

```
public void receive( com.stc.connector.appconn.file.FileTextMessage input,
com.stc.connector.batchadapter.appconn.ftps.FTPOverSSL BatchFTPOverSSL_1 )
        throws Throwable
    {
        if (!BatchFTPOverSSL_1.getClient().isConnected()) {
            logger.error( "Collab Start NOT CONNECTED ================ DO CONNECT" );
            BatchFTPOverSSL_1.getClient().connect();
            BatchFTPOverSSL_1.getConfiguration().setSecureType(
com.stc.connector.ssl.FTPSSLConstants.FTP_SECURE_TYPE_SSL_EXPLICIT );
        }
        BatchFTPOverSSL_1.getClient().get();
        if (BatchFTPOverSSL_1.getClient().isConnected()) {
            logger.error( "Collab End IS CONNECTED ================ DO DISCONNECT" );
            BatchFTPOverSSL_1.getClient().disconnect();
        }
    }
```

4. Once you have made your changes to the Collaboration, click the Commit Changes icon (from the Java Source Editor toolbar).

The OTD parameters listed below accept the following specified values:

BatchFTPOverSSL:

- CM Link configuration > FTP and SSL Settings > SecureType

- None:
  com.stc.connector.ssl.FTPSSLConstants.FTP_SECURE_TYPE_NONE
- Implicit SSL:
  com.stc.connector.ssl.FTPSSLConstants.FTP_SECURE_TYPE_IMPLICIT
- Explicit SSL:
  com.stc.connector.ssl.FTPSSLConstants.FTP_SECURE_TYPE_SSL_EXPLICIT

- Environment Link configuration > FTP and SSL Settings > KeyStoreType
  - JKS:
    com.stc.connector.ssl.FTPSSLConstants.KEY_STORE_TYPE_JKS
    (only one valid choice)
  - Other: (this is a place holder - reserved for future enhancement)
- CM Link configuration > FTP and SSL Settings > TransferMode
  - ASCII:
    com.stc.connector.ssl.FTPSSLConstants.FTP_TRANS_MODE_ASCII
  - BINARY:
    com.stc.connector.ssl.FTPSSLConstants.FTP_TRANS_MODE_BINARY

BatchSFTP:

- CM Link configuration > SFTP Settings > TransferMode
  - ASCII:
    com.stc.connector.ssh.SSHConstants.TRANS_MODE_ASCII
  - BINARY:
    com.stc.connector.ssh.SSHConstants.TRANS_MODE_BINARY

Chapter 4

Understanding Batch eWay OTDs

This chapter provides an overview of the Object Type Definitions (OTDs) available with the Batch eWay. The OTDs include fields that contain methods, properties, and their application.

What's in This Chapter

4.1 Overview of the Batch OTDs

An OTD contains a set of rules that define an object. The object encodes data as it travels through eGate. OTDs are used as the basis for creating Collaboration Definitions for a Project.

Each OTD acts as a template with a unique set of features of the eWay. The Batch eWay OTD template is not customizable and cannot be edited.

The four parts of an OTD are:

- **Element: An element** is the highest level in the OTD tree. The element is the basic container that holds the other parts of the OTD. The element can contain fields and methods.
- **Field**: Fields are used to represent data. A field can contain data in any of the following formats: string, boolean, int, double, or float.
- **Method**: Method nodes represent actual Java methods.
- **Parameter**: Parameter nodes represent the Java methods' parameters.

A high-level view of the OTD folder structure shows methods and attributes you can use in creating Business Rules that invoke FTP, secure FTP, batch record, or local file data exchange.

4.1.1 Types of Batch eWay OTDs

Table 59 shows the specialized OTDs available with the eWay.

Table 59 Batch eWay OTDs

OTD Name	Description
BatchFTP	Provides FTP access to remote systems.
BatchFTPOverSSL	Provides secure data transfer using FTP over SSL.
BatchSCP	Provides secure data transfer using Secure Copy Protocol with Secure Shell (SSH) as an underlying protocol.
BatchSFTP	Provides secure data transfer using SSH File Transfer Protocol or SFTP protocol.
BatchLocalFile	Provides easy access to local file systems.
BatchRecord	Allows the eWay to parse or create (or both) payloads of records in specified formats.
BatchInbound	Polls for input file, renames the file to a GUID, and triggers the Business Process or Collaboration.

This chapter describes each of these OTDs and how to use them with the eWay.

4.1.2 OTD Functions

OTDs provide the following functions:

- OTD are used in Collaborations to provide platform, system, and program specific functionality that allow you to create Business Rules.
- OTDs contain system specific parameters that can be configured using the Properties Editor.
- OTDs provide access to the information required to interface with specific external application.

All OTDs must be configured and administered using the Enterprise Designer. Any client components relevant to these OTDs have their own requirements. See the client system's documentation for details.

For the **BatchFTP**, **BatchLocalFile**, and **BatchRecord** OTDs, only those nodes for which there is a corresponding section in the Environment or Connectivity Map properties (From the Properties Editor) are implemented on the OTD. The remaining nodes are not implemented and are reserved for potential future use.

For the **BatchFTP**, **BatchLocalFile**, and **BatchRecord** OTDs, only those configuration parameters which appear in the Environment or Connectivity Map properties (From the Properties Editor) are supported for use on the OTD. The remaining configuration parameters are not implemented, and are reserved for potential future use. Even though an implemented configuration parameter might be accessed and used from a non-implemented node, such use is not recommended.

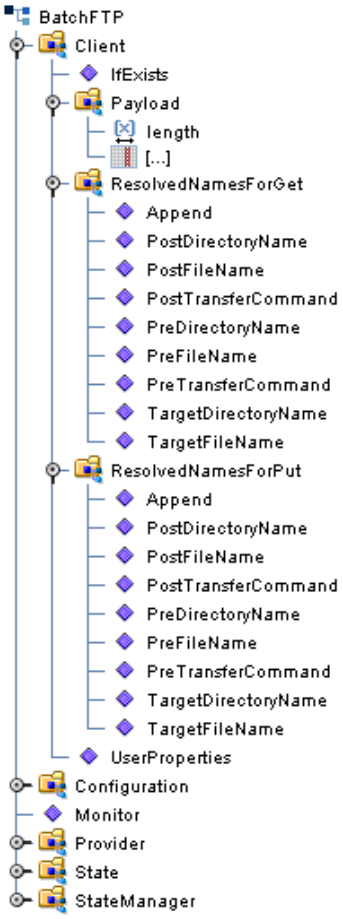
4.2 BatchFTP OTD

The Batch eWay includes four OTDs that allow you to perform FTP data-transfer functions. The **BatchFTP** OTD enables the eGate system to exchange data with other network hosts for the purpose of receiving and delivering objects stored in files. In addition, the **BatchFTPOverSSL**, **BatchSCP**, and **BatchSFTP** OTDs enable secure data transfer between the local host and a remote host using FTP over SSL and SSH.

BatchFTP OTD Structure

The BatchFTP OTD contains three top-level nodes, **Client, Configuration**, **Provider**, **State**, and **StateManager** (see Figure 4). Expand these nodes to reveal additional sub-nodes.

Figure 4 BatchFTP OTD Structure



Configuration Node

Each field sub-node in the **Configuration** node of the OTD corresponds to one of the eWay's FTP configuration parameters.

Client and Provider Nodes

This OTD includes two additional top-level nodes, the **Client** and **Provider**. These nodes implement their respective functionality interfaces in the eWay.

- The *client interface* represents how the functionality of the provider is actually used.
- The *provider interface* represents all of the general FTP operations that can be performed by the OTD.

4.2.1 BatchFTP OTD Node Functions

The following list provides an explanation of various nodes in the BatchFTP OTD, including primary functions:

- **BatchFTP**: Represents the OTD's root node.
- **Configuration**: Each field sub-node within this node corresponds to an eWay configuration parameter and contains settings information.
  - **InputStreamAdapter** and **OutputStreamAdapter**: Allow you to use and control the OTD's data-streaming features; see **"Streaming Data Between Components" on page 182** for details.

Note: *This OTD has configuration parameters that can be regular expressions. See* **"Using Regular Expressions" on page 175** *for details.*

- **Client**: This node contains the following sub-nodes, which implement the eWay's client interface in the OTD (FtpFileClient):
  - **Payload**: An in-memory buffer that contains the payload or message data you want to transfer by FTP, in the form of a byte array.
  - **UserProperties**: Only used if you have provided a user-defined implementation of the **FtpFileClient** interface; in such cases, the node represents the properties specified in the configuration.

Note: *You can transfer data using the Payload node or by using data streaming (InputStreamAdapter and OutputStreamAdapter nodes), but you cannot use both methods in the same OTD.*

  - **ResolvedNamesForGet** and **ResolvedNamesForPut**: Allow you to get the real file or directory name used during a transfer and perform an operation with it. For example, you could do a file transfer, with **get()** or **put(),** using the real name. You are able to retrieve the real file or directory name, even if these names have been expressed using regular expressions or special characters.

    These nodes contain sub-nodes that allow you to resolve file and directory names for target destinations, as well as names for pre- and post-transfer commands. See **"Pre/Post File Transfer Commands" on page 165**, **"Resolving Names" on page 179** for more information on these nodes. Also see **"Using Regular Expressions" on page 175** for more information on regular expressions.

- **get()**, **put()**, **reset()**, **connect()**, **disconnect()**, and **isConnected()**: See **"Essential BatchFTP OTD Methods" on page 154**.

- **Provider**: The sub-nodes contained in this node are methods that implement the eWay's provider interface in this OTD (**FtpFileProvider**). These methods allow you to do the general FTP operations that can be performed using the OTD.

4.2.2 Using the BatchFTP OTD

The BatchFTP OTD nodes allow you to configure specific eWay configuration parameters for the Collaboration controlling the FTP process. Once you have set the configuration parameters, you do not have to define the same parameters in each corresponding eWay component that uses this Collaboration.

Handling Type Conversions

The **Payload** node in the BatchFTP OTD is predefined as a byte array (**byte[]**). This definition allows the eWay to handle both binary and character data.

For example, you could use another OTD (such as an OTD from another eWay or a user-defined OTD) where the "data" node has been defined as a String (**java.lang.String**). If you were to map that String to the BatchFTP OTD's **Payload** node, the eGate Collaboration Editor can do an automatic type conversion and create code similar to that shown in the following example.

Note: *You must use care with this feature. While it works in many situations, there can be occasions when the default encoding causes errors in the translation.*

Code Conversion and Generation

For example, in a string-to-byte array conversion (or vice versa), the generated Java code could be:

```
getoutput().setPayload(STCTypeConverter.toByteArray
    (getinput().getBlob()));
```

or

```
getinput().setBlob(STCTypeConverter.toString
    (getoutput().getPayload()));
```

If you define the blob data as a byte array, no type conversion is necessary. When there is a conversion, the Collaboration Editor uses the Java Virtual Machine (JVM) default encoding to do the conversion to code, as shown in the previous examples.

Type Conversion Troubleshooting

As explained previously, the default encoding and translation works for many situations. There are cases, however (for example, binary data such as a .**zip** file), when the encoding could cause errors in the translation. Depending on the data character set and JVM default encoding, you *must* choose the appropriate encoding. In most cases, using the encoding string "ISO-8859-1" is the best choice.

To use this encoding, you can modify the code manually by adding the encoding String. Taking the previous examples, the resulting code using "ISO-8859-1" is:

```
getoutput().setPayload(STCTypeConverter.toByteArray
(getinput().getBlob(), "ISO-8859-1"));
```

or

```
getinput().setBlob(STCTypeConverter.toString
(getoutput().getPayload(), "ISO-8859-1"));
```

Using this String solves this type conversion problem. For more information, see the appropriate JVM encoding reference manuals.

Essential BatchFTP OTD Methods

In addition to the field elements, the BatchFTP OTD's **Client** node contains methods that extend the client interface functionality of the eWay. These methods are essential to the proper use of the OTD and require some additional explanation. They are:

- **get()**: Retrieves a file from the remote FTP server then stores its contents as a data payload. The method retrieves the first matching file based on the **Target Directory Name** and **Target File Name** parameters and stores the contents as a data payload (a byte array). It then performs any **Post Transfer Command**.

Note: *After this method call, you can get the payload's contents via the method getPayload().*

If no qualified file is available for retrieving, you get the exception containing **java.io.FTPFileException** as a nested exception.

- **put()**: Places the payload data on the FTP server, that is, it performs an append or put action from the **Payload** node to the remote FTP server and performs any **Post Transfer Command**.

  If no qualified file is available for sending, you get the exception containing **java.io.FTPFileException** as a nested exception.

Note: *When you are using the eWay's data-streaming feature, the get() and put() methods operate differently. See* **"Streaming Data Between Components" on page 182** *for details on this operation.*

- **reset()**: Allows you to return the **Client** node to its state immediately after the previous initialization.

Note: *The reset() method is available in the Batch FTP OTDs and BatchLocalFile OTD. The reset method must be called when the OTD has to be reused for another transfer during the same execution of executeBusinessRules() (for example, when you are using the Dynamic Configuration feature). The reset() method resets the content of the Client node without resetting the whole OTD. If you attempt another transfer without calling reset() first, the system throws an exception and makes an entry in the eWay's error log file.*

- **restoreConfigValues()**: Allows you to restore the configuration parameter defaults to the related eWay configuration.

- **connect()**, **disconnect()**, and **isConnected()**: Perform connection-related operations with respect to the FTP server.

Sequence Numbering

The sequence numbering feature allows you to set up the FTP target directory or file name to contain a sequence number. You can set the starting and maximum sequence numbers using the eWay configuration parameters for the OTD.

This parameter is used for the name pattern **%#**.

This feature is also available with the BatchLocalFile OTD. For more information on these configuration parameters, see **"Sequence Numbering (BatchFTP Connectivity Map)" on page 41**.

Additional FTP File Transfer Commands

The BatchFTP OTD also allows you to enter commands to be executed directly before and after the file transfer operation. See **"Pre/Post File Transfer Commands" on page 165** for details.

4.3 BatchFTPOverSSL OTD

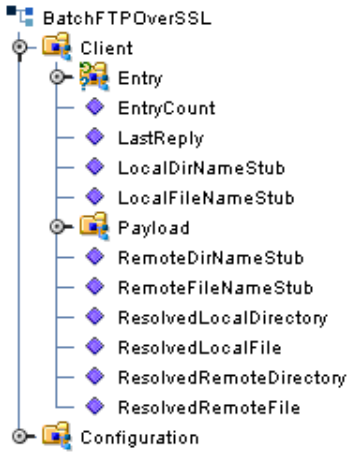
The Batch Secure FTP over SSL OTD (BatchFTPOverSSL) provides secure data transfer using Secure Sockets Layer (SSL) protocol.

For information about configuring external FTP servers, SSL servers, and so forth, refer to the application's documentation.

BatchFTPOverSSL OTD Structure

The BatchFTPOverSSL OTD contains two top-level nodes, **Client** and **Configuration** (see Figure 5). Expand these nodes to reveal additional sub-nodes.

Figure 5 BatchFTPOverSSL OTD Structure



4.3.1 BatchFTPOverSSL OTD Node Functions

The following list provides an explanation of various nodes in the BatchFTPOverSSL OTD, including primary functions:

BatchFTPOverSSL: Represents the OTD's root node.

Configuration Node

The BatchFTPOverSSL sub-nodes under the **Configuration** node correspond to BatchFTPOverSSL eWay's Connectivity Map and Environment configuration parameters.

BatchFTPOverSSL Client Node

The Client node contains sub-nodes that implement the eWay's client interface in the OTDs. The BatchFTPOverSSL Client node includes the following methods:

- **append()**: transfers data to the remote in append mode. The source of the data is determined by the configuration parameters **Local Directory** and **Local File**. If both are empty, then the data is from pay load.
- **connect()**: connects to the FTP server and does authentication as configured.
- **deleteDir(String dir)**: deletes the remote directory as specified by the argument.
- **deleteFile(String path)**: deletes the remote file as specified by the argument.
- **disconnect()**: disconnects from the FTP server.
- **doRawCommands(String commands)**: specifies the raw commands.
- **download()**: downloads data from the remote (specified in configuration parameters **Remote Directory** and **Remote File**) to the local (specified in configuration parameters **Local Directory** and **Local File**).
- **get()**: copies the file or directory specified by the configuration parameters **Remote Directory** and **Remote File**, to the local, as specified by the configuration parameters **Local Directory** and **Local File**. If the configuration parameter, **Is Copy Recursive**, is set to **Yes**, the copy will be recursive.
- **getEntry()**: gets the index entry in the current entry list.
- **getEntryCount()**: returns the count of directory entries, as the result of invoking **listDir** or **listDirLong**.
- **getLastReply()**: gets the FTP response code as a String.
- **getPayload()**: returns the payload.
- **hasEntry()**: returns **false** if there is more to the entry in the directory listing result list (when the result list is exhausted), and calls **resetEntries** to re-iterate the result list again.
- **getResolvedLocalDirectory**(): returns the resolved local directory name.
- **getResolvedLocalFile**(): returns the resolved local file name.
- **getResolvedRemoteDirectory**(): returns the resolved remote directory name.

- **getResolvedRemoteFile**(): returns the resolved remote file name.
- **hasEntry()**: returns whether the current entry list has entries.
- **isConnected()**: determines if the Java Integration Suite is connected to the FTP server.
- **listDir()**: returns the entry under the remote directory (specified in configuration parameters **Remote Directory** and **Remote File**). The entry only contains the name. After this method is invoked, use methods such as **hasEntry**, **nextEntry**, **getEntry**, **getEntryCount**, and so forth. to iterate the entry information.
- **listDirLong()**: returns the entries under the remote directory (specified in configuration parameters **Remote Directory** and **Remote File**). The entry contains details such as **name**, **size**, **time stamp**, **is directory or not**, and so forth. After this method is invoked, use methods such as **hasEntry**, **nextEntry**, **getEntry**, **getEntryCount**, and so forth, to iterate the entry information.
- **mkdir(String dir)**: creates a directory on the remote. The name of the directory is specified in the configuration parameters.
- **nextEntry()**: returns the next entry in the result list.
- **put()**: same as **get**, but the data transfer is from local to remote.
- **renameFile(String newName)**: renames the file specified by the configuration parameters **Remote Directory** and **Remote File** to a new name (arg).
- **reset()**: resets the internal life cycle methods, such as discard payload buffer.
- **resetEntries()**: resets the result list iterator so that it can be iterated again.
- **resolveLocalAsDestination()**: resolves the local directory name and local file name if they are patterns (used to generate real directory and file name for data transfer destination), upon the success of the resolution.
- **resolveLocalAsSource()**: resolves the local directory and file if they are regular expressions (filters for data transfer source); upon the success of the resolution.
- **resolveRemoteAsDestination()**: resolves the remote directory name and remote file name if they are patterns (used to generate real directory and file name for data transfer destination), upon the success of the resolution.
- **resolveRemoteAsSource()**: resolves the remote directory and file if they are regular expressions (filters for data transfer source); upon the success of the resolution.
- **setpayload(byte[] newPayload)**: sets the payload as specified by the argument.
- **setResolvedLocalDirectory(String s)**: sets the resolved local directory name.
- **setResolvedLocalFile(String s)**: sets the resolved local file name.
- **setResolvedRemoteDirectory(String s)**: sets the resolved remote directory name.
- **setResolvedRemoteFile(String s)**: sets the resolved remote file name.
- **upload()**: uploads data from the local (specified in configuration parameters **Local Directory** and **Local File**) to the remote (specified in configuration parameters **Remote Directory** and **Remote File**).

Note: *See the Batch eWay Javadoc for a list of all exposed FTPOverSSLClient methods.*

4.4 BatchSFTP OTD

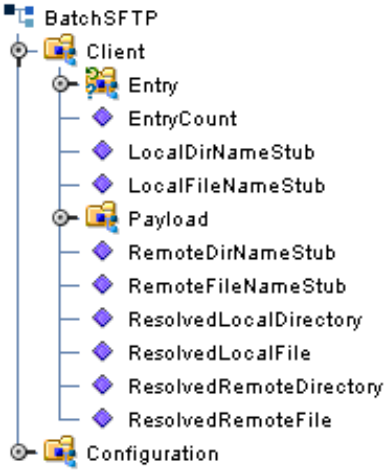
The Batch Secure SFTP OTD (BatchSFTP) provides secure data transfer using SSH File Transfer Protocol or SFTP. SFTP provides a range of operations on remote files, such as resuming interrupted transfers, directory listings, and remote file removal.

SFTP is one means of securely transferring computer files between a local and a remote host or between two remote hosts, using the Secure Shell (SSH) protocol. The BatchSFTP OTD uses SFTP to copy a file to or from a remote host.

BatchSFTP OTD Structure

The BatchSFTP OTD contains two top-level nodes, **Client** and **Configuration** (see Figure 6). Expand these nodes to reveal additional sub-nodes.

Figure 6 BatchSFTP OTD Structure



4.4.1 BatchSFTP OTD Node Functions

The following list provides an explanation of various nodes in the BatchSFTP OTD, including primary functions:

BatchSFTP: Represents the OTD's root node.

Configuration Node

The BatchSFTP sub-nodes under the **Configuration** node correspond to the BatchSFTP eWay's Connectivity Map and Environment configuration parameters.

BatchSFTP Client Node

The BatchSFTP OTD's Client node includes the following methods:

- **cd(**String dir**)**: changes the remote current directory to the specified path.
- **connect()**: connects to the remote SSH server and does authentication as configured.
- **delete()**: deletes remote files specified by configuration parameters *RemoteDirectory* and *RemoteFile*.
- **delete(**String file**)**: deletes remote files specified by file.
- **disconnect()**: disconnects the client from the remote SSH server.
- **get()**: copies data from a remote SSH server (specified by configuration parameters RemoteDirectory and RemoteFile) to the local machine. Depending on the current status of the configuration, the remote data can be stored into the payload (in memory buffer) or a local file specified by configuration parameters LocalDirectory and LocalFile.

Note: *Note that the remote could be a folder. In this case, if the configuration parameter Recursive is "Yes", the folder hierarchy is copied to the local destination.*

- **getEntry(int index)**: gets the index entry in the current entry list.
- **getEntryCount()**: returns the number of entries in the current entry list.
- **getPayload()**: returns the payload.

Note: *If you call getPayload() when using the BatchSFTP OTD, and the Local Directory and Local Filename are set, getPayload() returns null, even if a file has been retrieved.*

- **getResolvedLocalDirectory**(): returns the resolved local directory name.
- **getResolvedLocalFile**(): returns the resolved local file name.
- **getResolvedRemoteDirectory**(): returns the resolved remote directory name.
- **getResolvedRemoteFile**(): returns the resolved remote file name.
- **hasEntry()**: returns whether the current entry list has entries.
- **isConnected()**: determines if the Java Integration Suite is connected to the SSH server.
- **lcd(**String dir**)**: changes the local current directory.
- **listDir**(): lists all the entries under remote current directory.
- **lpwd()**: returns the local current path.
- **mkdir()**: creates a directory on the remote. The name of the directory is specified in the properties.
- **mkdir(String dir)**: creates a directory on the remote using the name specified in the configuration parameters, **Remote Directory** and **Remote File**.
- **nextEntry()**: returns the next entry in the current entry list.

- **put()**: copies local data (specified by configuration parameters LocalDirectory and LocalFile) to the remote SSH server (specified by configuration parameters RemoteDirectory and RemoteFile). Depending on the current status of the configuration, the local data can be from either a payload or local file.

Note: *Note that the local could be a folder. In this case, if the configuration parameter Recursive is "Yes", the folder hierarchy is copied to the remote destination.*

- **pwd()**: returns the remote current path.
- **rename(String newPath)**: renames the file or directory specified by the old name (first argument), to new name (second argument).
- **rename(String oldPath, String newPath)**: renames the file or directory specified by configuration parameters, **Remote Directory** and **Remote File**, to new name (argument).
- **reset()**: resets the internal life cycle methods, such as discard payload buffer.
- **resetEntries()**: resets the current entry list so that next call to *nextEntry()* will return the first entry in the list.
- **resolveLocalAsDestination()**: resolves the local directory name and local file name if they are patterns (used to generate real directory and file name for data transfer destination), upon the success of the resolution.
- **resolveLocalAsSource()**: resolves the local directory and file if they are regular expressions (filters for data transfer source); upon the success of the resolution.
- **resolveRemoteAsDestination()**: resolves the remote directory name and remote file name if they are patterns (used to generate real directory and file name for data transfer destination), upon the success of the resolution.
- **resolveRemoteAsSource()**: resolves the remote directory and file if they are regular expressions (filters for data transfer source); upon the success of the resolution.
- **setpayload(**byte[] newPayload**)**: sets the payload.
- **setResolvedLocalDirectory(String s)**: sets the current resolved local directory name.
- **setResolvedLocalFile(String s)**: sets the current local file name to s, should not be invoked directly from user Collaboration.
- **setResolvedRemoteDirectory(String s)**: sets the current resolved remote directory name.
- **setResolvedRemoteFile(String s)**: sets the current resolved remote file name.

Note: *See the Batch eWay Javadoc for a list of all exposed SFTPClient methods.*

4.5 BatchSCP OTD

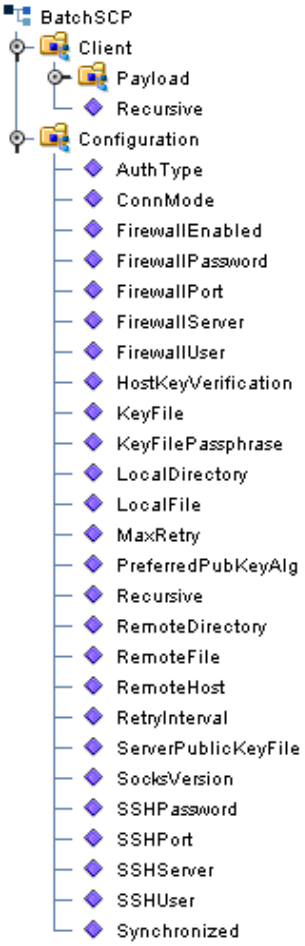
The Batch Secure SCP OTD (BatchSCP) provides secure data transfer using Secure Copy Protocol with Secure Shell (SSH) as an underlying protocol. SCP is similar to RCP (remote copy), but the file is copied over secure shell (SSH) rather than RSH (remote shell). When files are copied using SCP the data is encrypted during transfer.

For information about configuring external FTP servers, SSH servers, and so forth, refer to the application's documentation.

BatchSCP OTD Structure

The BatchSCP OTD contains two top-level nodes, **Client** and **Configuration** (see **Figure 7 on page 161**). Expand these nodes to reveal additional sub-nodes.

Figure 7 BatchSCP OTD Structure



4.5.1 BatchSCP OTD Node Functions

The following list provides an explanation of various nodes in the BatchSCP OTD, including primary functions:

BatchSCP: Represents the OTD's root node.

Configuration Node

The BatchSCP sub-nodes under the **Configuration** node correspond to BatchSCP eWay's Connectivity Map and Environment configuration parameters (see Figure 7).

BatchSCP Client Node

The BatchSCP OTD's Client node includes the following methods:

- **connect()**: connects to the SSH server and does authentication as configured.
- **disconnect()**: disconnects from the SSH server.
- **get()**: copies the file or directory specified by the configuration parameters, **Remote Directory** and **Remote File**, to the local, as specified by configuration parameters **Local Directory** and **Local File**. If the configuration parameter, **Is Copy Recursive**, is set to **Yes**, the copy will be recursive.
- **getPayload()**: returns the payload buffer as a byte array.
- **getRecursive()**: copies data from remote to local with configuration parameter Recursive set to "Yes".
- **isConnected()**: checks whether the client is connected to the SSH server.
- **put()**: Copies local data (specified by configuration parameters *LocalDirectory* and *LocalFile*) to the remote SSH server (specified by configuration parameters RemoteDirectory and RemoteFile).
- **putRecursive()**: copies data from local to remote with the configuration parameter Recursive set to "Yes".
- **setpayload(byte[] newPayload)**: sets the payload.

Note: *See the Batch eWay Javadoc for a list of all exposed SCPClient methods.*

4.6 BatchLocalFile OTD

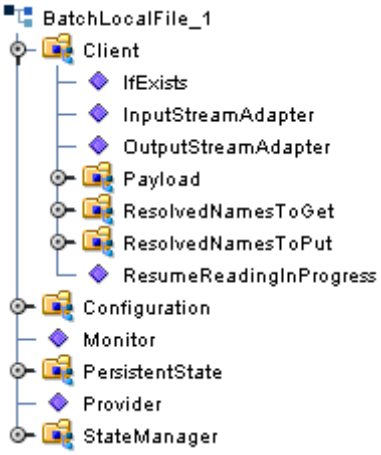
The BatchLocalFile OTD provides access to files on your local system. While file access is not always necessary in eGate, it makes sense for the Batch eWay to have this feature because file processing is one of its core functions.

Additional BatchLocalFile features include regular expressions for accessing files and a sequence-numbering scheme for creating files.

BatchLocalFile OTD Structure

The BatchLocalFile OTD contains four top-level nodes, **Client**, **Configuration**, **PersistentState**, and **State Manager** (see Figure 8). Expand these nodes to reveal additional sub-nodes.

Figure 8 BatchLocalFile OTD Structure



Configuration Node

As it iw with each of the Batch OTDs, each field sub-node under the **Configuration** node in the BatchLocalFile OTD corresponds to one of the eWay's configuration parameters for that OTD. See **"BatchLocalFile Connectivity Map Properties" on page 112** for details.

Client Node

This OTD includes an additional top-level node, the **Client**. This node implements its respective functionality interface in the eWay.

The *client interface* represents how the functionality of the OTD is actually used. This functionality includes the basic operations and features of the OTD. The client interface provides the OTD's the file services those who want to use them.

4.6.1 BatchLocalFile OTD Node Functions

The following list explains the nodes in the BatchLocalFile OTD, including primary functions:

- **Configuration**: The field sub-nodes within this node corresponds to an eWay configuration parameter and contains the corresponding settings information. See **"BatchLocalFile Connectivity Map Properties" on page 112** for details on these parameters and settings.

Note: *This OTD has configuration parameters that can be regular expressions. See* **"Using Regular Expressions" on page 175** *for details.*

- **Client**: The following sub-nodes, contained in this node, implement the eWay's client interface in the OTD (**LocalFileClient**):
  - **ResolvedNamesToGet** and **ResolvedNamesToPut**: Allow you to get the real file or directory name used during a transfer and perform an operation with it. For example, you could do a local file transfer, with **get()** or **put(),** using the real

name. You are able to retrieve the real file or directory name, even if these names have been expressed using regular expressions or special characters.

These nodes contain sub-nodes allowing you to resolve file and directory names for target destinations, as well as names for pre- and post-transfer commands (see **"Pre/Post File Transfer Commands" on page 165** for details).

See **"Using Regular Expressions" on page 175** and **"Using Name Patterns" on page 178** for more information on these features.

- **InputStreamAdapter** and **OutputStreamAdapter**: Allow you to use and control the OTD's data-streaming features; see **"Streaming Data Between Components" on page 182** for details.
- **Payload:** An in-memory buffer that contains the payload or message data you want to transfer by local file, in the form of a byte array.
- **get()**, **put()**, and **reset()**: See **"Essential BatchLocalFile OTD Methods" on page 166**.
- **ResumeReadingInProgress**: This node allows you to resume a data-streaming file transfer operation that was interrupted for whatever reason. These transfers occur piece by piece and usually involve large files. This feature allows you to resume at the same point where the transfer left off when it stopped.

Note: *You can transfer data using the Payload node or by using data streaming (InputStreamAdapter and OutputStreamAdapter nodes), but you cannot use both methods in the same OTD.*

4.6.2 Using the BatchLocalFile OTD

This section explains how to use the BatchLocalFile OTD's features.

There is no required particular order for the calls that can be made on the BatchLocalFile OTD. The only required call is **reset()** after a transfer, if it is used for more than one transfer per Collaboration Rule execution. An example of this would be a dynamic batch order with multiple files to be transferred.

If you are using a Java Collaboration to generate multiple files with sequence numbers, using the BatchLocalFile interface, you must call the **reset()** method to indicate the end of one file and the start of the next one.

BatchLocalFile Specific Features

Using the BatchLocalFile OTD to read records from a local file has the following advantages:

- **Data Streaming**: Allows your application to stream data directly to and from a local file system when used together with the BatchFTP OTD or the record-processing OTD. This feature minimizes the required RAM when large files are read, because the entire file is never loaded in memory.
- **Resume Reading**: Allows your application to read large files in a number of subsequent Business Rule executions, when you are using data streaming. This

operation is achieved by persisting information about the current successful file read operation and resuming the next read operation from that last stored position.

For more information on the Resume Reading feature, see **"Resume Reading Feature" on page 167**.

Pre/Post File Transfer Commands

The eWay has features that allow you to execute desired actions directly before or after the actual file transfer. You can enter these settings at the eWay configuration parameters or in the Configuration node of the desired OTD.

These features are available with both the BatchLocalFile OTD and the BatchFTP OTD.

Pre Commands

For an inbound transfer, the file can be made unavailable to other clients polling the target system with the same directory and file pattern or name (Rename). For an outbound transfer, you can perform an automatic backup of the existing file (Copy).

Your pre-transfer options are:

- **Rename**: Rename the target file for protection or recovery; you must provide a desired directory and file name. The directory is created if it does not already exist.
- **Copy**: Copy the target file for backup or recovery; you must enter a desired directory and file name.
- **None**: Do nothing.

Caution: *Each FTP server can behave differently when you are using Rename and a destination file already exists. For example, for some UNIX FTP servers, the destination file is overwritten without question. That is, no error or warning message is given. On other FTP servers, a Windows XP server for example, the system generates an error that results in exceptions being thrown in the called OTD method.*

Be sure you are familiar with the native behavior of the corresponding FTP server. If you are in doubt, try the action at the command prompt. If the action displays an error message, it is likely to result in the throwing of an exception in the Collaboration.

To gain proper protection, backup, or recovery, you must choose the appropriate setting that serves your purpose. For example, to recover from failures on an outbound appending transfer, use the **Copy** setting. When specifying file and directory names you can use regular expressions, special characters, or both.

Post Commands

For an inbound transfer, you can mark the transferred file as "consumed" by making an automatic backup (**Rename**) or by destroying it permanently (**Delete**). For an outbound transfer, you can make the transferred file available to other clients by renaming it. When specifying file and directory names you can use regular expressions, special characters, or both.

Your post-transfer options are:

- **Rename**: Rename the transferred file; you must provide a desired directory and file name.
- **Delete**: Delete the transferred file (inbound transfers only).
- **None**: Do nothing.

Note: *For an outbound transfer (publishing), the directory is created if it does not already exist.*

For more information on Pre and Post Transfer commands see the following:

BatchFTP OTD

- **Pre Transfer (BatchFTP Connectivity Map)** on page 33
- **Post Transfer (BatchFTP Connectivity Map)** on page 42

BatchLocalFile OTD

- **Pre Transfer (BatchLocalFile Connectivity Map)** on page 113
- **Post Transfer (BatchLocalFile Connectivity Map)** on page 116

Essential BatchLocalFile OTD Methods

In addition to the field elements, the BatchLocalFile OTD's **Client** node contains methods that extend the client interface functionality of the eWay. These methods are essential to the proper use of the OTD. These methods include:

- **get()**: Retrieves a local file then stores its contents as a data payload. The method retrieves the first matching file based on the **Target Directory Name** and **Target File Name** parameters and stores the contents as a data payload (a byte array). It then performs any **Post Transfer Command**.

  After this method call, you can get the payload's contents via the method **getPayload()**.
- **put()**: Stores the data payload (as a byte array) to a file. It then performs any **Post Transfer Command**.

  Before using this method call, you must set the file contents using the method **setPayload()**.

  The method throws an exception (**LocalFileException**) if an error occurs.
- **reset()**: Allows you to return the **Client** node to its state immediately after the previous initialization.

Note: *The reset() method is available in both BatchFTP and BatchLocalFile OTDs. It must be called when the OTD has to be reused for another transfer during the same execution of executeBusinessRules() (for example, when you are using the Dynamic Configuration feature). The reset() method resets the content of the Client node without resetting the whole OTD.*

Resume Reading Feature

The purpose of this feature is to allow an application to read large files in parts instead of processing the whole file at once. Resume Reading allows your system to read files in a number of subsequent Business Rule executions, when you are using data streaming.

General Operation

The Resume Reading feature's operation is achieved by keeping persistent information about the current successful file read operation, breaking, then resuming the next read operation from that last stored break position. As a result, the current file is read in parts, and the beginning and end of each part is determined by a predefined *break condition*.

You determine the break condition through the definition of your Business Rules. Since the Resume Reading feature operates based on reading one part of a file at a time per Business Rule, these rules must determine the break. Each Business Rule executes reading a part of the file, breaks, then passes to the next rule, which reads the next part up to the break, and so on, until the entire file is read.

A break condition can be any type of stopping point you determine in your Collaboration Rules. For example, this condition could be a fixed number of records, a delimiter, or reaching a specific character string.

The **Client** node in the OTD has a read-only property (**ResumeReadingInProgress** node) indicating whether there is a resume-reading operation in progress. This node is for informational purposes only. Also, the Resume Reading feature is available in the data-streaming mode only.

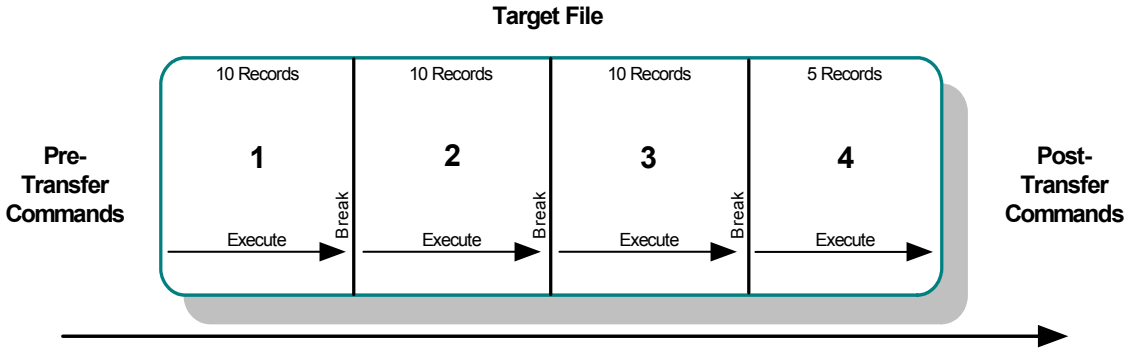
The feature has no special operational requirements besides setting the eWay configuration option. The eWay configuration has an option to enable or disable this feature. This option is also accessible at run time.

Note: *If this feature is enabled, the eWay always checks first for a resume-reading operation in progress. If this feature is not in progress, the eWay determines the next file based on the eWay configuration settings.*

Step-by-step Operation

Figure 9 shows a diagram of how the Resume Reading feature operates along with pre- and post-file-transfer commands. In this example, the Collaboration executes the same Business Rule four times. Each execution causes the Collaboration to read another section of the file. When the Collaboration reads the final records, it executes the Post-Transfer commands

Figure 9 Resume Reading Operation



In this example, the reading happens in the following steps:

1. The eWay starts reading the file then reaches a break condition after a partial data read (the end of **Part 1**), the eWay's pre-transfer commands have already been executed. The resume-reading state is stored, and no post-transfer commands are executed. The eWay is waiting for the next execution of the Business Rule.
2. The resume-reading operation is in progress but still attains only partial data reads. The eWay reads from one break condition to the next (**Part 2** and **Part 3** in the figure) The resume-reading state is stored in each case, and the eWay executes the Business Rule once per each part.
3. The resume-reading operation is in progress and completes its data read during the final execution of the Business Rule (**Part 4**). The eWay reads from a break condition to the end of a file. No resume-reading state is stored, and any post-transfer commands are then executed.

In all of the previous steps, the Business Rule is executed repeatedly, and the current read position in the file changes on each execution. If the file is smaller than **Part 1** in the figure, the eWay does not reach a break condition. The operation is normal, and no resume-reading state is stored. The pre- and post-transfer commands are executed.

Operation Without Resume Reading Enabled

If the Resume Reading feature is not enabled:

- **Data-read Stop Then Restart**: Any unread data at the end of the file is ignored.
- **Resume Reading in Progress**: If there is a resume-reading operation in progress from a previous execution, an error is generated, and an exception is thrown.

Note: *If there is a resume-reading operation in progress it cannot be interrupted and must be completed. The executeBusinessRules() method must be called enough times to fully consume the file. In other words, do not discontinue processing the file before it has been completely consumed.*

To Avoid Storing a Resume Reading State

Sometimes a partial data-stream read is necessary even when the Resume Reading feature is enabled. For example, there could be some application logic on top of the record parsers, which might abandon the rest of the file because of a corrupted record and close the file successfully after reading only part of the file's content.

In this case, you must set the **LocalFileOTD.Configuration.ResumeReading** node to **False** before calling **finish()**. This setting tells the BatchLocalFile OTD to complete the operation without storing a resume-reading state. You can set up the Collaboration Rule to then send notifications or take other measures, as desired.

Data Stream-adapter Provider

You can use the BatchLocalFile OTD to implement the eWay's data streaming feature. This feature is also available with the FTP and record-processing OTDs. However, the BatchLocalFile OTD is a data stream-adapter provider, while the other two OTDs are only consumers. See **"Streaming Data Between Components" on page 182** for details on how to use the OTD's data streaming feature.

Sequence Numbering

Sequence numbering for the BatchLocalFile OTD operates similar to sequence numbering for the BatchFTP OTD. See **"Sequence Numbering" on page 155** for details.

Generating Multiple Files with Sequence Numbering

When using a Java Collaboration to generate multiple files that have sequence numbering using the BatchLocalFile interface, **reset()** must be called to signify the end of one file and the start of the next.

Handling Type Conversions

This feature in this OTD operates in the same way as type conversion for the BatchFTP OTD. See **"Handling Type Conversions" on page 153** for details.

4.6.3 Recommended Practice

To parse records or construct payloads in your Collaboration, we recommend that you use the record-processing BatchRecord OTD together with the BatchLocalFile OTD. Using the two OTD together provides a number of advantages over just using the BatchFTP OTD.

Example 1: Parsing a Large File

For example, you have set up a Collaboration Rule to parse a large file and submit the records to a database or a JMS IQ Manager. If something goes wrong during the parsing process, the whole file needs to be transmitted again from the FTP server.

In contrast, streaming from a local file system can avoid later FTP transfers of the same file in case of error. This approach has the advantage of allowing you to use data streaming and the Resume Reading feature with large files (see **"Streaming Data Between Components" on page 182** and **"Resume Reading Feature" on page 167**.

Example 2: Slow, Complex Query

Another scenario could be a case where a slow, complex SQL query is used to retrieve a number of records. The Collaboration Rule packs them into a **Payload** node using the record-processing OTD then sends them via FTP to an external system. If the FTP transfer fails, the SQL query must be executed again.

In contrast, if the data payload has been stored locally with the BatchLocalFile OTD, the FTP transfer can be repeated without the need to re-execute the SQL query. In such cases, you can also use data streaming and local-file appending.

In both cases, the use of a data-streaming link can significantly reduce the memory requirements compared to the in-memory data-payload transfer used with the BatchFTP OTD.

OTD Limitations

The BatchLocalFile OTD supports mapped drives and NFS mounted drives. It does not, however, support the mapping of the drives. That is, the drive must already be mapped or mounted. The eWay itself does not perform any mapping or mounting.

The OTD supports Universal Reference Identifiers (URIs) but the scheme must be left off, for example, `\\drive\directory\file_name`.

4.7 BatchRecord OTD

BatchRecord, the Batch eWay's record-processing OTD, allows you to *parse* (extract) *records* from an incoming *payload* (payload data) or to create an outgoing payload consisting of records. Understanding the operation of this OTD and how to use it requires an explanation of some of these terms.

The word *payload* here refers to an in-memory buffer, that is, a sequence of bytes or a stream. Also, *records* in this context are not records in the database sense. Instead, a record simply means a sequence of bytes with a known and simple structure, for example, fixed-length or delimited records.

For example, each of the following record types can be parsed or created by this OTD:

- A large data file that contains a number of SAP IDocs, each with 1024 bytes in length.
- A data file that contains a large number of X12 purchase orders, each terminated by a special sequence of bytes.

The record-processing OTD can handle records in the following formats:

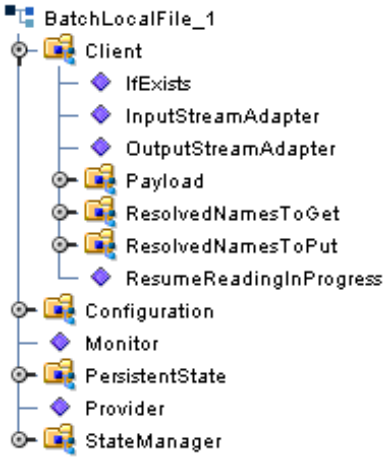
- **Fixed length**: Each record in the payload is exactly the same size.
- **Delimited**: Each record is followed by a specific sequence of bytes, for example, CR,LF.
- **Single**: The entire payload is the record.

When using character delimiters with DBCS data, use single byte character(s) or equivalent hex values with hex values that do not coincide with either byte of the double byte character.

BatchRecord OTD Structure

The BatchRecord OTD contains two top-level nodes, **Client**, **Configuration**, **PersistentState**, and **StateManager** (see Figure 10). Expand these nodes to reveal additional sub-nodes.

Figure 10 BatchRecord OTD Structure



4.7.1 OTD Structure and Operation

Each field node under the **Configuration** node of the OTD, corresponds to one of the eWay's record-processing configuration parameters.

4.7.2 Record-processing OTD Node Functions

The following list explains these primary nodes in the record-processing OTD, including their functions:

- **BatchRecord**: Represents the OTD's root node.
- **Configuration**: Each sub-node within this node corresponds to an eWay configuration parameter and contains the corresponding settings information, except for the **Parse or Create** parameter. See **"BatchRecord Connectivity Map Properties" on page 123** for details.
  - **InputStreamAdapter** and **OutputStreamAdapter**: Allow you to use and control the data-streaming features of the OTD. For details on their operation, see **"Streaming Data Between Components" on page 182**.

Note: *You can transfer data using the Payload node or by using data streaming (InputStreamAdapter and OutputStreamAdapter nodes), but you cannot use both methods in the same OTD.*

Note: *For the record-processing OTD, these configuration nodes are read-only. They are provided only for the purpose of accessing and checking the configuration information at run time.*

- **Record**: A properties node that represents either:
  - The current record just retrieved via the **get()** method, if the call succeeded
  - The current record to be added to the data payload when **put()** is called
- **Payload**: The in-memory buffer containing the data payload byte array you are parsing or creating.

Caution: *It is a good practice to use a byte array in all cases. Failure to do so can cause loss of data.*

  - **put()**: Adds whatever is currently in the **Record** node to the data payload. The method returns **true** if the call is successful.
  - **get()**: Retrieves the next record from the data payload (or stream), and populates the **Record** node with the record retrieved. **get()** returns **true** if the call is successful.
  - **finish()**: Allows you to indicate a successful completion of either a parse or create loop for both **put()** and **get()**.

Note: *Use reset() to indicate any errors and to allow the OTD to clean up any unneeded internal data structures.*

4.7.3 Using the Record-processing OTD

This OTD has the following basic uses:

- **Parsing a payload**: When the payload comes from an external system
- **Creating a payload**: Before sending the payload to an external system

A single instance of the OTD is not designed to be used for both purposes at the same time in the same Collaboration. To enforce this restriction, there is a setting under the eWay's General Settings parameters called **Parse or Create Mode, for which you can select either Parse or Create**.

Using get() and put()

The **get()** and **put()** methods are the heart of the OTD's functionality. If you call either method, the record retrieved or added is assumed to be of the type specified in the eWay configuration, for example, fixed-length or delimited.

The **get()** method can throw an exception, but generally this action only happens when there is a severe failure. One such failure is an attempt to call **get()** before the payload data (or stream if you are streaming) has been set. However, the best practice is to code the Collaboration to check the return value from a **get()** call. A return of **true** means a successful get operation; a **false** means the opposite.

Choosing the Parse or Create Mode

The eWay checks to ensure that the proper calls are made according to your mode setting. For example, calling **put()** in a parse-mode environment would cause the eWay

to throw an exception with an appropriate error message explaining why. Calling **get()** in the create mode would also result in an error.

The eWay requires these restrictions because:

- If you are processing an inbound payload, you are calling **get()** to extract records from the payload (parsing). In this situation it makes little sense to call **put()**. Doing so at this point would alter the payload while you are trying to extract records from it. Calling **put()** would overwrite the payload and destroy the data you are trying to obtain.
- Conversely, when you are creating a payload by calling **put()**, you have no need to extract or parse data at this point. Therefore, you cannot call **get()**.

As a result, you can place the OTD on the source or destination side of a given Collaboration, as desired, and use the OTD for either parsing or creating a payload. However, you cannot parse and create at the same time. Implement your OTD in a Collaboration using the eGate Collaboration Rules Editor.

Creating a Payload

When you want the payload data sent to an external system, you can place the OTD on the outbound side of the Collaboration interfacing with that system. Successive calls to **put()** build up the payload data in the format defined in the eWay configuration.

Once all the records have been added to the payload, you can drag and drop the payload onto the node or nodes that represent the Collaboration's outbound destination. Also, you can set an output stream as the payload's destination (see **Chapter 5** for details on payload streaming).

When you are building a data payload, you must take into account the type and format of the data you are sending. The eWay allows you to use the following formats:

- **Single Record**: This type of payload represents a single record to be sent. Each successive call to **put()** has the effect of growing the payload by the size of the data being put, and the payload is one contiguous stream of bytes.
- **Fixed-size Records**: This type of payload is made up of records, with each being exactly the same size. An attempt to **put()** a record that is not of the size specified causes an exception to be thrown.
- **Delimited Records**: This type of payload is made up of records that have a delimiter at the end. Each record can be a different size. Do not add any delimiters to this data type when it is passed to **put()**. The delimiters are added automatically by the eWay.
- **User Defined**: In this type of payload, the semantics are fully controlled by your own implementation.

Parsing a Payload

To represent payload data inbound from an external system, you map the data to the payload node in the OTD (from the Collaboration Rules Editor). In addition, you can specify an input stream as a source (see **Chapter 5** for details on payload streaming).

Either way, each successive call to **get()** extracts the next record from the payload. The type of record extracted depends on the parameters you set in the eWay's configuration, for example, fixed size or delimited.

You must design the parsing Collaboration with instructions on what to do with each record extracted. Normally, the record can be sent to another Collaboration where a custom OTD describes the record format and carries on further processing.

Fully Consuming a Payload

It is possible to fully consume a payload. That is, after a number of successive calls to **get()**, you can retrieve all the records in the payload. After this point, successive calls to **get()** return the Boolean **false**. You must design the business rules in the subject Collaboration to take this possibility into account.

Using Record Processing with Data Streaming

If you are using the record-processing OTD with data streaming, you must be careful not to overwrite the output files. If the OTD is continually streaming to a BatchLocalFile OTD that uses the same output file name, the OTD can write over files on the output side.

To avoid this problem, you must use either file sequence numbering or change the output file names in the Collaboration Rules. Sequence numbering allows the BatchLocalFile OTD to distinguish individual files by adding a sequence number to them. If you use target file names, post-transfer file names, or both, you can change the name of the output file to a different file name.

For more information on how to use these features, see **"Sequence Numbering" on page 169** and **"Pre/Post File Transfer Commands" on page 165**.

4.8 BatchInbound OTD

The BatchInbound Input (Trigger) File eWay polls for an input file, renames the file with a GUID prefix, and triggers the Business Process or Collaboration.

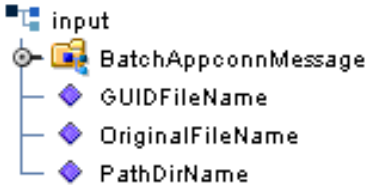
The Batch eWay's **BatchInbound** OTD acts similar to the inbound File eWay, in that it regularly polls an input directory for inbound target files. But unlike the File eWay, when a file with the appropriate name is received by the BatchInbound eWay, the target file is renamed with the following pattern: **<*GUID*>.<*original_filename*>** to ensure that the file is not over-written and is only sent once. A GUID (Globally Unique Identifier) provides a unique, formatted string that represents a 128-bit value.

Note that the BatchInbound OTD does not read the file, but renames the file in such a way that it provides the name of the file that triggers the Business Process or Collaboration

4.8.1 BatchInbound OTD Structure

The BatchInbound OTD contains one top-level node, **BatchAppconnMessage**, with three fields, **PathDirName**, **OriginalFileName**, and **GUIDFileName** (see Figure 11). These nodes provide the external input directory, original file name, and the GUID file name.

Figure 11 BatchInbound OTD Structure



4.9 Using Regular Expressions

A regular expression is a character string in which some characters provide special meaning in regard to matching patterns. This section explains some basic guidelines on how to use regular expressions with the Batch eWay.

Regular Expressions: Overview

Regular expressions allow you to specify patterns for file names and directory names. Regular expressions are used for "get" operations (receiving or source), as opposed to name patterns which are used for "put" operations (sending or destination).

The BatchFTP, Batch FTPOverSSL, BatchSFTP, BatchLocalFile, and BatchInbound OTDs allow you to use regular expressions, for example, if you want to access all files with a specific extension.

Regular expressions operate as follows:

- The directory/file names can be defined as either:
  - Actual file names (everywhere)
  - Name patterns (all names for "put" operations and pre/post transfer names for get operations)
  - Regular expressions (target names for "get" operations)
- The difference between the regular expressions and name patterns is:

- Regular expressions are used to match existing names on the FTP server or the local file system.
- Name patterns are used to create names by replacing the special characters in the pattern.

For more information on name patterns using special characters, see **"Using Name Patterns" on page 178**.

You can specify an extension, for example, .*\.**dat$**. Then, each time the **get()** method is called, the eWay gets the next file with a **.dat** extension. The eWay then retrieves each file into the OTD's **Payload** node and updates the working file-name attribute with the name of the file currently being accessed.

For another example, you can use the file-matching the pattern **data\.00[1-9]** to get the files **data.001**, then **data.002**, and so on. Note that in each case the "." is escaped, which is consistent with regular-expression syntax. It also matches to **xyzdata.001** and **xyz.data.001**, because it does not exclude anything before "data". To make "data" the exact start of the matching pattern you must use **^data\.00[1-9]** or **\A data\.00[1-9]**.

Caution: *The use of regular expressions is an advanced feature and must be implemented carefully. An improperly formed regular expression can cause undesired data or even the loss of data. You must have a clear understanding of regular-expression syntax and construction before attempting to use this feature. It is recommended that you test such configurations thoroughly before moving them to production.*

Entering Regular Expressions

You can enter a regular expression for the FTP or local file name in a variety of ways, for example, .***\.dat$** or **^xyz.*\.dat$**. The first case indicates all files with an extension of **.dat**. The second case indicates all file names with an extension of **.dat** whose names start with **xyz**.

Another example could be **file[0-9]\.dat**. This expression specifies **file0.dat**, **file1.dat**, **file2.dat**, and so on, through **file9.dat**. This will also match xyz.file0.dat, xyz.file1.dat, and so on. This type of expression will not exclude anything in front of "file". To exclude any characters before "file" (to make "file" the exact beginning) use **^file[0-9].dat** or **\Afile[0-9].dat**.

These types of regular expression patterns can be used for a get operation.

Regular Expressions and the eWay

Note that the eWay provides a **File Name Is Pattern** or **Directory Name Is Pattern** configuration parameter after every property that allows a regular expression as an option. This feature allows you to specify that the pattern entered is a regular expression or just a static text entry to be interpreted literally.

Important: *Regular expressions will resolve even with a partial match to the file name. The resolution process searches for the file name contents rather than the file name.*

4.9.1 Rules for Directory Regular Expressions

There are special considerations you must be aware of when you use regular expressions for directory names. This section describes these restrictions and provides some examples.

Restrictions for Using Regular Expressions as Directory Names

The following restrictions apply when using regular expressions as directory names:

- The directory root, the drive name, and directory separators must be expressed exclusively. That is, do not express any of these elements as a regular expression. Only folder names are expected to appear as regular expressions.
- A regular expression must not span over the directory separators. If you use a regular expression between two directory separators, it must be one whole expression.
- Escape all directory separators in a directory pattern if the separator conflicts with a regular expression special character (that is, **‘ * [ ] ( ) | + { } : . ^ $? \"**). The back slash (\) is the special character used to escape other special characters in regular expressions. For Windows platforms, the directory separator is the back slash, so it must be escaped as `\\`.
- For the Windows Universal Naming Convention (UNC), the directory root (including the computer name and the shared root folder name) must be expressed exclusively. That is, do not express the computer name and shared root folder as a regular expression.
- Directory separators are platform dependent, for example:
  - For **Windows** platforms, the directory path follows this pattern:

    ```
    drive:\\regexp1\\regexp2\\regexp3 ...
    ```

    or for **Windows UNC** notation, the directory path follows this pattern:

    ```
    \\\\machineName\\shared_folder\\regexp1\\regexp2\\regexp3 ...
    ```

  - For **UNIX** platforms, including mounted directories, the directory path follows this pattern:

    ```
    /regexp1/regexp2/regexp3 ...
    ```

Regular Expression Directory Name Examples

The following are several examples of regular expression directory name usage:

- Windows:

  ```
  c:\\eGate$\\^client\\collab\D\\ ...
  ```

  The expression **\D** indicates any non-digit character.

  ```
  d:\\a.b\\c.d\\e.f\\g.h\\[0-9]\\ ...
  ```

  The symbol “.” means any character

- Windows UNC notation:

```
\\\\My_Machine\\public\\xyz$\\^abc
```

The prefix for Windows UNC notation is \\. After escaping, it becomes \\\\.

- UNIX:

```
/abc\d/def/ghi/ ...
```

The expression **\d** means any digit character.

```
/^PRE[0-9]{5}\.dat$/ ...
```

This expression means to begin with **PRE** followed by a five-digit number and use a **.dat** extension. The symbol **\.** means to interpret the real character (a period) instead of any character. Therefore, **PRE12345.dat** does match, but **PRE123456dat** does not.

4.10 Using Name Patterns

The Batch eWay allows you to use a Name Pattern, that is, special characters that symbolize often-used information as *short-hand*. You can use these character combinations to specify place holders for this specific information. Using these characters, you can quickly convey date/time, number, and file-name information.

The BatchFTP, Batch FTPOverSSL, BatchSFTP, BatchLocalFile, and BatchInbound OTDs allow you to use special characters or specify a name pattern. A name pattern allows you to specify patterns for file names and directory names. Name patterns are used for "put" operations (sending or destination) , as opposed to regular expressions which are used for "get" operations (receiving or source).

Special characters are utilities the eWays use for file-name pattern. The general rules for their use are:

- Use **%** to indicate the special character that needs to be expanded.
- Use **%%** to indicate the escaped character %; for example, **abc%%d** means **abc%d**, and the **%d** is not expanded again.

For example, for a put operation, a pattern such as **file%#.dat** can be used. This pattern uses the sequence number setting in the configuration, and each put creates successive files named **file1.dat**, **file2.dat**, and so on.

For information on regular expressions, see **"Using Regular Expressions" on page 175**.

4.10.1 Types of Name Patterns

The eWay provides the following types of name patterns:

- **Date/Time stamp**: Uses the format **%[GyMdhHmsSEDFwWakKz]**, for example, **abc%y%y%y%y** means **abc2001** (see **Table 60 on page 179** for more information).
- **Sequence number**: Uses the format **%#, %5#**, for example, **abc%#** means **abc1**, **abc2**, **abc3**, and so on; for another example, **abc%5#** (zero-padded) means **abc00001**, **abc00002**, **abc00003**, ..., **abc00010**, ..., **abc00100**, and so on.

- **Working-file name**: Uses the format **%f**; normally, it is used for pre- or post-file-transfer commands (see **"Pre/Post File Transfer Commands" on page 165**), for example, **%f.abc** means **working_filename.abc**.

The sequence of expansion operates in the reverse order of the previous list, that is, first the file name is expanded, then the sequence number, and finally the time stamp.

Some additional examples of name pattern:

- abc.%y%y%y%y%M%M%d%d.%h%h%m%m%s%s%S%S%S means abc.20011112.162532678
- abc%#.def%# means abc2.def3
- %f.%# means xxxxx.4, xxxxx.5, ... (Where xxxxx is the working-file name)

4.10.2 Resolving Names

Typically, the pre/post names with name patterns or regular expressions are resolved during **get()** and **put()** method calls. But sometimes, in using Collaboration Rules, the eWay has to get the resolved names before the actual **get()** or **put()** call.

In such cases, you can get the resolved names in this way through the **ResolvedNamesForGet** and **ResolvedNamesForPut** nodes in the BatchFTP OTD, for example:

getResolvedNamesForPut().getTargetFileName()

The previous code yields **file1** based on the pattern **file%#**. In this usage, the OTD nodes can be used to make the desired method call. See **"BatchFTP OTD Node Functions" on page 152** for more information on BatchFTP OTD nodes.

4.10.3 Date/time Format Syntax

The eWay uses the Java simple default date and time format syntax (U.S. locale). To specify these formats for name pattern, you must use a time pattern string.

Note: *The eWay uses the Java standard for date/time stamps from the Java class java.text.SimpleDateFormat. Some of these formats can differ from the list given here, depending on the Java SDK version you are using.*

In these patterns, all ASCII letters are reserved as pattern letters. See Table 60 for a complete list.

Table 60 Time Pattern Strings and Meanings

Symbol	Meaning	Presentation	Example
%G	Era designator	Text	AD
%y	Year	Number	1996
%M	Month in year	Text and number	July & 07
%d	Day in month	Number	10
%h	Hour in a.m./p.m. (1 through12)	Number	12

Table 60 Time Pattern Strings and Meanings (Continued)

Symbol	Meaning	Presentation	Example
%H	Hour in day (0 through 23)	Number	0
%m	Minute in hour	Number	30
%s	Second in minute	Number	55
%S	Millisecond	Number	978
%E	Day in week	Text	Tuesday
%D	Day in year	Number	189
%F	Day of week in month	Number	2 (second Wednesday in July)
%w	Week in year	Number	27
%W	Week in month	Number	2
%a	Marker for a.m./p.m.	Text	PM
%k	Hour in day (1 through 24)	Number	24
%K	Hour in a.m./p.m. (0 through 1)	Number	0
%z	Time zone	Text	Pacific Standard Time

The general rules for date/time formats are:

- **Text**: The count of pattern letters determines the format as follows:
  - For four or more pattern letters, use the full form.
  - For fewer than four, use the short or abbreviated form if one exists.
- **Number**: The minimum number of digits as follows:
  - Shorter numbers are zero-padded to this amount.
  - The year is handled differently; that is, if the count of "y" is two, the year is truncated to two digits.
- **Text and number**: For three or more pattern letters, use text; otherwise use a number.
- **Quotes and delimiters**: Use these symbols as follows:
  - Enclose literal text you want rendered within single quotes.
  - Use double quotes to mean single quotes.
  - Use commas for delimiters.

Table 61 shows some examples using the U.S. locale.

Table 61 U.S. Locale Date/time Patterns

Format Pattern	Result
yyyy.MM.dd, G, 'at' hh:mm:ss, z	1996.07.10 AD at 15:08:56 PDT
E, M, dd, ''yy	Wednesday, July 10, '96
h:mm, a	12:08 PM
h, 'o''clock' a, z	12 o’clock PM., Pacific Daylight Time
K:mm a, z	0:00 p.m., PST
yyyyy.M.dd, G, hh:mm, a	1996.July.10 AD 12:08 PM

Chapter 5

Additional Batch eWay Features

This chapter explains additional features of the Batch eWay, including Data Streaming, SOCKS, SSH Tunneling, and Ensuring Secure FTP Data Transfers.

What's in This Chapter

5.1 Streaming Data Between Components

Components in the Batch eWay implement a feature for *data streaming*. This chapter explains data streaming, how it works, and how to use it with the eWay and eGate Integrator.

5.1.1 Introduction to Data Streaming

Data streaming provides a means for interconnecting any two components of the eWay by means of a *data stream channel*. This channel provides an alternate way of transferring the data between the Batch eWay components. Streaming is available between BatchLocalFile and BatchFTP, or BatchLocalFile and BatchRecord.

Each OTD component in the eWay has a **Payload** node. This node represents the in-memory data and is used when the data is known to be relatively small in size or has already been loaded into memory. The node can represent, for example, the buffer in the record-processing OTD, as it is being built or parsed, or the contents of a file read into memory.

Instead of moving the data all at once between components in eGate's memory, you can use a *data-stream channel* to provide for streaming the data between them a little at a time, outside of eGate.

Data streaming was designed primarily to handle large files, but you can use it for smaller data sizes as well.

Use the eGate Enterprise Designer's Collaboration Rules Editor to set up data-streaming operations. The rest of this section explains the data streaming feature and how to set it up.

Note: *Payload-based and streaming-based transfers are mutually exclusive. You can use one or the other but not both for the same data.*

5.1.2 Overcoming Large-file Limitations

The primary advantage of using data streaming is that it helps to overcome the limitations of dealing with large files. For example, if you have a 1-gigabyte file that contains a large number of records, you need a large amount of resources to load the payload into memory just to parse it.

Streaming allows you to read from the file, little by little, using a data-streaming mechanism. This way, you do not need to load the file into the eGate system's memory. Using streaming is not as fast as using in-memory operations, but it is far less resource-intensive.

5.1.3 Using Data Streaming

Each data-streaming transfer involves two OTDs in a Collaboration as follows:

- One provides the *stream adapter*.
- The other consumes the stream adapter to perform the data transfer.

Caution: *Implementing InputStream must support skip() with negative numbers as an argument.*

This section explains how the two data-streaming OTDs operate to effect the transfer of data.

Data-streaming Operation

Each of the OTDs in the Batch eWay exposes stream adapter nodes that enable any OTD to participate in data-streaming transfers. The nodes are named **InputStreamAdapter** and **OutputStreamAdapter**. You can associate the stream adapters by using the drag-and-drop features of the eGate Enterprise Designer.

The **InputStreamAdapter** (highlighted) and **OutputStreamAdapter** nodes in the OTD are used for data streaming. This feature operates as follows:

- **Stream-adapter consumers**: The FTP and the record-processing OTDs can only *consume* stream adapters. Therefore, their stream-adapter nodes are *write-only*. Their node values can be *set* (modified).
- **Stream-adapter provider**: The local file OTD can only *provide* stream adapters, so its stream-adapter nodes are *read-only*. Its node value can only be *retrieved*.

The local file OTD is always the stream provider, and the FTP and record-processing OTDs are the consumers.

Note: *For an explanation of the eWay's different types of OTDs, see* **"Understanding Batch eWay OTDs" on page 149**.

Data Streaming Versus Payload Data Transfer

Use of the **InputStreamAdapter** and **OutputStreamAdapter** nodes is an alternative to using the **Payload** node as follows:

- Use these stream adapter nodes to transfer data if you want data streaming.
- Use the **Payload** node for a data transfer *without* data streaming (payload data transfer).

All operations that, in payload data transfer, *read* from the **Payload** node require the **InputStreamAdapter** node when you are setting up data streaming. Using the same logic, all operations that, in payload data transfer, *write* to the **Payload** node require **OutputStreamAdapter** node for data streaming.

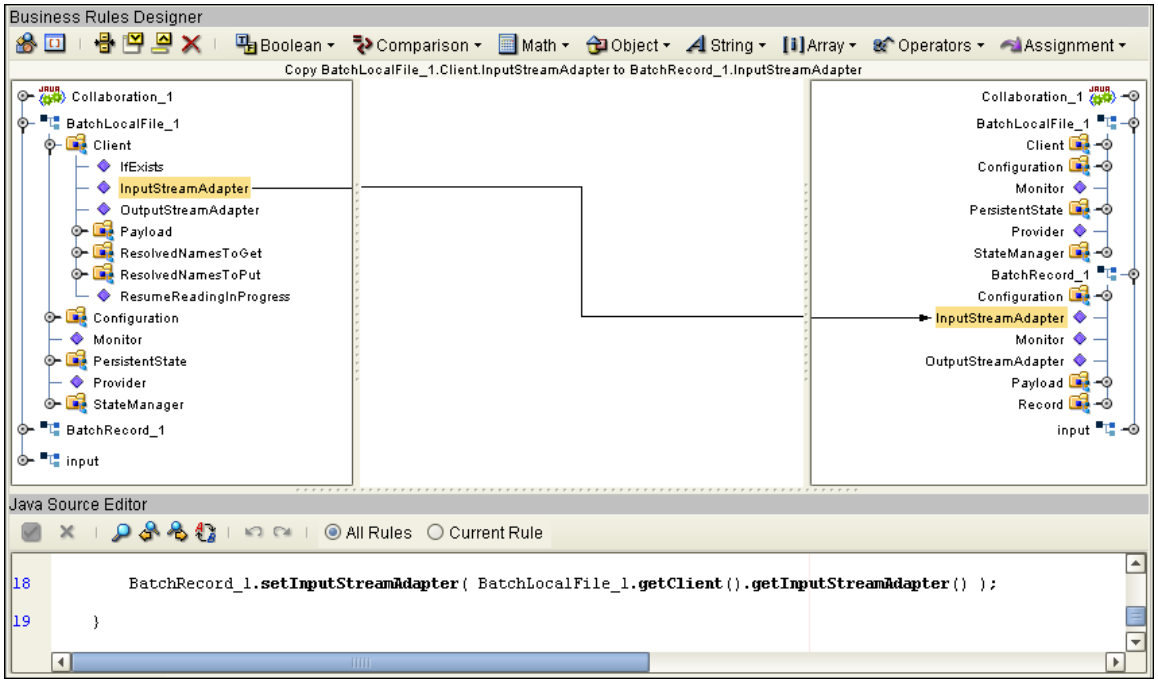
Do *not* confuse the stream adapter nodes with the **get()** and **put()** methods on the OTDs. For example, the BatchFTP OTD's client interface **get()** method *writes* to the **Payload** node during a payload transfer, so it requires an **OutputStreamAdapter** node to write to for data streaming. In contrast, the record-processing OTD's **get()** method *reads* from the **Payload** node during a payload transfer, so for data streaming, **get()** requires an inputStreamAdapter node to read from.

Data Streaming Scenarios

The four typical data-streaming scenarios are:

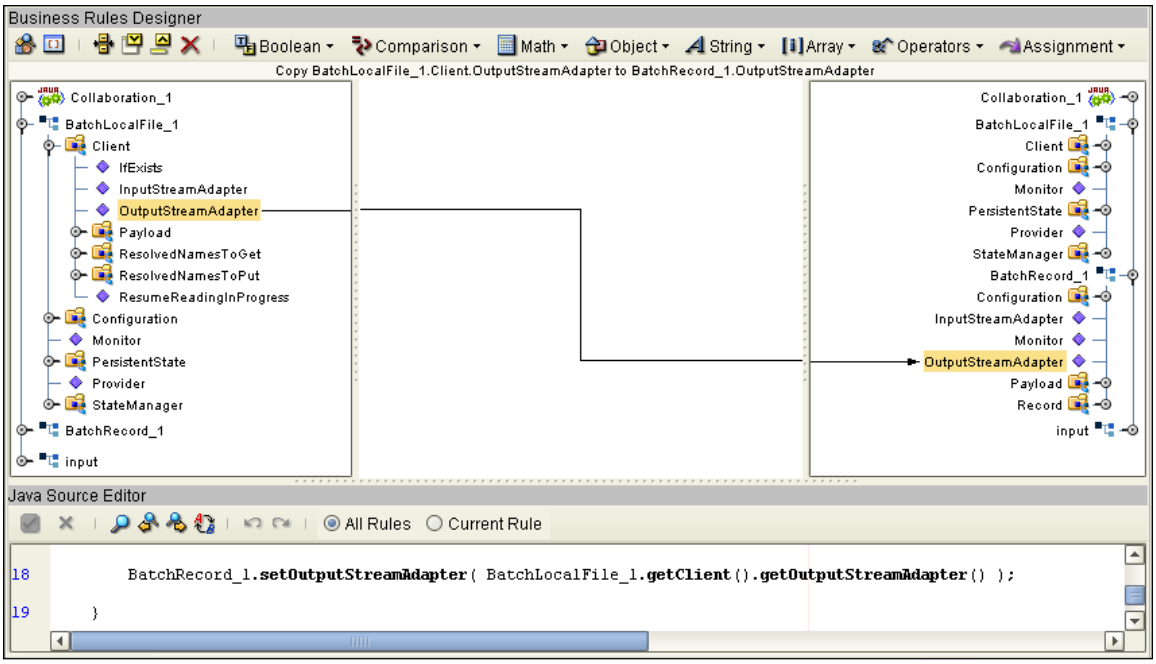
- Transfer data from a local file system (BatchLocalFile) to a record-processing (BatchRecord) setup. This scenario uses the **InputStreamAdapter** OTD node (see Figure 12).

Figure 12 BatchLocalFile to BatchRecord InputStreamAdapter



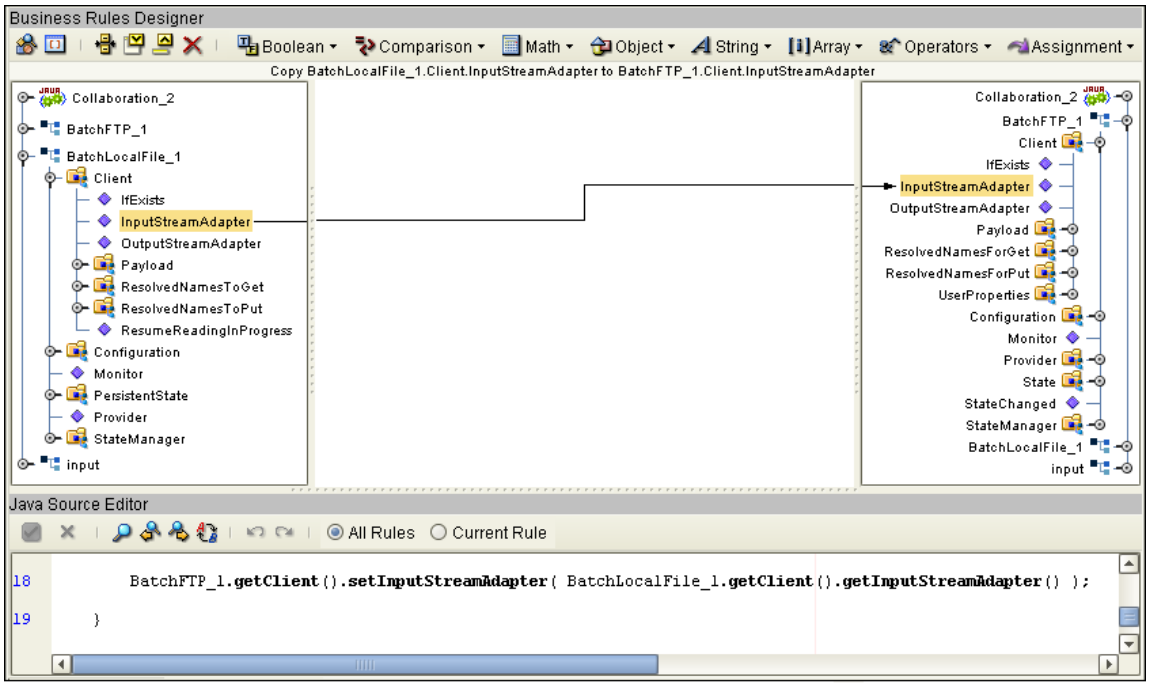
- Transfer data from a record-processing (BatchRecord) setup to a local file system (BatchLocalFile) using the **OutputStreamAdapter** OTD node (see Figure 13).

Figure 13 BatchLocalFile to BatchRecord OutputStreamAdapte



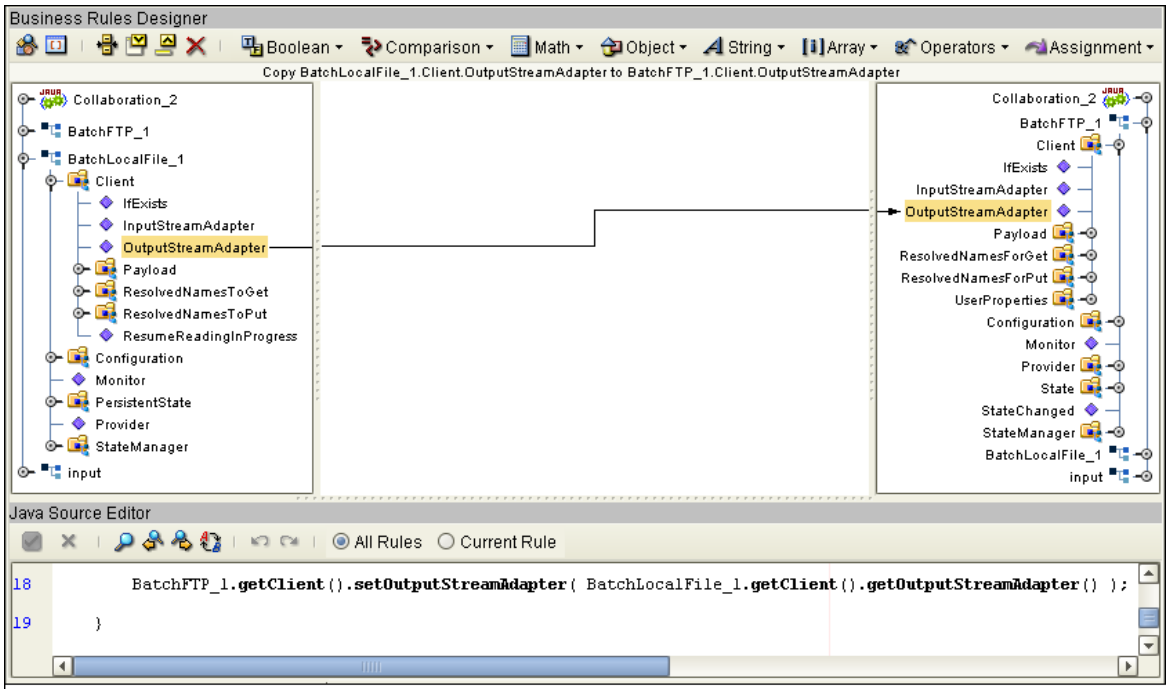
- Transfer data from a local file system (BatchLocalFile) to a remote FTP (BatchFTP) setup. This scenario uses the **InputStreamAdapter** OTD node (see Figure 14).

Figure 14 BatchLocalFile to BatchFTP InputStreamAdapter



- Transfer data from a remote FTP server (BatchFTP) to a local file system (BatchLocalFile) using the **OutputStreamAdapter** OTD node (see Figure 15).

Figure 15 BatchLocalFile to BatchFTP OutputStreamAdapter



Consuming-stream Adapters

This section explains how to use consuming-stream adapters.

To obtain a stream

- Use the **requestXXStream()** method to obtain the corresponding **XX** stream.

To use a stream

- Perform the transfer using the methods provided by the stream.

To dispose of a stream

- Release any references to the stream.
- Release the stream (**XX**) using the **releaseXXStream()** method. Some of the OTDs support post-transfer commands. The **Success** parameter indicates whether these commands are executed. Do not close the stream.

5.1.4 Stream-adapter Interfaces

This section provides the Batch eWay's OTD stream-adapter Java interfaces. This information is only for advanced users familiar with Java programming, who want to provide custom OTD implementations for stream-adapter consumers or providers.

Inbound Transfers

The following Java programming-language interface provides support for inbound transfers from an external system:

```
public interface com.stc.eways.common.eway.streaming.InputStreamAdapt
er {
public java.io.InputStream requestInputStream() throws
StreamingException;
public void releaseInputStream(boolean success) throwsStreamingExcept
ion;
}
```

Outbound Transfers

The following Java interface provides support for outbound transfers to an external system:

```
public interface com.stc.eways.common.eway.streaming.OutputStreamAdap
ter {
public java.io.OutputStream requestOutputStream() throws
StreamingException;
public void releaseOutputStream(boolean success) throws
StreamingException;
}
```

5.2 SOCKS FTP Support

This section explains the SOCKS FTP features available for the Batch eWay.

5.2.1 SOCKS

SOCKS is an Internet Engineering Task Force (IETF) -approved standard (RFC 1928) generic, proxy protocol for TCP/IP-based network applications. This simple protocol supports a flexible framework for developing secure communications. SOCKS accomplish this by easily integrating other security technologies.

Note: *The eWay only supports SOCKS protocols that conform to this IETF standard.*

There are two versions of the SOCKS protocol.

- **SOCKSv4** (version 4), that provides the following functions:
  - requests connections
  - Setup proxy clients
  - transmits application data
- **SOCKSv5** (version 5) that includes all the functionality of version 4 and also provides authentication

Both the SOCKSv4 and SOCKSv5 protocols are supported by the Batch eWay. To enable support, the following properties must be specified in the Batch eWay Properties Sheet:

- SOCKS server name
- SOCKS server port number
- User name
- Encrypted password

Details of these configuration parameters are provided under **"SOCKS Configuration Properties" on page 189**.

Note: *In the Collaboration Rules, make sure you set the SOCKS version number to 4, 5, or -1 (unknown). Do not set this value to any other number.*

SOCKS: Overview

SOCKS embodies two components, the **SOCKS Server** (implemented at the application layer), and the **SOCKS Client** (implemented between the application and transport layers).

In essence, the purpose of the SOCKS protocol is to allow a host on one side of a SOCKS Server to interact with a host on the other side of the Server, subject to authentication, without passing IP packets directly between the two.

SOCKS Proxy Server

The SOCKS proxy server connects to the application server on behalf of the application client and relays data between the client and an application server. From the application server's perspective, the SOCKS proxy is the client.

SOCKS and the Batch eWay

Negotiation Methods

The BatchFTP eWay supports the following methods used to define the negotiation phase of authentication between the Socks Client and Server:

- No-authentication (no authentication required)
- User/password (user name and password)

SOCKS Configuration Properties

The Batch eWay contains a number of properties used to configure SOCKS with the BatchFTP eWay. These properties are configured using the BatchFTP Properties Sheet accessed from the Connectivity Map and the Environment Explorer.

- **Socks Enabled:** Specifies whether the FTP command connection goes through a SOCKS server. A value of No indicates that the eWay is not connecting to a SOCKS server. In this case, all other parameters under the SOCKS section are ignored.
- **Socks Host Name**: Specifies the SOCKS host name. When you are communicating with a SOCKS server, enter the SOCKS server name in this parameter.
- **Socks Server Port**: Specifies the port number of the SOCKS server.
- **Socks Version**: Specifies the SOCKS server version. A value of **4** or **5** for SOCKSv4 or SOCKSv5 provides the best performance, but the default value **Unknown can if the version is in question.**
- **Socks User Name**: Specifies the user name that matches the associated password used for authentication with a SOCKS5 server. This parameter is applied when user/password negotiation method is used.
- **Socks Password**: Specifies the password to use along with the user name for authentication with a SOCKS5 server. This parameter is applied when user/password negotiation method is used.

For information on the BatchFTP configuration properties, see **"BatchFTP eWay Connectivity Map Properties" on page 32**, and **"BatchFTP eWay Environment Properties" on page 52**.

5.3 SSH Tunneling Support

This section explains the Batch eWay's Secure Shell (SSH) tunneling features. SSH tunneling is also called SSH port forwarding.

The Batch eWay encrypts the command channel of FTP utilizing SSH. To encrypt data, you can encrypt a file prior to sending it, using your preferred method or that of the receiver. The received file can then be decrypted by the recipient. If Secure FTP (FTP over SSH or FTP over SSL) is required, use the Secure FTP OTDs ((**BatchFTPOverSSL**, **BatchSFTP**, and **BatchSCP**).

SSH Tunneling: Overview

Developed by SSH Communications Security Ltd., Secure Shell (SSH) is a program that allows a computer to log onto another computer over a network to move files over the network and execute commands. SSH is intended as a replacement for **rlogin**, **rsh**, **rcp**, and **rdist**.

SSH provides strong authentication and secure communications over non-secure channels. SSH protects a network from attacks such as IP and DNS spoofing, IP source routing, and interception of plaintext passwords and authentication data. If an attacker manages to take over a network, he can only force SSH to disconnect. The content and the connection are secure when encryption is enabled.

When you are using the SSH **slogin** (instead of **rlogin**), the entire logged-on session, including the transmission of the password, is encrypted. As a result, it is almost impossible for an outsider to collect passwords.

Note: *For improved security, the number of times the eWay can log on during a single session is limited because, during a disconnect, the SSH tunnel is not closed. This method of operation allows you to establish another connection without logging on.*

For more information on SSH and how to use it, see the following Web site:

http://www.openssh.com

Additional Software Requirements

The eWay makes use of additional software applications. The eWay also supports either of the following applications for SSH tunneling:

- **OpenSSH**: an encryption and authentication tool for UNIX. For more information go to:

  http://www.openssh.org

- **Plink.exe**: Plink is a Win32-only command-line interface to the PuTTY Telnet/SSH client. For more information visit:

  http://www.chiark.greenend.org.uk/~sgtatham/putty

In either case, the you are responsible for downloading, installing, and properly configuring the necessary software. You must refer to the appropriate software provider for support and documentation.

SSH Tunneling and the Batch eWay

To use SSH tunneling to provide for secure logon IDs and passwords, the BatchFTP eWay uses the additional SSH-tunneling software (see **"Additional Software Requirements" on page 190**).

Enabling SSH Tunneling

To enable SSH tunneling, select **Yes** under the **SSH Tunneling Enabled** parameter in the eWay Connection configuration (see **"SSH Tunneling Configuration Parameters" on page 192**). You can use the SSH-tunneling software in either of the following ways:

- By using an existing SSH channel where a secure connection has already been established
- By internally launching an SSH process for the eWay's use

Using an Existing Channel

To use an existing channel, select **Yes** under the **SSH Channel Established** parameter in the configuration. The eWay then operates under the assumption that you have already established the SSH channel using the additional software. Once you set this parameter to **Yes**, the eWay automatically uses that channel.

Using an Internal Channel

If you choose **No**, under the **SSH Channel Established** parameter, the eWay launches a process within eGate to establish a channel. In this case, you must specify, under the **SSH Command Line** parameter, a full and correct command-line statement for your SSH-tunneling application and environment.

Note: *You can obtain this information from the SSH-tunneling application's configuration. See the application's documentation for details.*

You must enter a correct and complete command-line statement. That is, all necessary command line parameters must be provided so that the SSH-tunneling software can run correctly without requiring further interaction.

Check the accuracy of this information by executing the command line from the shell. If the software prompts for more information, add the required information to the command line and try again. Continue this process until the software starts and operates properly without additional action.

Note: *You may need to launch the application at least once from the shell before using it in the eWay. This requirement depends on the SSH-tunneling application and platform. Some applications prompt for trust-related information on the first attempt, to connect to a remote host.*

Port-forwarding Configuration

Through SSH tunneling, the FTP command connection is protected. This mechanism is based on an existing SSH port-forwarding configuration. You must configure SSH port forwarding on the *SSH listen host* before you configure the supporting eWay Connection.

For example, on the eGate client host **localhost**, you can issue a command, such as:

```
ssh -L 4567:atlas:21 -o BatchMode=yes atlas
```

Under the eWay's configuration for the previous example, you must specify:

- **localhost** for the parameter **SSH Listen Host**
- **4567** for the parameter **SSH Listen Port**

In this case, the eWay connects to the FTP server **atlas:21** through an SSH tunnel.

SSH Tunneling Configuration Parameters

You must set the following SSH tunneling parameters to configure the eWay Connection:

- **SSH Tunneling Enabled**: Specifies whether the FTP command connection is secured through an SSH tunnel:
  - **No:** indicates that all other parameters in this section are ignored.
- **SSH Channel Established**: Specifies whether the eWay needs to launch an SSH subprocess:
  - **No**: indicates that there is no existing SSH channel for an FTP transfer.
  - **Yes**: indicates that an SSH channel has been established, so it is not necessary for the eWay to spawn an SSH subprocess. If you select **Yes**, the following parameters are required:
    - **SSH Listen Host**
    - **SSH Listen Port**
- **SSH Command Line**: Specifies the command line used to establish an SSH channel. This parameter is required only when you set the **SSH Channel Established** parameter to **No**.

  The command-line syntax can be different, depending on the specific SSH client implementation. See your SSH-tunneling support software user's guides for details.
- **SSH Listen Host**: Specifies the host name where the SSH support software runs, as well as the host it listens to.

  This parameter is required only when you set the **SSH Channel Established** parameter to **Yes**. If you choose **No**, the **Listen Host** is always **localhost** because the SSH support software is always started from the local host.
- **SSH Listen Port**: Specifies the port number that the SSH-tunneling support software uses to check for incoming connections. This port number can be any unused port number on the SSH listen host.
- **SSH User Name**: Specifies an SSH user name. This parameter can be required when the setting for the **SSH Channel Established** parameter is **No**.
- **SSH Password**: Specifies an SSH password corresponding to the user name entered under **SSH User Name**. This parameter can be required only when the setting for the **SSH Channel Established** parameter is **No**. For more information, see **SSH User Name**.

For more information, see **"SSH Tunneling Configuration Parameters" on page 192**.

Chapter 6

Using the Batch eWay with eInsight

This chapter describes how to use the Batch eWay with Java Integration Suite's eInsight Business Process Manager and its engine's Web Services interface.

Note: *You must have the* **eInsight.sar** *file installed to use the Web Services interface.*

What's in This Chapter

6.1 eInsight Engine and eGate Components

You can deploy an eGate component as an Activity in an eInsight Business Process. Once you associate the desired component with an Activity, eInsight invokes it using a Web Services interface. eGate components that can interface with eInsight in this include the following:

- Java Messaging Service (JMS)
- Object Type Definitions (OTDs)
- eWays
- Collaborations

Using the eGate Enterprise Designer and eInsight, you can add an Activity to a Business Process, then associate that Activity with an eGate component, for example, an eWay. Then, when eInsight runs the Business Process, it automatically invokes that component via its Web Services interface.

See the *Sun SeeBeyond eInsight™ Business Process Manager User's Guide* for details.

6.2 Batch eWay With eInsight

You can associate an eInsight Business Process Activity with Product Name Variable during the system design phase. To make this association, select the desired operator

under the eWay in the Enterprise Explorer and drag it onto the eInsight Business Process canvas.

For Business Process operations, the Batch eWay has the following operators available under the for Batch configuration nodes:

BatchFTP

- **get** (Invokes the **get** operation using the current property values in the configuration.)
- **getIfExists** (Same as get, but if no file is found that matches the name pattern given in "Target Directory Name" and "Target File Name" the return is silent.
- **put (**Invokes the **put** operation using the current property values in the configuration.

BatchFTPOverSSL

- **get** (Invokes the **get** operation using the current property values in the configuration.)
- **put (**Invokes the **put** operation using the current property values in the configuration.
- **GET** (Functions in the same way as "**get**" but applies pre and post transfer operations.)
- **PUT** (Functions in the same way as "**put**" but applies pre and post transfer operations.)

BatchInbound

- **receive**

BatchSFTP

- **get** (Invokes the **get** operation using the current property values in the configuration.)
- **put (**Invokes the **put** operation using the current property values in the configuration.
- **GET** (Functions in the same way as "**get**" but applies pre and post transfer operations.)
- **PUT** (Functions in the same way as "**put**" but applies pre and post transfer operations.)

BatchSCP

- **get** (Invokes the **get** operation using the current property values in the configuration.)
- **put (**Invokes the **put** operation using the current property values in the configuration.

BatchLocalFile

- **read**
- **readIfExists**

- **write**

BatchInbound

- **receive**

BatchRecord

Not Applicable

The operator automatically changes to an Activity with an icon identifying the component that is the basis for the Activity.

At run time, the eInsight engine invokes each step in the order defined in the Business Process. Using the engine's Web Services interface, the Activity invokes the Batch eWay.

6.3 Considerations

The following items must be considered when implementing a Batch eWay Project:

- When using FTP with an **AS400 UNIX (USF)** system, the following FTP configuration settings are required:
  - FTP - Use PASV: **No** (see **"Use PASV" on page 38**)
  - FTP Raw Commands - Pre Transfer Raw Commands: **site namefmt 1** (see **"Pre Transfer Raw Commands" on page 40**)

6.4 Importing a Sample Project

Sample eWay Projects are included as part of the installation package. To import a sample eWay Project to the Enterprise Designer do the following:

1. The sample files are uploaded with the eWay's documentation SAR file and downloaded from the Sun Java Composite Application Platform Suite Installer's Documentation tab. The **Batch_eWay_Sample.zip** file contains the various sample Project ZIP files. Extract the samples to a local file.
2. Save all unsaved work before importing a Project.
3. From the Enterprise Designer's Project Explorer pane, right-click the Repository and select **Import** from the shortcut menu. The **Import Manager** appears.
4. Browse to the directory that contains the sample Project zip file. Select the sample file (for example, **prjBatch_Inbound_LocalFileIn_FTPOut.zip**) and click **Import**. After the sample Project is successfully imported, click **Close**.
5. Before an imported sample Project can be run you must do the following:
   - Create an Environment (see **"Creating an Environment" on page 202**)

- Configure the eWays for your specific system (see **"Configuring the eWays Properties" on page 203**)
- Create a Deployment Profile (see **"Creating the Deployment Profile" on page 206**)
- Create and start a domain (see **"Creating and Starting the Domain" on page 207**)
- Build and deploy the Project (see **"Building and Deploying the Project" on page 208**)

6.5 The Batch eWay eInsight Sample Project

The **prjBatch_Inbound_LocalFileIn_FTPOut** sample Project demonstrates how eInsight Business Processes are used with the Batch eWay.

The **prjBatch_Inbound_LocalFileIn_FTPOut** Project demonstrates the following:

- The BatchInbound eWay subscribes to a local directory and periodically checks for a specific file. If the file is present, it renames the file with a GUID, and triggers the Business Process.
- The BatchLocalFile gets the GUID file name from the BatchInbound eWay, gets the associated file from a local directory and writes the file to the BatchFTP payload.
- The BatchFTP eWay publishes the file to an external directory.

Sample data files

Sample data files for the Batch eWay Projects are included with the samples.

6.6 Creating the prjBatch_Inbound_LocalFileIn_FTPOut Project

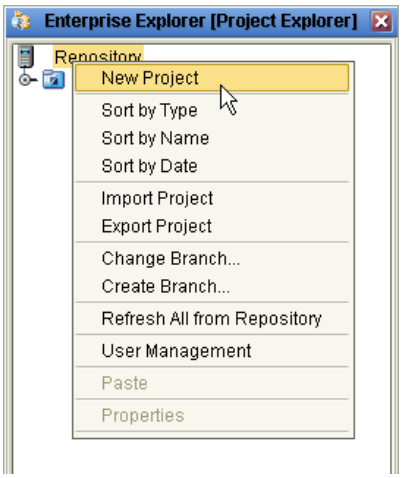
The following pages provide step by step directions for manually creating the **prjBatch_Inbound_LocalFileIn_FTPOut** Project.

6.6.1 Create a Project

The first step is to create a new Project in the Enterprise Designer.

1. Start the Enterprise Designer.
2. From the Enterprise Explorer's Project Explorer tab, right-click the Repository and select **New Project** (see **Figure 16 on page 197**). A new Project (**Project1**) appears on the Project Explorer tree.

Figure 16 Project Explorer - New Project



3 Double-click Project1 and rename the Project as **prjBatch_Inbound_LocalFileIn_FTPOut**.

6.6.2 Creating the bpBatch_Inbound_LocalFileIn_FTPOut Business Process

Creating the Business Process Flow

1 From the Enterprise Designer's Project Explorer tree, right-click **prjBatch_Inbound_LocalFileIn_FTPOut**, and select **New** > **Business Process** from the shortcut menu. The eInsight Business Process Designer appears and **BusinessProcess1** is added to the Project Explorer tree.

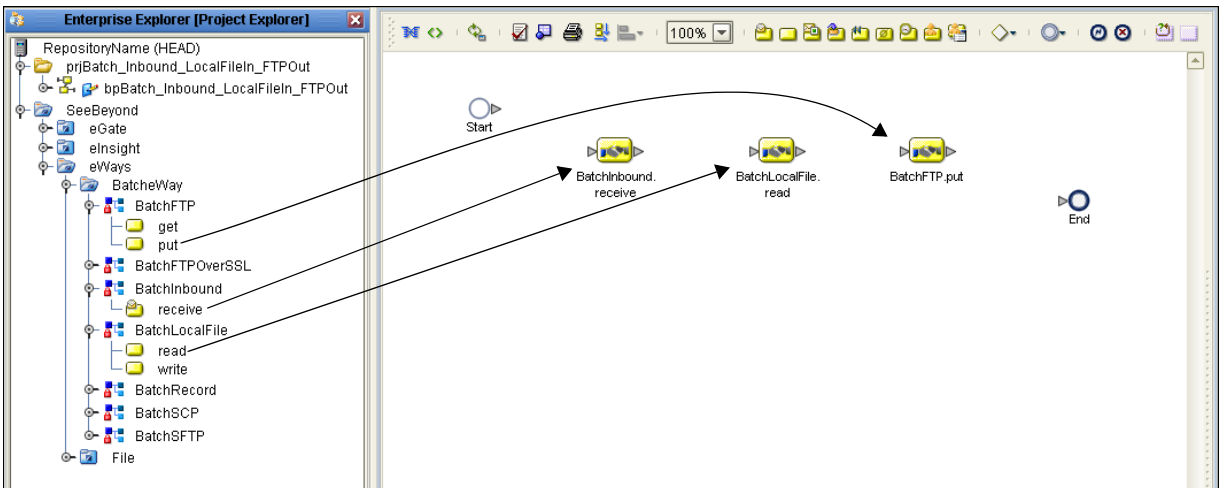
2 Rename the Business Process as **bpBatch_Inbound_LocalFileIn_FTPOut**.

3 From the Project Explorer tree, expand the **Sun SeeBeyond > eWays > BatcheWay**, and expand the **BatchFTP**, **BatchInbound**, and **BatchLocalFile** nodes to expose the available Business Process elements.

4 Populate the eInsight Business Process Designer's modeling canvas with the following elements from the Project Explorer tree, as displayed in Figure 17:

- receive, under **Sun** SeeBeyond > eWays > BatcheWay > BatchInbound
- read, under **Sun** SeeBeyond > eWays > BatcheWay > BatchLocalFile
- put, under **Sun** SeeBeyond > eWays > BatcheWay > BatchFTP

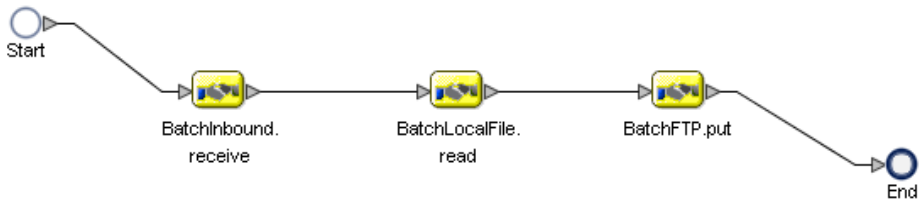
Figure 17 eInsight Business Process Designer - Populate the Canvas



5 Link the modeling elements by clicking on the element's connector and dragging your cursor to the next element's connector, making the following links as displayed in Figure 18.

- **Start** > **BatchInbound.receive**
- **BatchInbound.receive** > **BatchLocalFile.read**
- **BatchLocalFile.read** > **BatchFTP.put**
- **BatchFTP.put** > **End**

Figure 18 eInsight Business Process Designer - Link the Modeling Elements

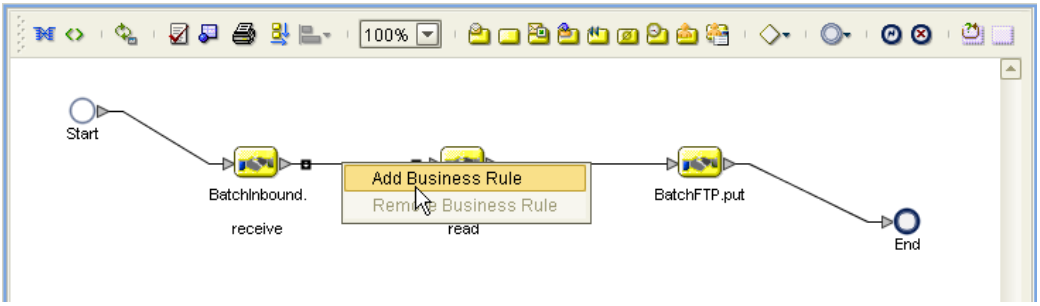


Configuring the Modeling Elements

Business Rules, created between the Business Process elements, define the relationship between the input and output Attributes of the elements.

1 Right-click the link between **BatchInbound.receive** and **BatchLocalFile.read** Activities and select **Add Business Rule** from the shortcut menu as displayed in **Figure 19 on page 199**. An icon indicating a Business Rule, is added to the link.

Figure 19 eInsight Business Process Designer - Adding Business Rules



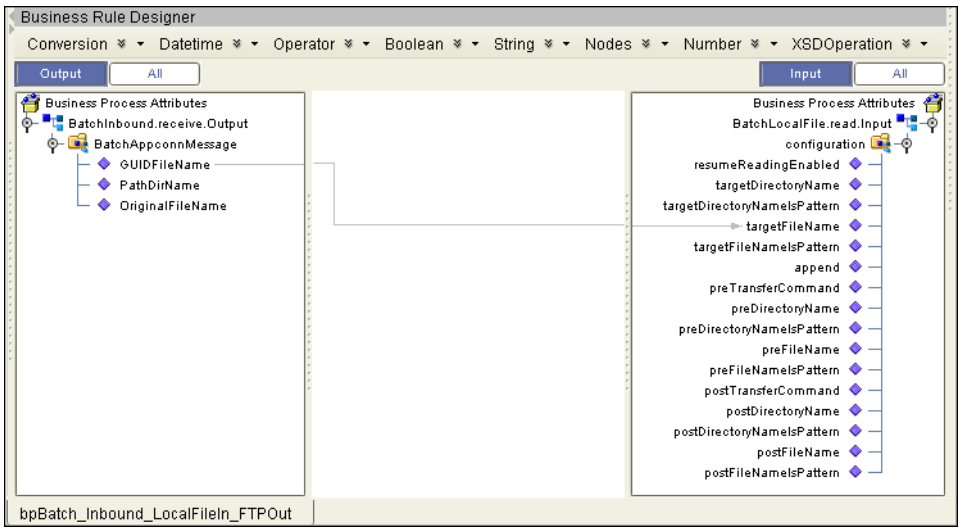
2 From the eInsight Business Process Designer toolbar, click the **Display Business Rules Designer** button. The Business Rule Designer appears at the bottom of the eInsight Business Process Designer.

3 Click on the **Business Rule** icon in the link between **BatchInbound.receive** and **BatchLocalFile.read** to display the Business Rule Output and Input Attributes in the Business Rule Designer. These Attributes can now be modified.

4 Create the **BatchInbound.receive > BatchLocalFile.read** Business Rule:

Map **GUIDFileName,** under **BatchInbound.receive.Output > BatchAppconnMessages** in the Output pane of the Business Rule Designer, to **targetFileName** under **BatchLocalFile.read.Input > configuration** in the Input pane of the Business Rule Designer. To do this click on **GUIDFileName** and drag your cursor to **targetFileName**. A link now displays between the two nodes. (see Figure 20).

Figure 20 eInsight Business Rule Designer



5 Add a Business Rule to the link between the **BatchLocalFile.read** and **BatchFTP.put** elements as described in **step 1**. Double-click the new Business Rule.

6 Create the **BatchLocalFile.read > BatchFTP.put** Business Rule:

From the Business Rule Designer, map **payload,** under **BatchLocalFile.read.Output** in the Output pane of the Business Rule Designer, to **payload** under **BatchFTP.put.Input** in the Input pane of the Business Rule Designer.

7 From the Business Process Designer toolbar, click the **Synchronize Graphical Model and Business Process Code** icon to synchronize the graphical interface to the Business Process code.

8 Click the Enterprise Designer's icon **Save All** to save your current changes.

6.6.3 Create a Connectivity Map

The Connectivity Map provides a canvas for assembling and configuring a Project's components.

1 In Enterprise Explorer's Project Explorer, right-click the **prjBatch_Inbound_LocalFileIn_FTPOut** Project and select **New > Connectivity Map** from the shortcut menu.

2 The New Connectivity Map appears and a node for the Connectivity Map is added to the Project Explorer tree labeled **CMap1**. Rename the Connectivity Map to **cmBatch_Inbound_LocalFileIn_FTPOut**.

The icons in the toolbar represent the available components used to populate the Connectivity Map canvas.

Select the External Applications

When creating a Connectivity Map, the eWays are associated with External Application. For example, to establish a connection to BatchFTP eWay, you must first select BatchFTP as an External Application to use in your Connectivity Map.

To create the External Applications used by the prjBatch_Inbound_LocalFileIn_FTPOut Project do the following:

1 Click the **External Application** icon on the Connectivity Map toolbar, and select the applications necessary for your Project. For this sample select the following External Applications:

- BatchInbound External Application
- BatchLocalFile External Application
- BatchFTP External Application

2 Icons representing the selected applications are added to the Connectivity Map toolbar.

Populate the Connectivity Map

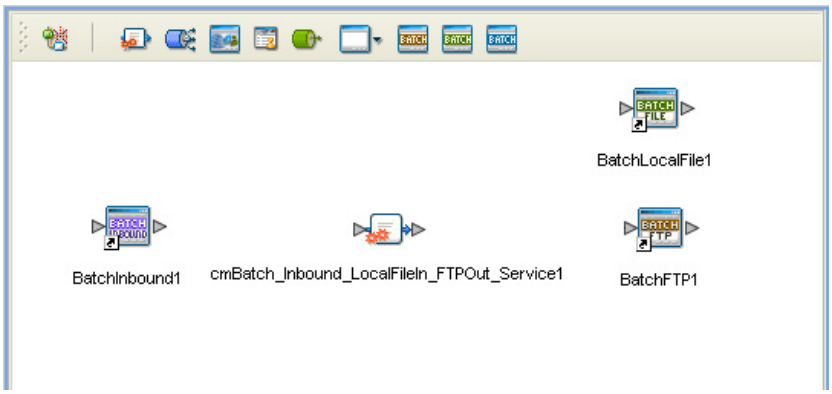
Add the Project components to the Connectivity Map by dragging the icons from the toolbar to the canvas.

1 Drag the following components onto the Connectivity Map canvas as displayed in **Figure 21 on page 201**:

- BatchInbound External Application
- Service
- BatchLocalFile External Application

- BatchFTP External Application

Figure 21 Connectivity Map with Components



2 From the Connectivity Map, rename the **Service** container to **bpBatch_Inbound_LocalFileIn_FTPOut1.**

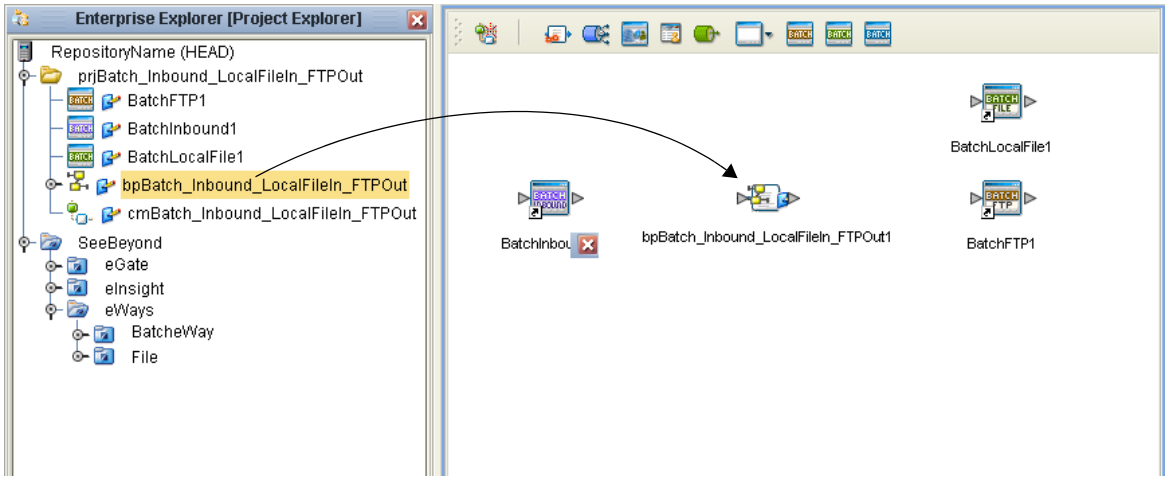
3 Click the **Save All** icon to save your current changes.

6.6.4 Binding the Project Components

After the Business Processes have been completed, the components are associated and the bindings are created from the Connectivity Map.

1 From the **cmBatch_Inbound_LocalFileIn_FTPOut** Connectivity Map, Drag and drop the **bpBatch_Inbound_LocalFileIn_FTPOut** Business Process from the Project Explorer to the **bpBatch_Inbound_LocalFileIn_FTPOut1 service**. The Services icon changes to indicate that it now contains a Business Process (see **Figure 22 on page 201**).

Figure 22 Connectivity Map - Binding the Components



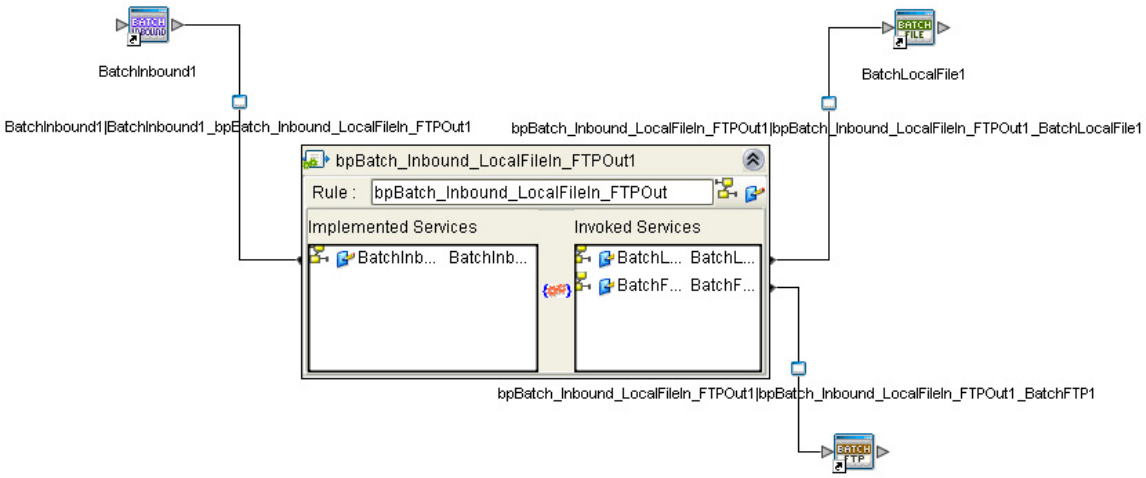
2 Double-click the **bpBatch_Inbound_LocalFileIn_FTPOut1** service. The **bpBatch_Inbound_LocalFileIn_FTPOut1** binding box appears using the **bpBatch_Inbound_LocalFileIn_FTPOut** Rule.

3 From the **bpBatch_Inbound_LocalFileIn_FTPOut1** binding box, map **BatchInboundMDB** (under Implemented Services) to the **BatchInbound1** application.

4 From the **bpBatch_Inbound_LocalFileIn_FTPOut1** binding box, map **BatchLocalFileReceiver** (under Invoked Services) to the **BatchLocalFile1** application.

5 From the **bpBatch_Inbound_LocalFileIn_FTPOut1** binding box, map **BatchFTPReceiver** (under Invoked Services) to **BatchFTP1** (see Figure 23).

Figure 23 Connectivity Map - Binding the Components



6 Minimize the **bpBatch_Inbound_LocalFileIn_FTPOut1** binding box and save your current changes.

6.6.5 Creating an Environment

Environments include the External Systems, Logical Hosts, Integration Servers and Message Servers used by a Project and contain the configuration information for these components. Environments are created using the Enterprise Designer's Environment Explorer and Environment Editor.

1 From the Enterprise Designer's Enterprise Explorer, click the **Environment Explorer** tab.

2 Right-click the Repository and select **New Environment**. A new Environment is added to the Environment Explorer tree.

3 Rename the new Environment to **envBatch_Inbound_LocalFileIn_FTPOut**.

4 From the Project Explorer tree, right-click **envBatch_Inbound_LocalFileIn_FTPOut** and select **New > BatchInbound External System**. Name the External System **esBatchInbound**. Click **OK**. The **esBatchInbound** box is added to the Environment Editor.

5 From the Project Explorer tree, right-click **envBatch_Inbound_FTPIn_LocalFileOut** and select **New > BatchFTP External System**. Name the External System **esBatchFTP**. Click **OK**. The **esBatchFTP** box is added to the Environment Editor.

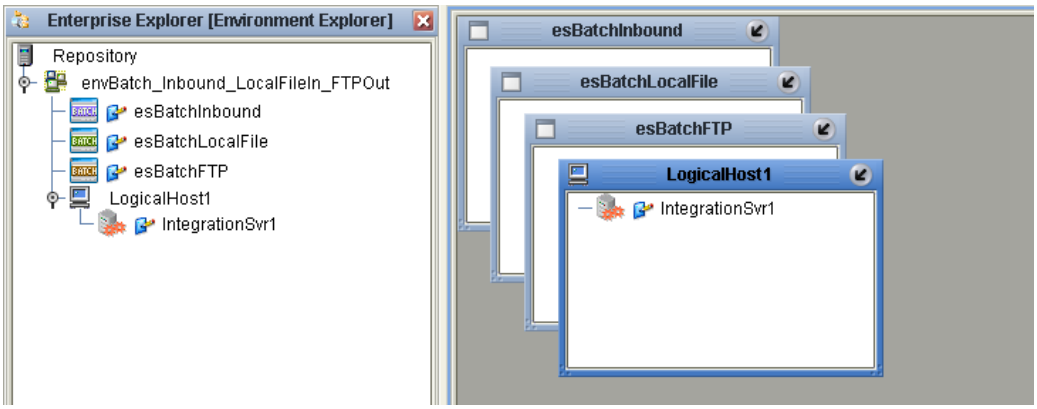
6 From the Project Explorer tree, right-click **envBatch_Inbound_FTPIn_LocalFileOut** and select **New > BatchLocalFile External System**. Name the External System **esBatchLocalFile**. Click **OK**. The **esBatchLocalFile** box is added to the Environment Editor.

7 From the Project Explorer tree, right-click **envBatch_Inbound_FTPIn_LocalFileOut** and select **New > Logical Host**. The **LogicalHost1** box is added to the Environment and **LogicalHost1** is added to the Environment Editor tree.

8 From the Environment Explorer tree, right-click **LogicalHost1** and select **New > Sun SeeBeyond Integration Server**. A new Integration Server (**IntegrationSvr1**) is added to the Environment Explorer tree under **LogicalHost1**.

9 Save changes to the Repository. The Environment Explorer and Environment Editor now appear as displayed in Figure 24.

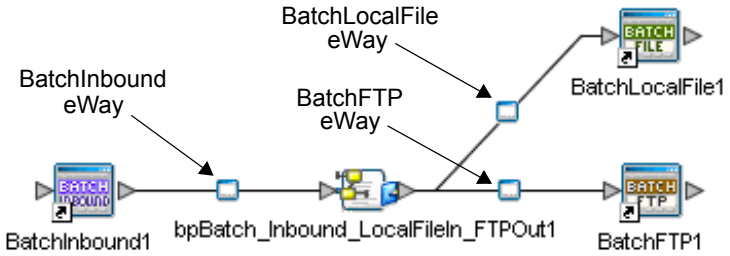
Figure 24 Environment Editor



6.6.6 Configuring the eWays Properties

The **prjBatch_Inbound_LocalFileIn_FTPOut** sample Project contains three eWays, each represented in the Connectivity Map as a node between an External Application and a Collaboration. The eWays facilitate communication and movement of data between the external applications and the eGate system (see Figure 25).

Figure 25 eWay Properties



The **eWay** properties must be set from both the Project Explorer's Connectivity Map and the Environment Explorer tree. To configure the eWays do the following:

Configuring the BatchInbound eWay Properties

1 From the Connectivity Map, double-click the **BatchInbound eWay**. The eWay Properties Editor appears.

2 Modify the properties for your system, including the settings for the **BatchInbound eWay** in Table 62, and click **OK**. The new eWay properties are saved.

Table 62 BatchInbound eWay Connectivity Map Properties
Table 63

BatchInbound eWay Connectivity Map Properties	
Settings - Set as directed, otherwise use the default settings	
Directory Name	*Your input directory (full address)*
File Name	BatchInbound.txt

3 The BatchInbound eWay Environment properties can keep the default settings for this sample.

Configuring the BatchLocalFile eWay Properties

1 From the Connectivity Map, double-click the **BatchLocalFile eWay**. The eWay Properties Editor appears.

2 Modify the properties for your system, including the settings in Table 64, and click **OK**. The properties are saved for the eWay.

Table 64 BatchLocalFile eWay Settings
Table 65

BatchLocalFile eWay Properties	
Target Location - Set as directed, otherwise use the default settings	
Append	Yes
Target File Name	BatchLocalFileOut.txt

Table 65

BatchLocalFile eWay Properties (Continued)	
Target File Name is Pattern	No
Target Directory Name	*The directory on the system where files are sent.*
Target Directory Name is Pattern	No

3 Refresh and save your default BatchLocalFile eWay properties. To do this, double-click the BatchLocalFile External System in the Environment Explorer to open the Properties Editor. Click OK to save the default settings.

Configuring the BatchFTP eWay Properties

The BatchFTP eWay properties are set from both the Connectivity Map and Environment Explorer. For more information on the BatchFTP eWay properties and the Properties Editor, see **"Creating and Configuring Batch eWays" on page 27** or see the *Sun SeeBeyond eGate™ Integrator User's Guide*.

Modifying the BatchFTP eWay Connectivity Map Properties

1 From the Connectivity Map, double-click the **BatchFTP** eWay. The eWay Properties Editor appears.

2 Modify the **BatchFTP** eWay Connectivity Map properties for your system, including the settings in Table 66, and click **OK**.

Table 66 BatchFTP eWay Connectivity Map Properties
Table 67

BatchLocalFile eWay Properties	
Target Location - Set as directed, otherwise use the default settings	
Target Directory Name	*The directory on the external system (absolute path) from which files are retrieved or sent*
Target File Name	*The FTP remote file name which is retrieved or sent*

Modifying the BatchFTP eWay (Environment Explorer) Properties

1 From the **Environment Explorer** tree, right-click the BatchFTP External System (**esBatchFTP** in this sample), and select **Properties**. The Properties Editor opens to the BatchFTP eWay Environment properties.

2 Modify the BatchFTP eWay Environment properties for your system, including the settings in Table 68, and click **OK**.

Table 68 BatchFTP eWay Connectivity Map Properties
Table 69

BatchFTP eWay Environment-Configuration Parameters
Target Location - Set as directed, otherwise use the default settings

Table 69

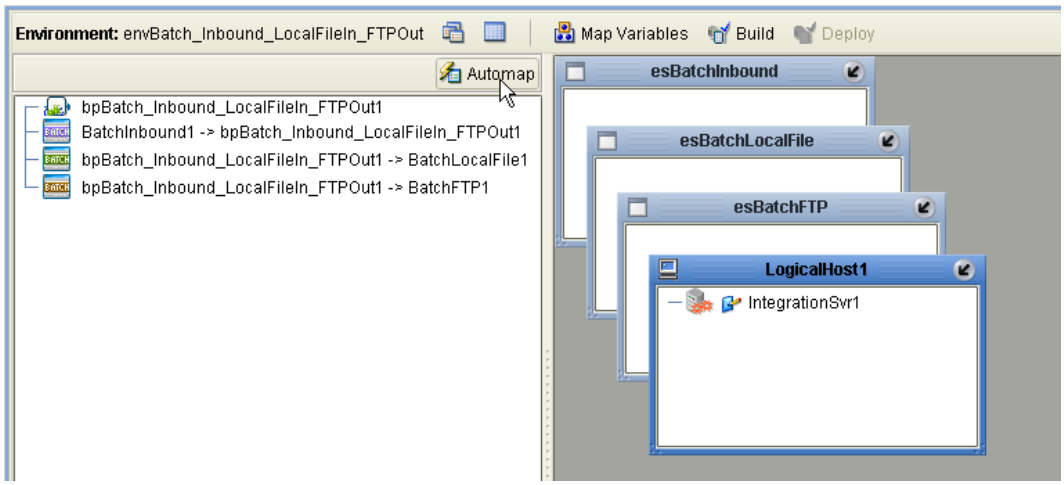
BatchFTP eWay Environment-Configuration Parameters (Continued)	
FTP - Set as directed, otherwise use the default settings.	
Host Name	The name of the external system to which the eWay connects
Password	Password required to log into the external system
Server Port	Port number to use to connect to the FTP server
User Name	*User ID used to login to the external system*

6.6.7 Creating the Deployment Profile

A Deployment Profile is used to assign Collaborations and message destinations to the integration server and message server. Deployment Profiles are created using the Deployment Editor.

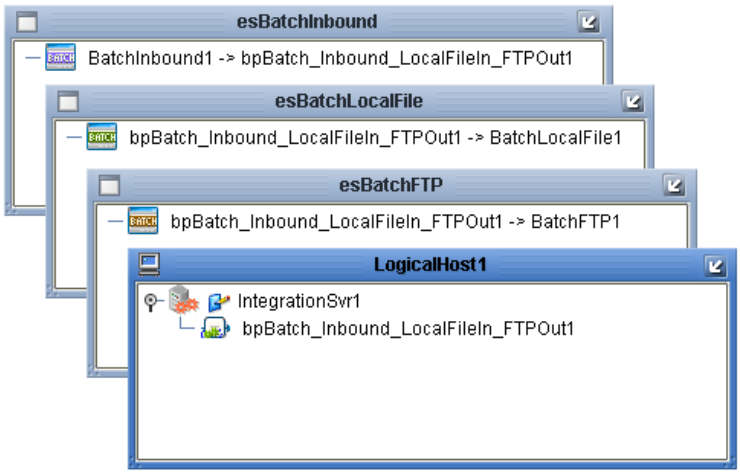
1. From the Enterprise Explorer's Project Explorer, right-click the Project (**prjBatch_Inbound_LocalFileIn_FTPOut**) and select **New** > **Deployment Profile**.
2. Enter a name for the Deployment Profile (for this sample **dpBatch_Inbound_LocalFileIn_FTPOut**). Make sure that the selected Environment is **envBatch_Inbound_LocalFileIn_FTPOut**. Click **OK**.
3. Click the **Automap** icon as displayed in **Figure 26 on page 206**.

Figure 26 dpBatch_Inbound_LocalFileIn_FTPOut Deployment Profile



The Project's components are automatically mapped to their system window as seen in **Figure 27 on page 207**.

Figure 27 Completed dpBatch_Inbound_LocalFileIn_FTPOut Profile



4 If any of your Project components did not successfully map to an external system, open each of your eWay's configuration properties (Connectivity Map and Environment) and click **OK** to close and save the current configuration, then click **Automap** again.

5 Save your changes to the Repository.

6.6.8 Creating and Starting the Domain

To deploy your Project, you must first create a domain. A domain is an instance of a Logical Host. After the domain is created, the Project is built and then deployed.

Create and Start the Domain

1 Navigate to your **<*JavaCAPS51*>\logicalhost** directory (where <*JavaCAPS51*> is the location of your Sun Java Composite Application Platform Suite installation.

2 Double-click the **domainmgr.bat** file. The **Domain Manager** appears.

3 If you have already created a domain, select your domain in the Domain Manager and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

4 If there are no existing domains, a dialog box indicates that you can create a domain now. Click **Yes**. The **Create Domain** dialog box appears.

5 Make any necessary changes to the **Create Domain** dialog box and click **Create**. The new domain is added to the Domain Manager. Select the domain and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

6 For more information about creating and managing domains see the *Sun SeeBeyond eGate™ Integrator System Administration Guide*.

6.6.9 Building and Deploying the Project

The **Build** process compiles and validates the Project's Java files and creates the Project EAR file.

For more information on deploying the EAR file, refer to **Deploying an EAR File** on page 24

Build the Project

1. From the Deployment Editor toolbar, click the **Build** icon.
2. If there are any validation errors, a **Validation Errors** pane will appear at the bottom of the Deployment Editor and displays information regarding the errors. Make any necessary corrections and click **Build** again.
3. After the Build has succeeded you are ready to deploy your Project.

Deploy the Project

1. From the Deployment Editor toolbar, click the **Deploy** icon. Click **Yes** when the **Deploy** prompt appears.
2. A message appears when the project is successfully deployed. You can now test your sample.

Note: *Projects can also be deployed from the Enterprise Manager. For more information about using the Enterprise Manager to deploy, monitor, and manage your projects, see the Sun SeeBeyond eGate™ Integrator System Administration Guide.*

6.6.10 Running the Sample

To run your deployed sample Project do the following

1. From your configured input directory, paste (or rename) the sample input file to trigger the eWay.
2. From your output directory, verify the output data.

Note: *For UNIX, be sure to upload the appropriate LogicalHost*

For more information on running a Project that utilizes eInsight from the Enterprise Designer see the *Sun SeeBeyond eInsight™ Business Process Manager User's Guide* and the *Sun SeeBeyond eGate™ Integrator User's Guide*.

Chapter 7

Using the Batch eWay with Java Collaborations

This chapter provides an introduction to the Batch eWay components used in Java Collaboration based Projects. This chapter assumes that you are already familiar with Java Integration Suite concepts and that you understand how to create a Project using the Enterprise Designer.

For a complete explanation of Java Integration Suite terminology and concepts, see the *Sun SeeBeyond eGate™ Integrator User's Guide*.

What's in This Chapter

7.1 Batch eWay Components

eWay components that are unique to the Batch eWay include the following:

Batch eWay OTDs

Object Type Definitions (OTDs) map input and output message segments at the field level. The Batch eWay has the following seven OTDs:

- **BatchFTP OTD**: supports connections to external FTP servers.
- **BatchFTPOverSSL OTD**: supports secure data transfer using Secure Sockets Layer (SSL) protocol.
- **BatchSCP OTD**: supports secure data transfer using Secure Copy Protocol with Secure Shell (SSH) as an underlying protocol.
- **BatchSFTP OTD**: supports SSH File Transfer Protocol (SFTP protocol). SFTP protocol. SFTP provides a range of operations on remote files, such as resuming interrupted transfers, directory listings, and remote file removal.
- **BatchLocalFile OTD**: supports data file publish and subscribe functions for local file systems.

- **BatchRecord OTD**: provides functions for extracting records out of files, parsing files into specific records, and defining the content of files as records.
- **BatchInbound OTD**: provides functionality for receiving files, renaming files with GUID file names, and triggering a Business Process or Collaboration.

For information on the Batch eWay OTDs, see **"Overview of the Batch OTDs" on page 149**.

7.2 Considerations

The following items must be considered when implementing a Batch eWay Project:

- An error may occur when activating an existing or imported Project that uses the **BatchLocalFile** eWay. This is due to a new BatchLocalFile Environment property that has been added to the eWay. To resolve this, refresh (open and close) your Project's BatchLocalFile Environment properties, and save the Project to the repository before activating the Project.
- When using FTP with an **AS400 UNIX (USF)** system, the following FTP configuration settings are required:
  - FTP - Use PASV: **No** (see **"Use PASV" on page 38**)
  - FTP Raw Commands - Pre Transfer Raw Commands: **site namefmt 1** (see **"Pre Transfer Raw Commands" on page 40**)

7.3 Importing a Sample Project

Sample eWay Projects are included as part of the installation package. To import a sample eWay Project to the Enterprise Designer do the following:

1. The sample files are uploaded with the eWay's documentation SAR file and downloaded from the Sun Java Composite Application Platform Suite Installer's Documentation tab. The **Batch_eWay_Sample.zip** file contains the various sample Project ZIP files. Extract the samples to a local file.
2. Save all unsaved work before importing a Project.
3. From the Enterprise Designer's Project Explorer pane, right-click the Repository and select **Import** from the shortcut menu. The **Import Manager** appears.
4. Browse to the directory that contains the sample Project zip file. Select the sample file (for example, **prjBatch_Record.zip**) and click **Import**. After the sample Project is successfully imported, click **Close**.
5. Before an imported sample Project can be run you must do the following:
   - Create an Environment (see **"Creating an Environment" on page 221**)
   - Configure the eWays for your specific system (see **"Creating and Configuring Batch eWays" on page 27**)

- Create a Deployment Profile (see **"Creating the Deployment Profile" on page 239**)
- Create and start a domain (see **"Creating and Starting the Domain" on page 225**)
- Build and deploy the Project (see **"Building and Deploying the Project" on page 226**)

7.4 The Batch eWay Sample JCD Projects

This chapter provides step by step directions for manually creating the sample Batch eWay Projects that use Java Collaboration Definitions.

- **The prjBatch_Inbound_FTPIn_LocalFileOut Sample Project** on page 213,
- **The prjBatch_Streaming Sample Project** on page 227
- **The prjBatch_Record Sample Project** on page 242
- **The prjBatch_Secure Project** on page 256

Sample data files

Sample data files for the Batch eWay Projects are included with the samples. See Input_Files_Readme.txt included with the sample data files for more information.

7.4.1 Sample Project Descriptions

The prjBatch_Inbound_FTPIn_LocalFileOut Sample Project

The prjBatch_Inbound_FTPIn_LocalFileOut Sample Project demonstrates the following:

- The BatchInbound eWay subscribes to a local directory and periodically checks for a specific file. If the file is present, it renames the file with a GUID, and triggers the Java Collaboration.
- The BatchFTP eWay gets the associated target file from the FTP external system and writes the file content to the BatchLocalFile eWay payload.
- The BatchLocalFile eWay gets the GUID file name from the BatchInbound eWay, and publishes its payload to a file in a local directory.

The prjBatch_Streaming Project

Stream allows you to read large files by streaming the data. Three of the Batch OTDs are able to receive streamed data: **BatchFTP**, **BatchLocalFile**, and **BatchRecord**. The **prjBatch_Streaming** Project demonstrates the following:

- The File eWay polls the input directory periodically. When it sees a specified file, it triggers the Collaboration.

- The BatchFTP eWay streams data to the BatchLocalFile eWay.
- The BatchLocalFile eWay renames the resulting data file and moves the output data to a new location.
- The BatchLocalFile streams the new created file to the BatchRecord eWay.
- The BatchRecord eWay parses the file in the output file and publishes the parsed records to the File eWay.
- The File eWay publishes the files to a local directory.

This sample works with an input file of any size. but to see the streaming feature in action, use a larger file.

The prjBatch_Record Project

The **prjBatch_Record** Project demonstrates the following:

- The File eWay polls the input directory periodically. When it sees a specified file, it triggers the Collaboration.
- When triggered, the BatchRecord_Create eWay creates a record with an incrementing number (string), and puts the record into the payload along with the specified record delimiter.
- When the payload contains ten records, the payload out using the File eWay. The payload is published to the BatchRecord_Parse eWay.
- The BatchRecord_Parse eWay parses the ten records and publishes each record to the file eWay.
- The FileOut eWay writes the Files to a local directory.

The prjBatch_Secure Project

The **prjBatch_Secure** Project contains three scenarios (almost like three separate Projects) to demonstrate the SSL and SSH secure file transfer functions of the **BatchFTPOverSSL**, **BatchSFTP**, and **BatchSCP OTDs**.

The project uses inbound File eWays to poll an external directory for a specific input file. When the file is present appropriate File eWay triggers a Collaboration to do the following:

Scenario 1 - FTP Over SSL

- Gets a file from a remote directory and publishes it to a local directory.
- Publishes a file from local directory to a remote directory.
- Downloads a file from a remote directory to a local directory.
- Uploads a file from a local directory to remote directory.
- Creates remote directories and lists them.
- Deletes remote directories and lists the results.

Scenario 2 - SFTP

- Gets a file from remote directory and publishes it to a local directory.
- Creates a new remote directory
- Changes directories to the newly created remote directory, and publishes a file there.
- Renames the published file.
- Deletes an unwanted file from remote directory.

Scenario 3 - SCP

- Recursively gets a file from a remote directory and publishes it to a local directory.
- Recursively puts a file from a local directory to a remote directory.
- Gets a file (non-recursive) from a remote directory and publishes it to a local directory.
- Puts a file (non-recursive) from a local directory to a remote directory.

7.5 The prjBatch_Inbound_FTPIn_LocalFileOut Sample Project

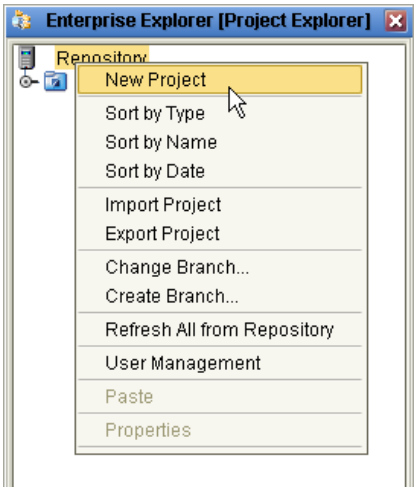
This section describes how the components of the Batch_FTPIn_LFOut_Sample Project are created. To manually create the Batch_FTPIn_LFOut_Sample Project, do the following:

7.5.1 Create a Project

The first step is to create a new Project in the Enterprise Designer.

1. Start the Enterprise Designer.
2. From the Enterprise Explorer's Project Explorer tab, right-click the Repository and select **New Project** (see **Figure 28 on page 214**). A new Project (Project1) appears on the Project Explorer tree.

Figure 28 Enterprise Explorer - New Project



3 Rename the Project (for this sample, **prjBatch_Inbound_FTPIn_LocalFileOut**).

7.5.2 Creating Java Collaboration Definitions

The next step in the sample Project is to create the **jcdBatch_Inbound_FTPIn_LocalFileOut** Collaboration Definition. A Java Collaboration Definition contains Business Rules that define the processing and transport of data between the eGate components.

The Collaboration Definition Wizard (Java) is used to create the Java Collaboration Definitions. Once a Collaboration is created, the Collaboration Editor is used to create the Business Rules of the Collaboration.

Create the jcdBatch_Inbound_FTPIn_LocalFileOut Collaboration

The **jcdBatch_Inbound_FTPIn_LocalFileOut** Collaboration defines how data is transferred between the BatchInbound application, the BatchFTP application and the Outbound BatchLocalFile.

1 From the Project Explorer, right-click **prjBatch_Inbound_FTPIn_LocalFileOut** and select **New** > **Collaboration Definition (Java)** from the shortcut menu. The **Collaboration Definition Wizard (Java)** appears.

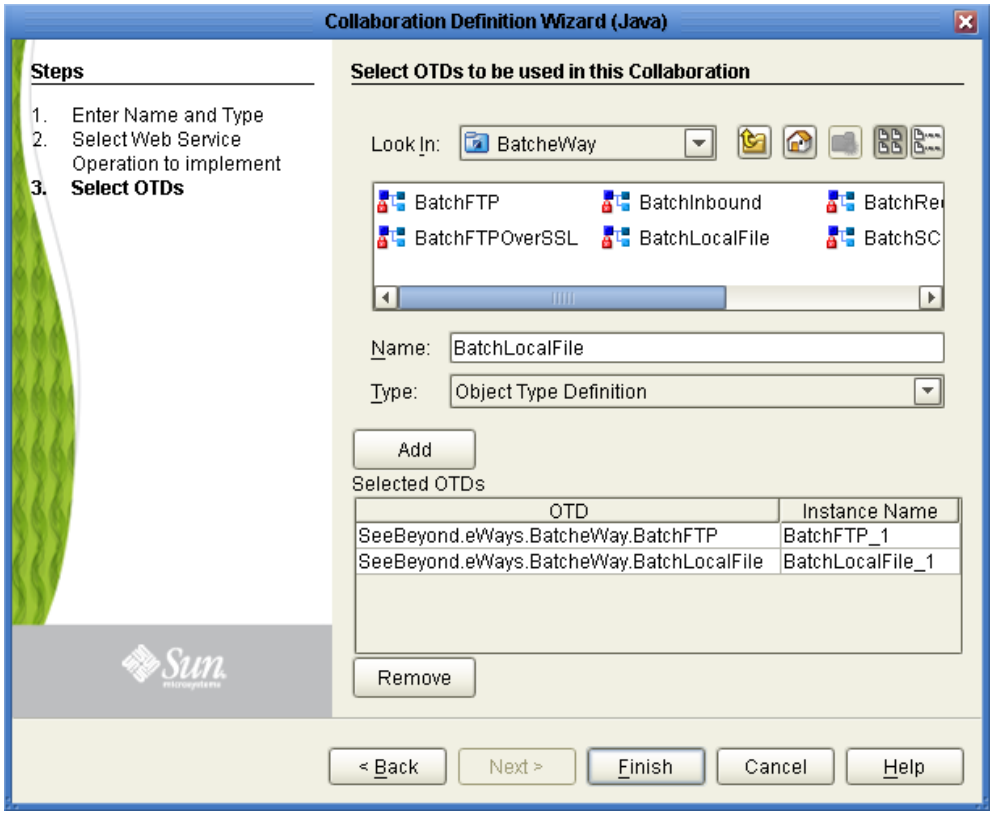
2 Enter a Collaboration Definition name (for this sample **jcdBatch_Inbound_FTPIn_LocalFileOut**) and click **Next**.

3 For Step 2 of the Wizard, double-click **SeeBeyond** > **eWays** > **BatcheWay** > **BatchInbound** > **receive**. The File Name field now displays **receive**. Click **Next**.

4 For Step 3 of the Wizard, from the Select OTDs selection window, double-click **SeeBeyond** > **eWays** > **BatcheWay > BatchFTP**. The **BatchFtp** OTD is added to the Selected OTDs field.

5 From the Select OTDs selection window, double-click **BatchLocalFile (**SeeBeyond > eWays > BatcheWay > **BatchLocalFile)**. The **BatchLocalFile** OTD is added to the Selected OTDs field (see **Figure 29 on page 215**).

Figure 29 jcdBatch_Inbound_FTPIn_LocalFileOut - Select OTDs



6 Click **Finish**. The new **jcdBatch_Inbound_FTPIn_LocalFileOut** Collaboration appears in the Project Explorer tree.

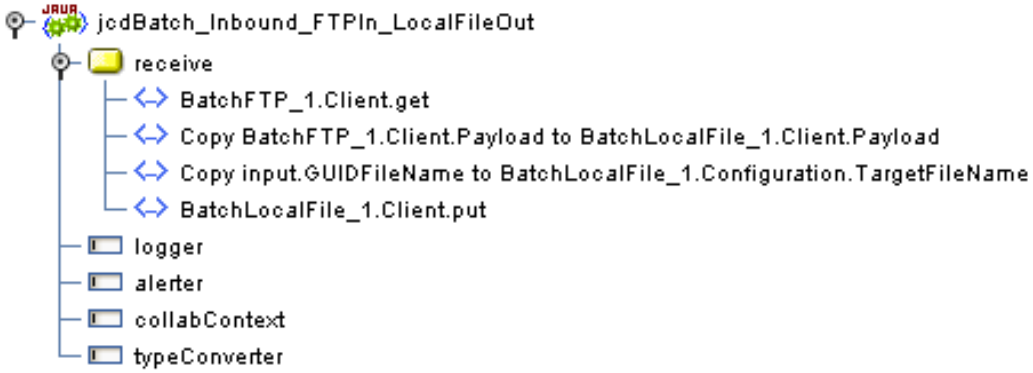
7.5.3 Using the Collaboration Editor (Java)

The **prjBatch_Inbound_FTPIn_LocalFileOut** Project uses the **jcdBatch_Inbound_FTPIn_LocalFileOut** Collaboration created in the previous section. To complete the Collaboration, use the Collaboration Editor (Java) to create the Business Rules.

Create the jcdBatch_Inbound_FTPIn_LocalFileOut Collaboration Business Rules

Be careful to open all nodes specified in the directions to connect the correct items. The **jcdBatch_Inbound_FTPIn_LocalFileOut** Collaboration contains the Business Rule displayed in **Figure 30 on page 216**.

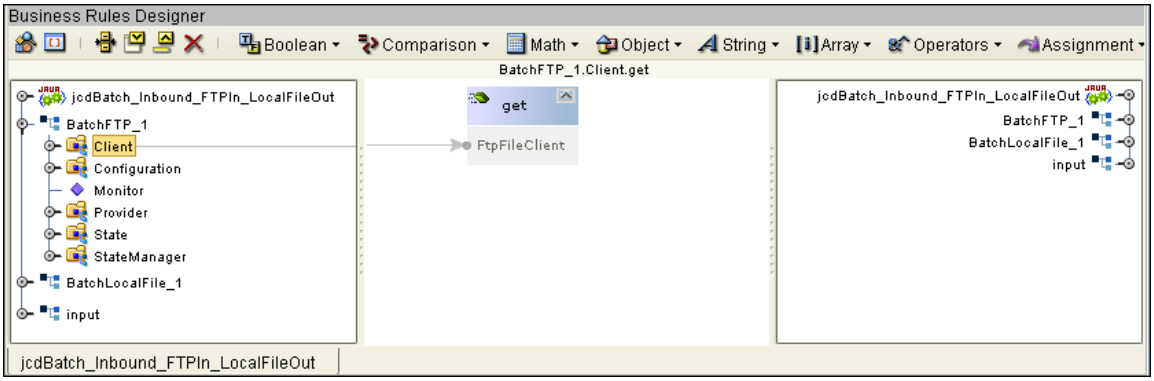
Figure 30 jcdBatch_Inbound_FTPIn_LocalFileOut Business Rules



To create the **jcdBatch_Inbound_FTPIn_LocalFileOut** Collaboration Business Rules, do the following:

1. From the Project Explorer tree, double-click **jcdBatch_Inbound_FTPIn_LocalFileOut**. The Collaboration Editor (Java) opens to the **jcdBatch_Inbound_FTPIn_LocalFileOut** Collaboration.
2. To create comments for the Business Rules, from the Business Rules toolbar, click the **comment** icon. The **Enter a Comment** dialog box appears. Enter the comment and click **OK**. The comment is placed on the Business Rules tree under the last selected item. Once the Comment is created, it can be moved by clicking the comment and dragging it up or down the Business Rules tree to a new location.
3. The **BatchFTP_1.Client.get** rule gets the file from the FTP Client. Create the **BatchFTP_1.Client.get** rule:
   A. Right-click **Client** under the **BatchFTP_1** node in the left pane of the Business Rules Designer, and choose **Select method to call** from the shortcut menu. The method selection window appears.
   B. Select **get()** from the method selection window. The **get** method box appears in the Business Rules Designer canvas (see Figure 31).

Figure 31 Collaboration Editor (Java) - Business Rules Designer



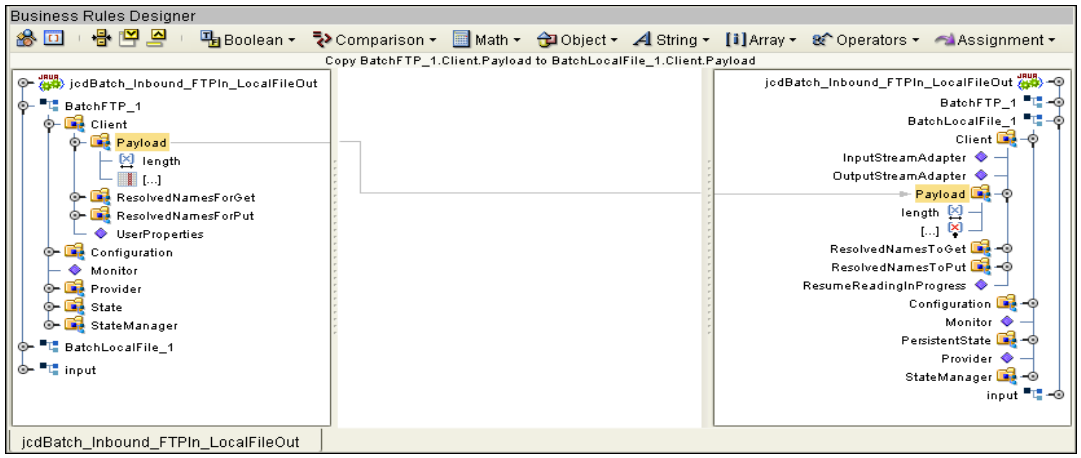
4. The **Copy BatchFTP_1.Client.Payload to BatchLocalFile_1.Client.Payload rule writes the contents of the retrieved file to the payload of the BatchLocalFile.**

Create the **Copy BatchFTP_1.Client.Payload to BatchLocalFile_1.Client.Payload** rule:

A Click **rule** on the Business Rules toolbar to add a new rule in the Business Rules pane.

B Map **Payload** under **BatchFTP_1 > Client** in the left pane of the Business Rules Designer, to **Payload** under **BatchFTP_1 > Client** in the right pane of the Business Rules Designer. To do this, Click on **Payload** in the left pane of the Business Rules Designer, and drag your cursor to **Payload** in the right pane of the Business Rules Designer (see Figure 32).

Figure 32 Collaboration Editor (Java) - Business Rules Designer



5 The **Copy input.GUIDFileName to BatchLocalFile_1.Configuration.TargetFileName** rule gives a GUID to the target file the BatchLocalFile eWay. Create the **Copy input.GUIDFileName to BatchFTP_1.Configuration.TargetFileName** rule:

A From the Business Rules toolbar, click the **Rule** icon to add a new rule to the Business Rules tree.

B Map **GUIDFileName** under **input** (OTD) in the left pane of the Business Rules Designer, to **TargetFileName** under **BatchLocalFile_1 > Configuration** in the right pane of the Business Rules Designer.

6 The **BatchLocalFile_1.Client.put** rule publishes the file to a local directory. Create the **BatchLocalFile_1.Client.put** rule:

A Click **rule** on the Business Rules toolbar to add a new rule in the Business Rules pane.

B Right-click **Client** under the **BatchLocalFile_1** node in the left pane of the Business Rules Designer, and choose **Select method to call** from the shortcut menu. The method selection window appears.

C Select **put()** from the method selection window. The **put** method box appears in the Business Rules Designer canvas.

7 From the editor's toolbar, click **Validate** to check the Collaboration for errors.

8 Save your current changes to the Repository.

Note: *See the Sun SeeBeyond eGate™ Integrator User's Guide for more information on editing Collaborations.*

7.5.4 Creating a Connectivity Map

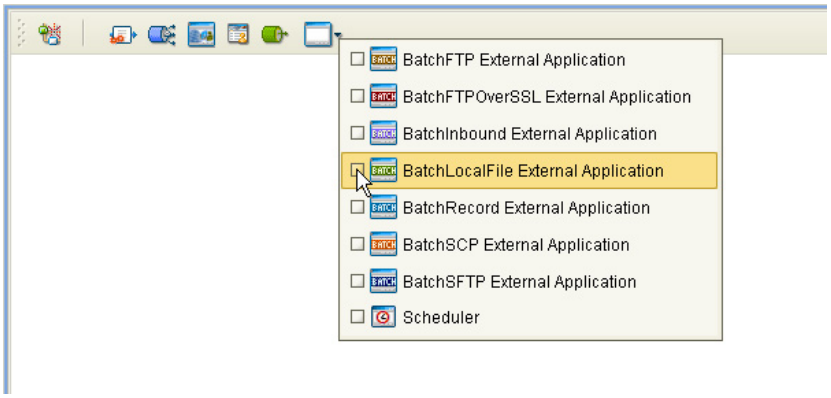
A Connectivity Map provides a canvas for assembling and configuring a Project's components.

1. In the Enterprise Explorer's Project Explorer, right-click **prjBatch_Inbound_FTPIn_LocalFileOut** and select **New > Connectivity Map** from the shortcut menu.
2. The new Connectivity Map appears and adds a node on the Project Explorer tree labeled **CMap1**.
3. Rename the **CMap1** Connectivity Map to **cmBatch_Inbound_FTPIn_LocalFileOut**.

Selecting the External Applications

In the Connectivity Map, eWays are associated with External Applications. For example, to establish a connection to an external FTP server, you must first select BatchLocalFile as an External System to use in the Connectivity Map (see Figure 33).

Figure 33 Connectivity Map - External Applications



1. Click the **External Application** icon on the Connectivity Map toolbar.
2. Select the External Applications for your Project. For this sample, select the following:
   - **BatchFTP External Application**
   - **BatchInbound External Application**
   - **BatchLocalFile External Application**

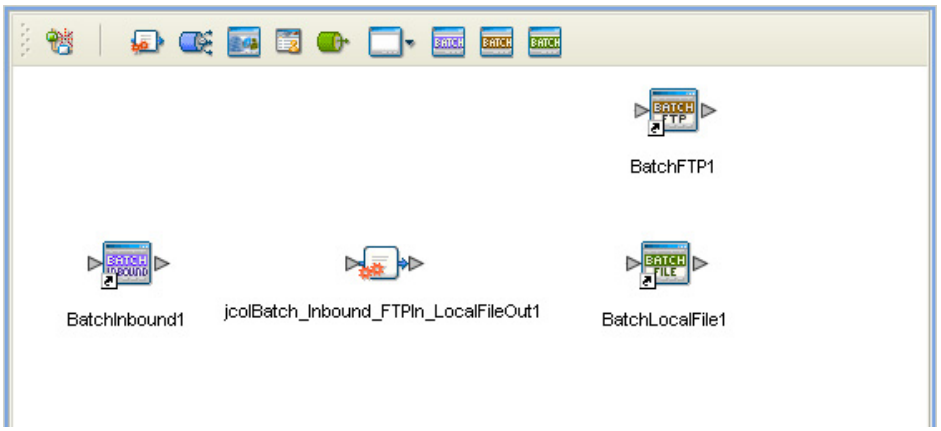
   This adds icons representing the **BatchFTP, BatchInbound,** and **BatchLocalFile** External Application icons to the Connectivity Map toolbar.

7.5.5 Populating the Connectivity Map

Add the Project components to the Connectivity Map by dragging the icons from the Connectivity Map toolbar to the canvas.

1. For the **prjBatch_Inbound_FTPIn_LocalFileOut** sample, drag and drop the following components onto the Connectivity Map canvas as displayed in Figure 34:
   - BatchInbound External Application
   - Service (container for the Collaboration)
   - **BatchLocalFile** External Application
   - **BatchFTP** External Application

Figure 34 prjBatch_Inbound_FTPIn_LocalFileOut Connectivity Map



Rename **Service1** in the Connectivity Map by right-clicking the item, selecting Rename. Rename the item **jcolBatch_Inbound_FTPIn_LocalFileOut1**.

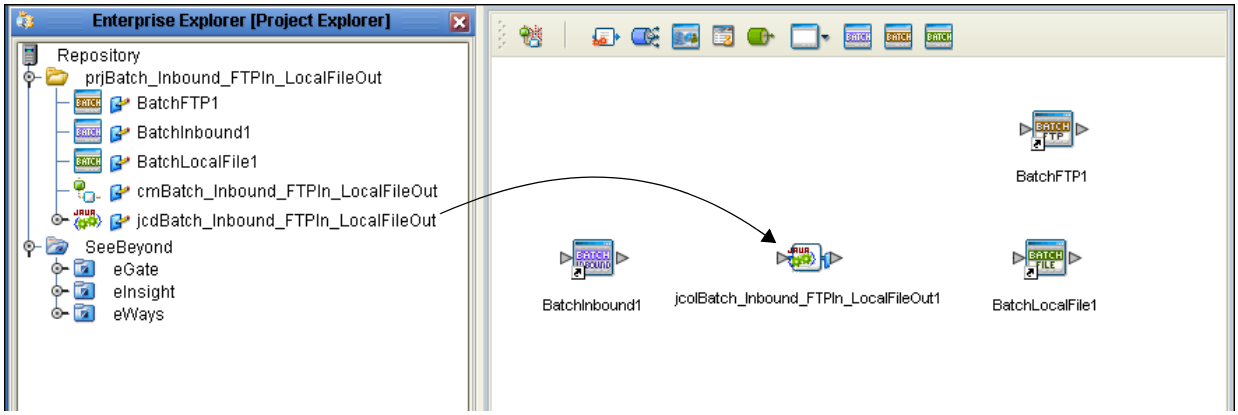
7.5.6 Creating Collaboration Bindings

After the Collaboration is complete, associate the components and create Collaboration Bindings by linking the components in the Connectivity Map.

Create the Collaboration Bindings

1. From the Project Explorer, double-click the **cmBatch_Inbound_FTPIn_LocalFileOut** Connectivity Map.
2. Drag and drop the **jcdBatch_Inbound_FTPIn_LocalFileOut** Collaboration from the Project Explorer onto the **jcolBatch_Inbound_FTPIn_LocalFileOut1** (service). If the Collaboration is successfully associated the service icon's "gears" changes from red to green **(see** Figure 35).

Figure 35 Connectivity Map - Binding the Project components



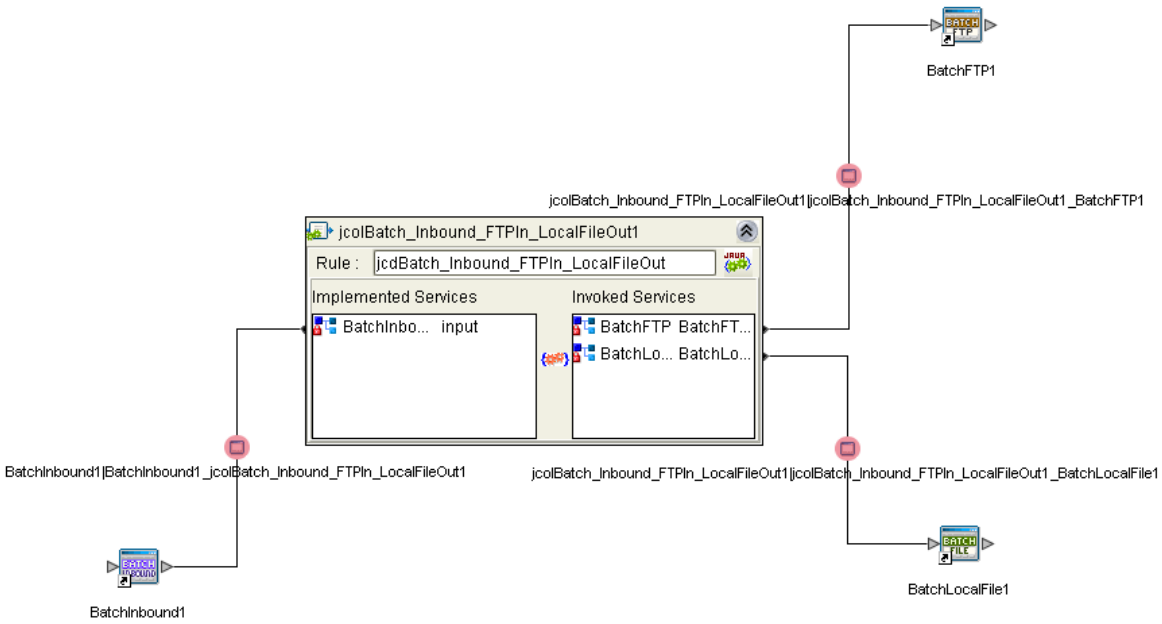
3 From the connectivity Map, double-click **jcolBatch_Inbound_FTPIn_LocalFileOut1. The jcolBatch_Inbound_FTPIn_LocalFileOut1 binding dialog box opens with jcdBatch_Inbound_FTPIn_LocalFileOut** as the rule.

4 From the Connectivity Map, map **BatchInbound input** under Implemented Services in the **jcolBatch_Inbound_FTPIn_LocalFileOut1 binding dialog box**, to **BatchInbound1**.

5 Map **BatchFTP_1 BatchFTP** under Invoked Services in the **binding dialog box**, to **BatchFTP1**.

6 Map **BatchLocalFile_1 BatchLocalFile** under Invoked Services in the **binding dialog box**, to **BatchLocalFile1 (see** Figure 36**)**.

Figure 36 Connectivity Map - Binding the Project components



7 Minimize the **jcolBatch_Inbound_FTPIn_LocalFileOut1 binding dialog box and save your changes to the Repository.**

7.5.7 Creating an Environment

Environments include the External Systems, Logical Hosts, Integration Servers and Message Servers used by a Project and contain the configuration information for these components. Environments are created using the Enterprise Designer's Environment Explorer and Environment Editor.

1 From the Enterprise Designer's Enterprise Explorer, click the **Environment Explorer** tab.

2 Right-click the Repository and select **New Environment**. A new Environment is added to the Environment Explorer tree.

3 Rename the new Environment to **envBatch_Inbound_FTPIn_LocalFileOut**.

4 From the Project Explorer tree, right-click **envBatch_Inbound_FTPIn_LocalFileOut** and select **New > BatchInbound External System**. Name the External System **esBatchInbound**. Click **OK**. The **esBatchInbound** box is added to the Environment Editor.

5 From the Project Explorer tree, right-click **envBatch_Inbound_FTPIn_LocalFileOut** and select **New > BatchFTP External System**. Name the External System **esBatchFTP**. Click **OK**. The **esBatchFTP** box is added to the Environment Editor.

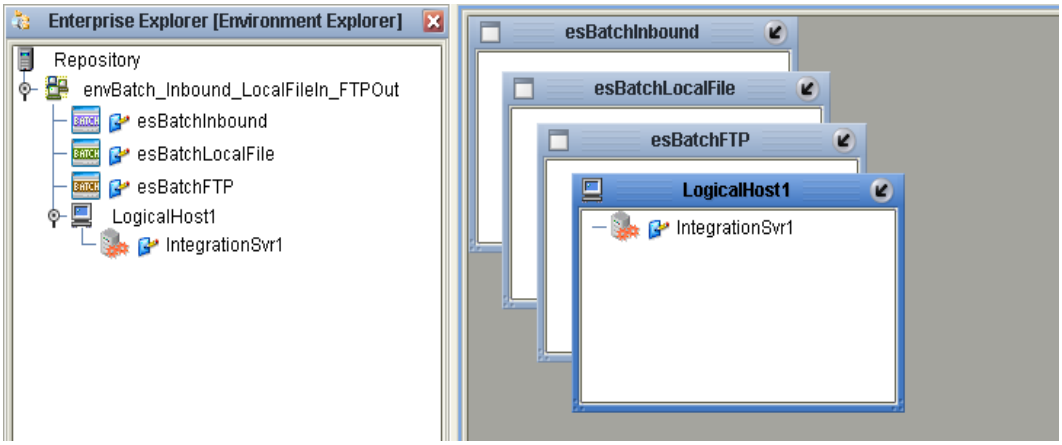
6 From the Project Explorer tree, right-click **envBatch_Inbound_FTPIn_LocalFileOut** and select **New > BatchLocalFile External System**. Name the External System **esBatchLocalFile**. Click **OK**. The **esBatchLocalFile** box is added to the Environment Editor.

7 From the Project Explorer tree, right-click **envBatch_Inbound_FTPIn_LocalFileOut** and select **New > Logical Host**. The **LogicalHost1** box is added to the Environment and **LogicalHost1** is added to the Environment Editor tree.

8 From the Environment Explorer tree, right-click **LogicalHost1** and select **New > Sun Integration Server**. A new Integration Server (**IntegrationSvr1**) is added to the Environment Explorer tree under **LogicalHost1**.

9 Save changes to the Repository. The Environment Explorer and Environment Editor now appear as displayed in **Figure 37 on page 222**.

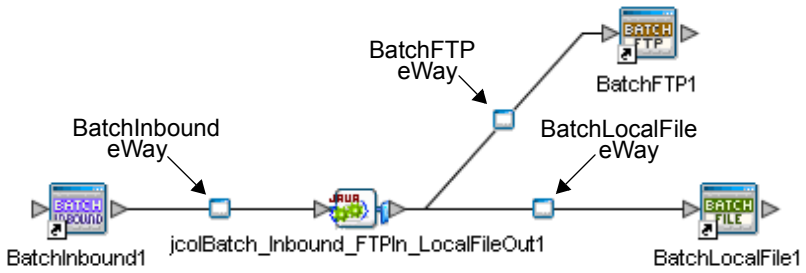
Figure 37 Environment Editor



7.5.8 Configuring the eWays Properties

The **prjBatch_Inbound_FTPIn_LocalFileOut** sample Project contains three eWays, each represented in the Connectivity Map as a node between an External Application and a Collaboration. The eWays facilitate communication and movement of data between the external applications and the eGate system (see Figure 38).

Figure 38 eWay Properties



The **eWay** properties must be set from both the Project Explorer's Connectivity Map and the Environment Explorer tree. To configure the eWays do the following:

Configuring the BatchInbound eWay Properties

1 From the Connectivity Map, double-click the **BatchInbound eWay**. The eWay Properties Editor appears.

2 Modify the properties for your system, including the settings for the **BatchInbound eWay** in Table 70, and click **OK**. The new eWay properties are saved.

Table 70 BatchInbound eWay Connectivity Map Properties
Table 71

BatchInbound eWay Connectivity Map Properties	
Settings - Set as directed, otherwise use the default settings	
Directory Name	*Your input directory (full address)*
File Name	BatchInbound.txt

3 The BatchInbound eWay Environment properties can keep the default settings for this sample.

Configuring the BatchFTP eWay Properties

The BatchFTP eWay properties are set from both the Connectivity Map and Environment Explorer. For more information on the BatchFTP eWay properties and the Properties Editor, see **"Creating and Configuring Batch eWays" on page 27** or see the *Sun SeeBeyond eGate™ Integrator User's Guide*.

Modify the BatchFTP eWay Connectivity Map Properties

1 From the Connectivity Map, double-click the **BatchFTP** eWay. The eWay Properties Editor appears.

2 Modify the **BatchFTP** eWay Connectivity Map properties for your system, including the settings in Table 72, and click **OK**.

Table 72 BatchFTP eWay Connectivity Map Properties
Table 73

BatchFTP eWay Connectivity Map Properties	
Target Location - Set as directed, otherwise use the default settings	
Target Directory Name	Batch51
Target File Name	BatchFTPIn.txt

Modify the BatchFTP eWay (Environment Explorer) Properties

1 From the **Environment Explorer** tree, right-click the BatchFTP External System (**esBatchFTP** in this sample), and select **Properties**. The Properties Editor opens to the BatchFTP eWay Environment properties.

2 Modify the BatchFTP eWay Environment properties for your system, including the settings in Table 74, and click **OK**.

Table 74 BatchFTP eWay Environment Properties
Table 75

BatchFTP eWay Environment Properties	
FTP - Set as directed, otherwise use the default settings.	
Host Name	*The name of the external system to which the eWay connects*
Password	*Password required to log into the external system*
Server Port	*Port number to use to connect to the FTP server*
User Name	*User ID used to login to the external system*

Configuring the BatchLocalFile eWay Properties

1. Double-click the **BatchLocalFile eWay**. The eWay Properties Editor appears.
2. Modify the properties for your system, including the settings in Table 76, and click **OK**. The properties are saved for the eWay.

Table 76 BatchLocalFile eWay Connectivity Map Properties
Table 77

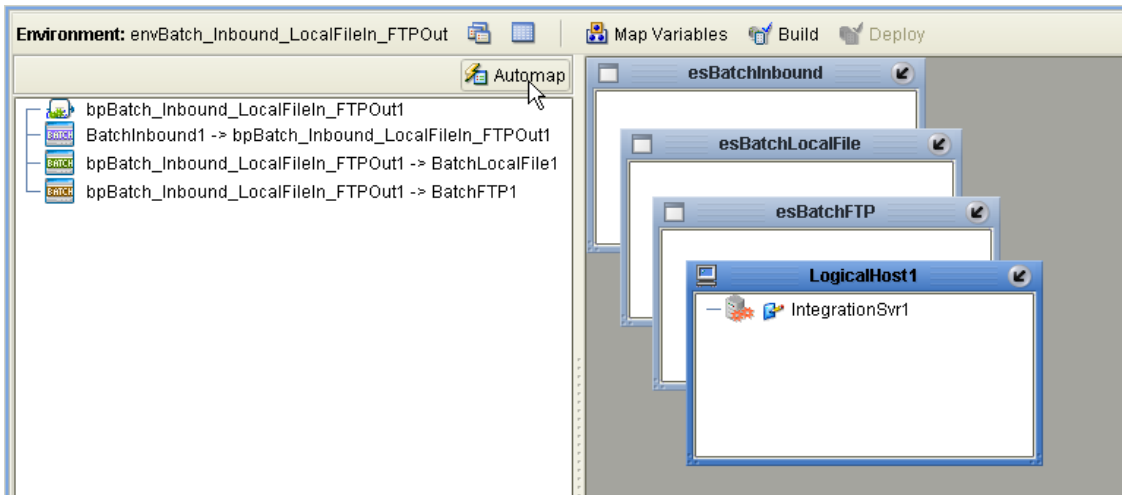
BatchLocalFile eWay Properties	
Target Location - Set as directed, otherwise use the default settings	
Target File Name	dummy *(this name is overwritten with the GUID file name in the Java Collaboration)*
Target Directory Name	X:/Batch51 *(where X is your target drive)*

7.5.9 Creating the Deployment Profile

A Deployment Profile is used to assign Collaborations and message destinations to the integration server and message server. Deployment Profiles are created using the Deployment Editor.

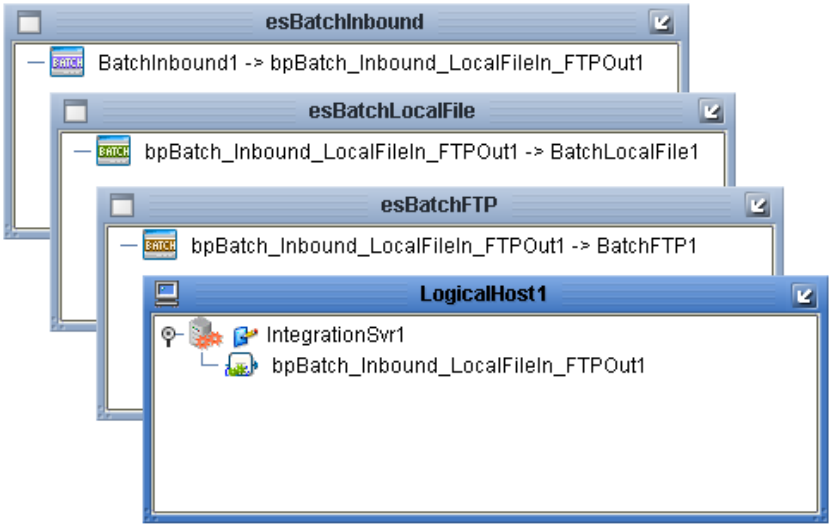
1. From the Enterprise Explorer's Project Explorer, right-click the Project (**prjBatch_Inbound_FTPIn_LocalFileOut**) and select **New** > **Deployment Profile**.
2. Enter a name for the Deployment Profile (for this sample **dpBatch_Inbound_FTPIn_LocalFileOut**). Make sure that the selected Environment is **envBatch_Inbound_FTPIn_LocalFileOut**. Click **OK**.
3. Click the **Automap** icon as displayed in Figure 39.

Figure 39 dpBatch_Inbound_FTPIn_LocalFileOut Deployment Profile



The Project's components are automatically mapped to their system window as seen in **Figure 40 on page 225**.

Figure 40 Completed Batch_Inbound_FTPIn_LocalFileOutDeployment Profile



4 If any of your Project components did not successfully map to an external system, open each of your eWay's configuration properties (Connectivity Map and Environment) and click **OK** to close and save the current configuration, then click **Automap** again.

5 Save your changes to the Repository.

7.5.10 Creating and Starting the Domain

To deploy your Project, you must first create a domain. A domain is an instance of a Logical Host. After the domain is created, the Project is built and then deployed.

Create and Start the Domain

1. Navigate to your **<*JavaCAPS51*>\logicalhost** directory (where <*JavaCAPS51*> is the location of your Sun Java Composite Application Platform Suite installation.
2. Double-click the **domainmgr.bat** file. The **Domain Manager** appears.
3. If you have already created a domain, select your domain in the Domain Manager and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.
4. If there are no existing domains, a dialog box indicates that you can create a domain now. Click **Yes**. The **Create Domain** dialog box appears.
5. Make any necessary changes to the **Create Domain** dialog box and click **Create**. The new domain is added to the Domain Manager. Select the domain and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.
6. For more information about creating and managing domains see the *Sun SeeBeyond eGate™ Integrator System Administration Guide*.

7.5.11 Building and Deploying the Project

The **Build** process compiles and validates the Project's Java files and creates the Project EAR file.

Build the Project

1. From the Deployment Editor toolbar, click the **Build** icon.
2. If there are any validation errors, a **Validation Errors** pane will appear at the bottom of the Deployment Editor and displays information regarding the errors. Make any necessary corrections and click **Build** again.
3. After the Build has succeeded you are ready to deploy your Project.

Deploy the Project

1. From the Deployment Editor toolbar, click the **Deploy** icon. Click **Yes** when the **Deploy** prompt appears.
2. A message appears when the project is successfully deployed.

Note: *Projects can also be deployed from the Enterprise Manager. For more information about using the Enterprise Manager to deploy, monitor, and manage your projects, see the Sun SeeBeyond eGate™ Integrator System Administration Guide.*

7.5.12 Running the Sample

To run your deployed sample Project do the following

1. Create any necessary directories (target, input, output, and so forth).
2. From your configured input directory, paste (or rename) the sample input file to trigger the eWay.

3 From your output directory, verify the output data.

7.6 The prjBatch_Streaming Sample Project

This section describes how the components of the prjBatch_Streaming sample Project are created. To create the Project manually, do the following:

7.6.1 Create a Project

The first step is to create and name a new Project in eGate Enterprise Designer.

1 Start the Enterprise Designer.

2 From the Enterprise Explorer's Project Explorer tab, select your Repository on the Project Explorer tree, right-click the Repository and select **New Project**. A new Project appears on the Project Explorer tree.

3 Rename the Project (for this sample, **prjBatch_Streaming**).

7.6.2 Creating a Java Collaboration Definition

The next step in the sample is to create a Java Collaboration using the Collaboration Definition Wizard (Java). Once a Collaboration Definition has been created, the Business Rules of the Collaboration are written using the Collaboration Editor.

The jcdBatch_Streaming Collaboration

1 From the Project Explorer, right-click the **prjBatch_Streaming** Project and select **New > Collaboration Definition (Java)** from the shortcut menu. The **Collaboration Definition Wizard (Java)** appears.

2 Enter a Collaboration Definition name (for this sample **jcdBatch_Streaming**) and click **Next**.

3 For Step 2 of the Wizard, from the Web Services Interfaces selection window, double-click **SeeBeyond** > **eWays** > **BatcheWay** > **BatchInbound** > **receive**. The File Name field now displays **receive.** Click **Next**.

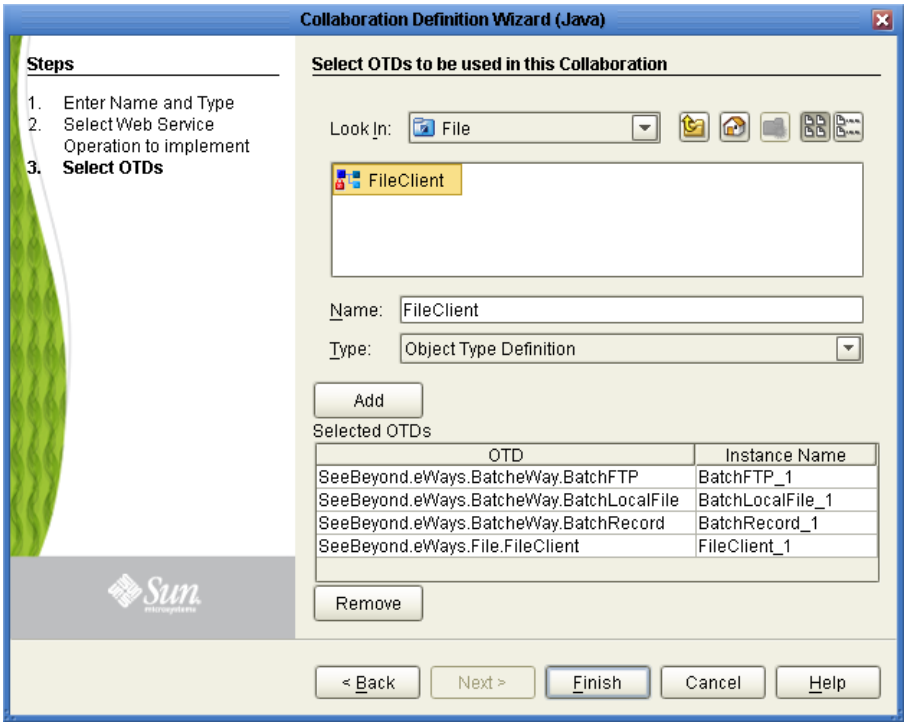
4 For Step 3 of the Wizard, from the Select OTDs selection window, double-click **SeeBeyond** > **eWays** > **BatcheWay**, and select the following three OTDs:

- **BatchFTP**
- **BatchLocalFile**
- **BatchRecord**

5 From the Select OTDs selection window, double-click **SeeBeyond** > **eWays** > **File**, and select the **FileClient** OTD.

The Selected OTDs field now displays the four OTDS (see Figure 41).

Figure 41 Collaboration Definition Wizard (Java) - Select OTDs



6 Click **Finish**. The Collaboration Editor (Java) opens to the new Collaboration in the right pane of the Enterprise Designer.

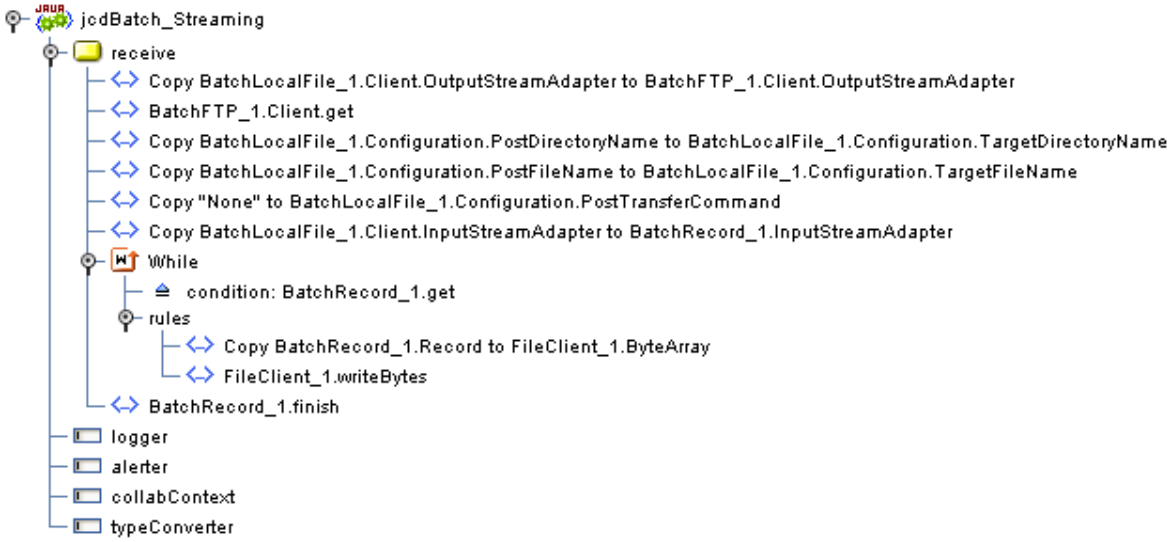
7.6.3 Using the Collaboration Editor (Java)

The **prjBatch_Streaming** Project uses the **jcdBatch_Streaming** Collaboration created in the previous section. To complete the Collaboration, use the Collaboration Editor to create the Business Rules.

Create the jcdBatch_Streaming Collaboration Business Rules

Be careful to open all nodes specified in the directions to connect the correct items. The **jcdBatch_Streaming** Collaboration contains the Business Rule displayed in Figure 42.

Figure 42 jcdBatch_Streaming Business Rules



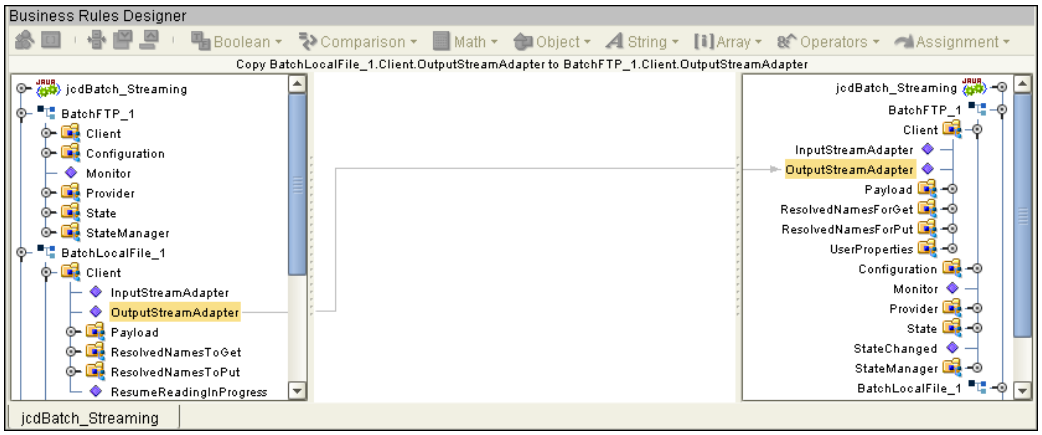
To create the **jcdBatch_Streaming** Collaboration Business Rules, do the following:

1. From the Project Explorer tree, double-click **jcdBatch_Streaming**. The Collaboration Editor (Java) opens to the **jcdBatch_Streaming** Collaboration.
2. To create comments for the Business Rules, from the Business Rules toolbar, click the **comment** icon. The **Enter a Comment** dialog box appears. Enter the comment and click **OK**. The comment is placed on the Business Rules tree under the last selected item. Once the Comment is created, it can be moved by clicking the comment and dragging it up or down the Business Rules tree to a new location.
3. The **Copy BatchLocalFile_1.Client.OutputStreamAdapter to BatchFTP_1.Client.OutputStreamAdapter** rule configures the BatchFTP and BatchLocalFile stream adapters so that the BatchFTP eWay does a stream get, and the BatchLocalFile eWay receives the stream.

   Create the **Copy BatchLocalFile_1.Client.OutputStreamAdapter to BatchFTP_1.Client.OutputStreamAdapter** rule:

   A. Map **OutputStreamAdapter** under **BatchLocalFile_1 > Client** in the left pane of the Business Rules Designer, to **OutputStreamAdapter** under **BatchFTP_1 > Client** in the right pane of the Business Rules Designer. To do this, click on **OutputStreamAdapter** in the left pane of the Business Rules Designer, and drag your cursor to the destination **OutputStreamAdapter** (see Figure 43).

Figure 43 Collaboration Editor (Java) - Business Rules



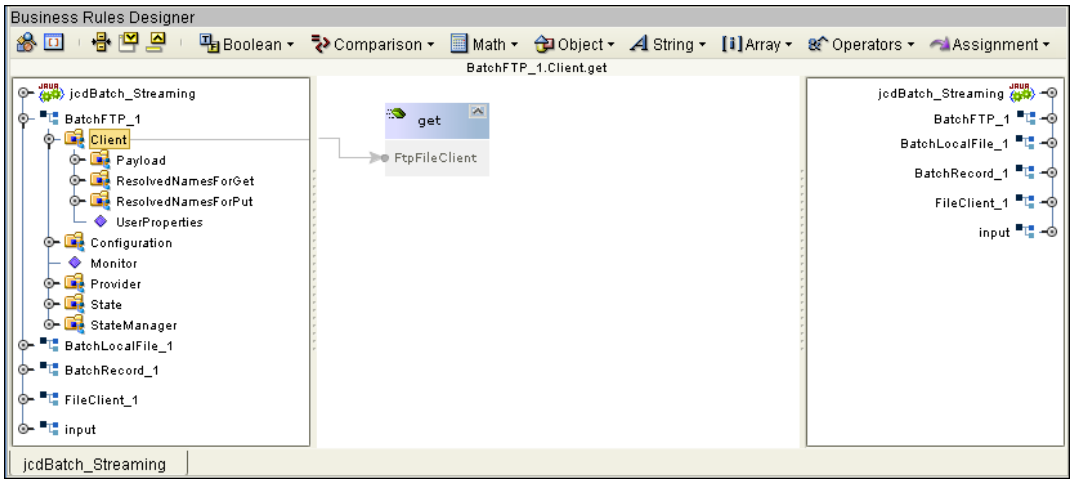
4 The **BatchFTP_1.Client.get** rule gets the file. Create the **BatchFTP_1.Client.get** rule:

A Click **rule** on the Business Rules toolbar to add a new **rule**.

B Right-click **Client** under **BatchFTP_1** in the left pane of the Business Rules Designer, and click **Select method to call** from the shortcut menu. The method selection window appears.

C Select **get()** from the method selection window. The **get** method box appears in the Business Rules Designer canvas (see Figure 44).

Figure 44 Collaboration Editor (Java) - Business Rules



5 The **Copy BatchLocalFile_1.Configuration.PostDirectoryName to BatchLocalFile_1.Configuration.TargetDirectoryName** rule dynamically configures the name of the target directory to the post directory name.

Create the **Copy BatchLocalFile_1.Configuration.PostDirectoryName to BatchLocalFile_1.Configuration.TargetDirectoryName** rule:

A Click **rule** on the Business Rules toolbar to add a new **rule**.

B Map **PostDirectoryName** under **BatchLocalFile_1 > Configuration** in the left pane of the Business Rules Designer, to **TargetDirectoryName** under **BatchLocalFile_1 > Configuration** in the right pane.

6 The **Copy BatchLocalFile_1.Configuration.PostFileName to BatchLocalFile_1.Configuration.TargetFileName** rule dynamically configures the name of the target file to the post file name.

Create the **Copy BatchLocalFile_1.Configuration.PostFileName to BatchLocalFile_1.Configuration.TargetFileName** rule:

A Click **rule** on the Business Rules toolbar to add a new **rule**.

B Map **PostFileName** under **BatchLocalFile_1 > Configuration** in the left pane of the Business Rules Designer, to **TargetFileName** under **BatchLocalFile_1 > Configuration** in the right pane of the Business Rules Designer.

7 The **Copy "None" to BatchLocalFile_1.Configuration.PostTransferCommand** Dynamically configures the post transfer command to None so that no post transfer will take place.

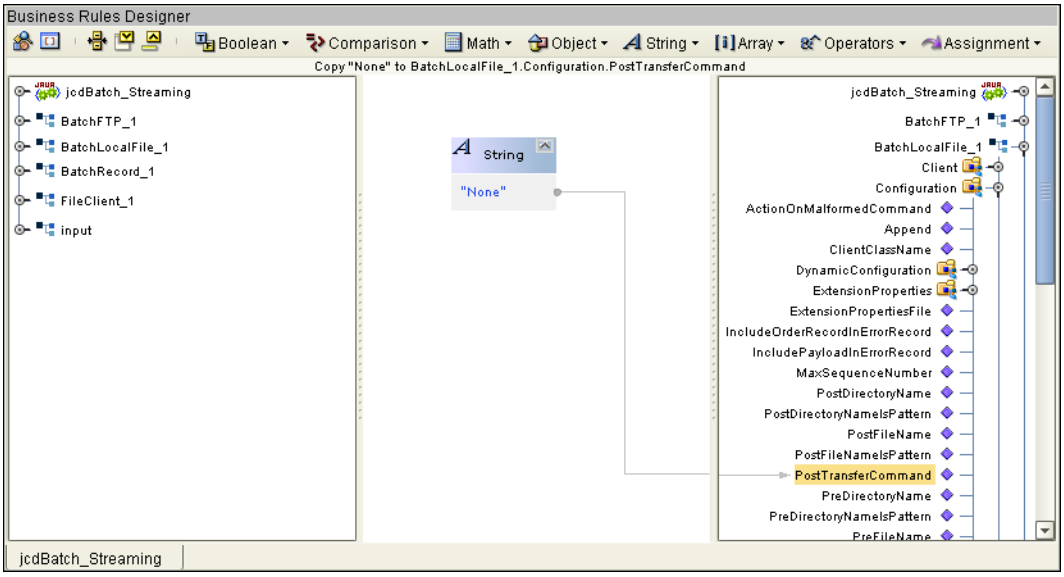
Create the **Copy "None" to BatchLocalFile_1.Configuration.PostTransferCommand** rule:

A From the Business Rules Designer's String menu, select **Literal String**. The **String** literal box appears.

B Double-click the value field of the String literal box and enter **None** as the value.

C Map the None output node of the String literal box to **PostTransferCommand** under **BatchLocalFile_1 > Configuration** in the right pane of the Business Rules Designer (see Figure 45).

Figure 45 Collaboration Editor (Java) - Business Rules



8 The **Copy BatchLocalFile_1.Client.InputStreamAdapter to BatchRecord_1.InputStreamAdapter** rule configures the BatchLocalFile and BatchRecord stream adapters so that the BatchLocalFile eWay does a stream get, and the BatchRecord eWay receives the stream.

Create the **Copy BatchLocalFile_1.Client.InputStreamAdapter to BatchRecord_1.InputStreamAdapter** rule:

A Click **rule** on the Business Rules toolbar to add a new **rule**.

B Map **InputStreamAdapter** under **BatchLocalFile_1 > Client** in the left pane of the Business Rules Designer, to **InputStreamAdapter** under **BatchRecord_1** in the right pane of the Business Rules Designer.

9 The **While** statement's Business Rules get the record, publishes it to the outbound File eWay, and the File eWay writes the file to a local directory.

Create the **While** statement:

A Click **While** on the Business Rules toolbar to add a new while statement in the Business Rules pane.

B From the Business Rules tree, expand the **While** statement and select the **condition**.

C From the Business Rules Designer, right-click **BatchRecord_1** in the left pane of the Business Rules Designer, and click **Select method to call** from the shortcut menu. The **method selection** window appears.

D Select **get()** from the method selection window. The **get** method box appears in the Business Rules Designer canvas.

E Map the **result (boolean)** output node of the **get** method box to **condition** in the right pane of the Business Rules Designer.

10 Create the **Copy BatchRecord_1.Record to FileClient_1.ByteArray** rule under the **while** statement:

A From the Business Rules tree, select **rules** under the **while** statement.

B Map **Record**, under **BatchRecord_1** in the left pane of the Business Rules Designer, to **ByteArray** under **FileClient_1** in the right pane of the Business Rules Designer.

11 To create the **FileClient_1.write** rule under the **while** statement do the following:

A Click **rule** on the Business Rules toolbar to add a new **rule** under the while statement.

B Right-click FileClient_1 in the left pane of the Business Rules Designer, and click **Select method to call** from the shortcut menu. The method selection window appears.

C Select **write()** from the method selection window. The write method box appears in the Business Rules Designer canvas.

12 To create the **BatchRecord_1.finish** rule do the following:

A Select the **while** statement on the Business Rules tree and click **rule** on the Business Rules toolbar. A new rule is added to the Business Rules tree.

B Right-click **BatchRecord_1** in the left pane of the Business Rules Designer, and click **Select method to call** from the shortcut menu. The method selection window appears.

C Select **finish()** from the method selection window. The **finish** method box appears.

13 From the editor's toolbar, click **Validate** to check the Collaboration for errors.

14 Save your changes to the Repository.

For more information on how to create Business Rules using the Collaboration Editor see the *Sun SeeBeyond eGate™ Integrator User's Guide*.

7.6.4 Create a Connectivity Map

The Connectivity Map provides a canvas for configuring a Project's components.

1 In Enterprise Explorer's Project Explorer, right-click the new Project (**prjBatch_Streaming**) and select **New > Connectivity Map** from the shortcut menu.

2 The New Connectivity Map appears and a node for the new Connectivity Map is added under the Project on the Project Explorer tree labeled **CMap1**. Rename the Connectivity Map **cmBatch_Streaming**.

Select the External Applications

1 Click the **External Application** icon on the Connectivity Map toolbar,

2 Select the following applications for your Project:

- BatchInbound External Application
- Service
- BatchFTP External Application
- BatchInbound External Application
- BatchLocalFile External System
- BatchRecord External System

Icons representing the selected applications are added to the Connectivity Map toolbar.

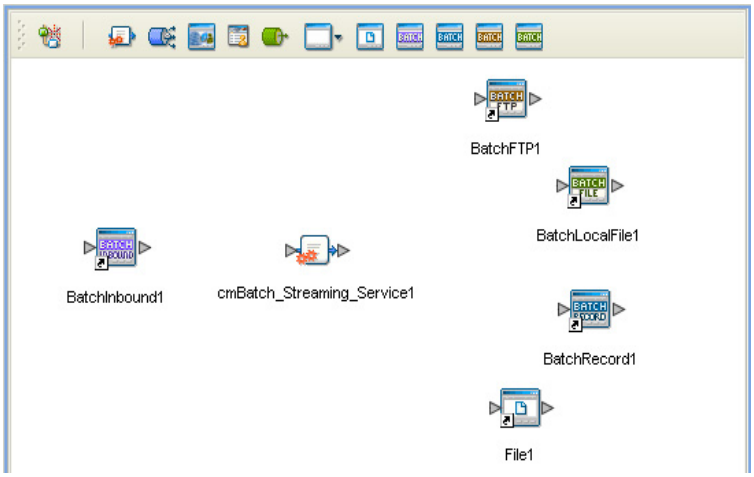
Populate the Connectivity Map

Add the Project components to the Connectivity Map by dragging the icons from the toolbar to the canvas.

1 For the prjBatch_Streaming Project, drag the following components onto the Connectivity Map canvas as displayed in **Figure 46 on page 234**:

- BatchInbound External Application (BatchInbound1)
- Service (cmBatch_Streaming_Service1)
- BatchFTP External Application (BatchFTP1)
- BatchLocalFile External Application (BatchLocalFile1)
- BatchRecord External Application (BatchRecord1)
- File External Application (File1)

Figure 46 cmBatch_Streaming Connectivity Map with Components



2 Rename the Service to **jcolBatch_Streaming1**.

3 Save your current changes to the Repository.

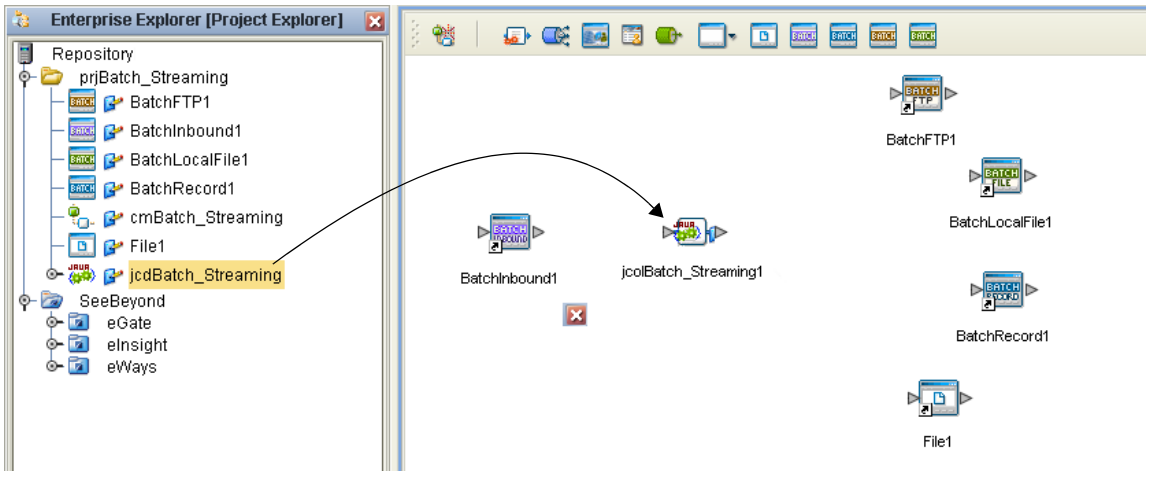
7.6.5 Binding the Project Components

The components are associated and the bindings are created in the Connectivity Map.

1 From the Project Explorer, double-click the Connectivity Map **cmBatch_Streaming** to display the **cmBatch_Streaming** Connectivity Map.

2 Drag and drop the **jcdBatch_Streaming** Collaboration from the Project Explorer tree onto the **jcolBatch_Streaming** service (see Figure 47).

Figure 47 Connectivity Map - Connecting the Project's Components



3 Double-click the **jcolBatch_Streaming** service. The **jcolBatch_Streaming** binding dialog box appears with **jcdBatch_Streaming** as the rule.

4 From the **jcolBatch_Streaming** binding box, map **BatchInbound1 input** (under Implemented Services) to the **BatchInbound1** application. A link is created between the two components.

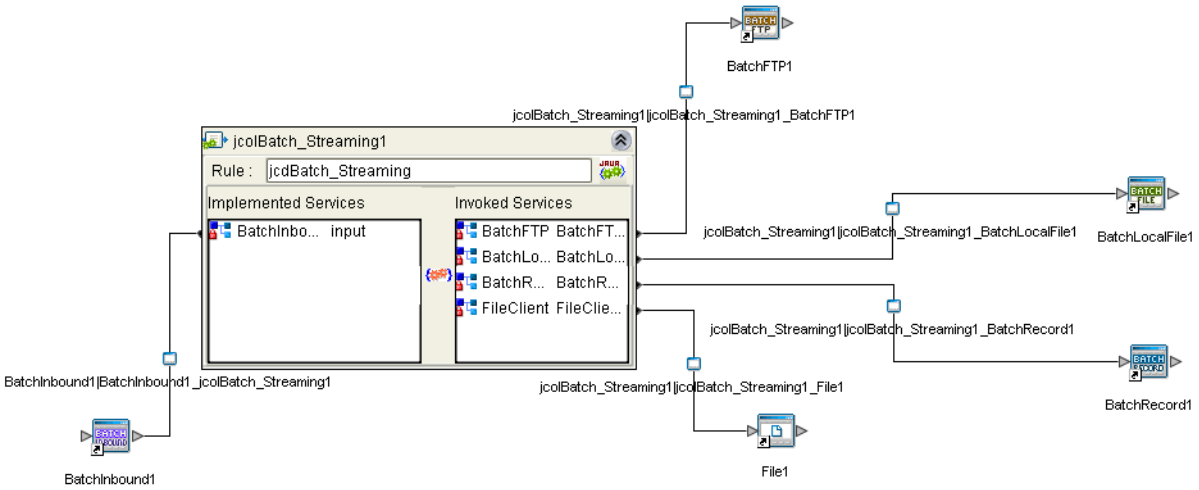
5 From the **jcolBatch_Streaming** binding box, map the **BatchFTP** (under Invoked Services) to the **BatchFTP1** External Application.

6 From the **jcolBatch_Streaming** binding box, map the **BatchLocalFile** (under Invoked Services) to the **BatchLocalFile1** External Application.

7 From the **jcolBatch_Streaming** binding box, map the **BatchRecord** (under Invoked Services) to the **BatchRecord1** External Application.

8 From the **jcolBatch_Streaming** binding box, map the **FileClient** (under Invoked Services) to the **File1** External Application (see Figure 48).

Figure 48 Connectivity Map - Connecting the Project's Components



9 Minimize the **jcolBatch_Streaming** binding box.

10 Save your current changes to your Repository.

7.6.6 Creating an Environment

Environments include the External Systems, Logical Hosts, Integration Servers and Message Servers used by a Project and contain the configuration information for these components.

1 From the Enterprise Designer's Enterprise Explorer, click the **Environment Explorer** tab.

2 Right-click the Repository and select **New Environment**. Rename the new Environment **envBatch_Streaming**.

3 Right-click the **envBatch_Streaming** Environment and select **New > BatchInbound External System**. Name this External System **esBatchInbound**. The **esBatchInbound** window is added to the Environment Editor.

4 Right-click **envBatch_Streaming** and select **New > BatchFTP External System**. Name this External System **esBatchFTP**.

5 Right-click **envBatch_Streaming** and select **New > BatchLocalFile External System**. Name the External System **esBatchLocalFile**.

6 Right-click **envBatch_Streaming** and select **New > BatchRecord External System**. Name the External System **esBatchRecord**.

7 Right-click **envBatch_Streaming** and select **New > File External System**. Name this External System **esFile**.

8 Right-click **envBatch_Streaming** and select **New Logical Host**. The **LogicalHost1 box** is added to the Environment and **LogicalHost1** is added to the Environment Editor tree.

9 From the Environment Explorer tree, right-click **LogicalHost1** and select **New Sun Integration Server**. A new Integration Server (**IntegrationSvr1**) is added to the Environment Explorer tree under **LogicalHost1**.

10 Save your current changes to the Repository.

7.6.7 Configuring the eWay Properties

The **prjBatch_Streaming** Project uses five eWays, each represented in the Connectivity Map as a node between an External Application and a Service.

The eWay properties are configured from the Connectivity Map and the Environment. To configure the eWays do the following:

Configuring the BatchInbound eWay Properties

1 From the Connectivity Map, double-click the **BatchInbound eWay**. The eWay Properties Editor appears.

2 Modify the properties for your system, including the settings for the **BatchInbound eWay** in Table 78, and click **OK**. The new eWay properties are saved.

Table 78 BatchInbound eWay Connectivity Map Properties
Table 79

BatchInbound eWay Connectivity Map Properties	
Settings - Set as directed, otherwise use the default settings	
Directory Name	*Your input directory*
File Name	BatchInbound.txt

3 The BatchInbound eWay Environment properties can keep the default settings for this sample.

Configuring the BatchFTP eWay Properties

The BatchFTP eWay properties are set from both the Connectivity Map and Environment Explorer. For more information on the BatchFTP eWay properties and the Properties Editor, see **"Creating and Configuring Batch eWays" on page 27** or see the *Sun SeeBeyond eGate™ Integrator User's Guide*.

Modify the BatchFTP eWay Connectivity Map Properties

1 From the Connectivity Map, double-click the **BatchFTP** eWay. The eWay Properties Editor appears.

2 Modify the **BatchFTP** eWay Connectivity Map properties for your system, including the settings in Table 80, and click **OK**.

Table 80 BatchFTP eWay Connectivity Map Properties
Table 81

BatchFTP eWay Connectivity Map Properties	
Target Location - Set as directed, otherwise use the default settings	
Target Directory Name	*The directory (absolute path) from which files are retrieved*
Target File Name	BatchFTPIn.txt

Modify the BatchFTP eWay (Environment Explorer) Properties

1 From the **Environment Explorer** tree, right-click the BatchFTP External System (**esBatchFTP** in this sample), and select **Properties**. The Properties Editor opens to the BatchFTP eWay Environment properties.

2 Modify the BatchFTP eWay Environment properties for your system, including the settings in Table 82, and click **OK**.

Table 82 BatchFTP eWay Environment Explorer Properties
Table 83

BatchFTP eWay Environment Properties	
FTP - Set as directed, otherwise use the default settings.	
Host Name	localhost
Server Port	*Port number to use to connect to the FTP server*
User Name	*User ID used to log into the system*
Password	*Password required to log into the system*

Configuring the BatchLocalFile eWay Properties

The **BatchLocalFile** eWay's specific properties are set from the Connectivity Map. Environment properties can use the default settings.

1 Double-click the **BatchLocalFile eWay**. The eWay Properties Editor appears.

2 Modify the properties for your system, including the settings in Table 84, and click **OK**. The properties are saved for the eWay.

Table 84 BatchLocalFile eWay Properties
Table 85

BatchLocalFile eWay Properties
Post Transfer - Set as directed, otherwise use the default settings

Table 85

BatchLocalFile eWay Properties (Continued)	
Post Directory Name	X:/Batch51/ready
Post File Name	BatchLocalFileReady.txt
Target Location - Set as directed, otherwise use the default settings	
Target File Name	*The name of the file to be sent.*
Target Directory Name	*The directory on the system where files are sent.*

Configuring the BatchRecord eWay Properties

The **BatchRecord** eWay's specific properties are set from the Connectivity Map. Environment properties can use the default settings.

1. From the Connectivity Map, double-click the BatchRecord eWay. The Properties Sheet opens to the eWay Connectivity Map properties.
2. Modify the configuration for your system, including the settings in Table 86).

Table 86 BatchRecord eWay Properties
Table 87

BatchRecord eWay Connectivity Map Properties	
General Settings - Set as directed, otherwise use the default settings	
Parse or Create Mode	Parse
Record - Set as directed, otherwise use the default settings	
Delimiter on Last Record	Yes
Record Delimiter	\n
Record Type	Delimited

Configuring the File eWay Properties

Modify the outbound File eWay Connectivity Map properties

1. From the Connectivity Map, double-click the outbound **File1 eWay**.
2. The Properties Editor opens to the outbound File eWay properties. Modify the configuration for your system, including the settings in Table 88.

Table 88 Outbound File eWay Properties
Table 89

Outbound File eWay Connectivity Map Properties	
Parameter Settings - Set as directed, otherwise use the default settings	
Output File Name	FileOutput%d.txt

Modify the File eWay Environment properties

1. From the **Environment Explorer** tree, right-click the **esFile** External System, and select **Properties**. The Properties Editor opens to the File eWay Environment properties.
2. Modify the File eWay Environment properties for your system, including the settings in Table 90, and click **OK**.

Table 90 File eWay Environment Properties
Table 91

File eWay Environment Properties	
Parameter Settings - Set as directed, otherwise use the default settings.	
Directory	*The directory on the system where files are sent.*

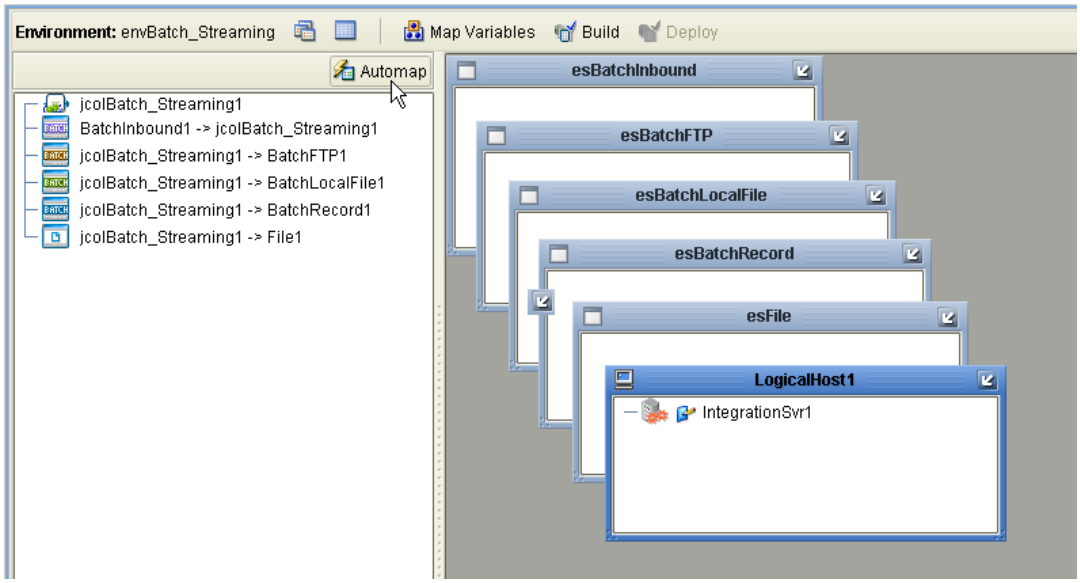
For more information on the Batch eWay properties and the Properties Editor, see **"Creating and Configuring Batch eWays" on page 27**.

7.6.8 Creating the Deployment Profile

A Deployment Profile is used to assign Collaborations and message destinations to the integration server and message server. Deployment Profiles are created using the Deployment Editor.

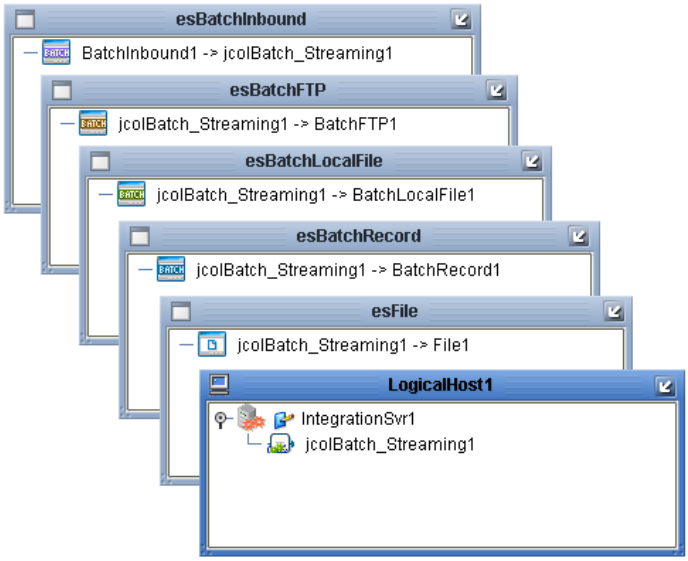
1. From the Enterprise Explorer's Project Explorer, right-click the Project (**prjBatch_Streaming**) and select **New** > **Deployment Profile**.
2. Enter a name for the Deployment Profile (for this sample **dpBatch_Streaming**). Make sure that the selected Environment is **envBatch_Streaming**. Click **OK**.
3. Click the **Automap** icon as displayed in Figure 49.

Figure 49 dpBatch_Inbound_FTPIn_LocalFileOut Deployment Profile



The Project's components are automatically mapped to their system window as seen in **Figure 50 on page 240**.

Figure 50 Completed dpBatch_Inbound_FTPIn_LocalFileOut Deployment Profile



4 If any of your Project components did not successfully map to an external system, open each of your eWay's configuration properties (Connectivity Map and Environment) and click **OK** to close and save the current configuration, then click **Automap** again.

5 Save your changes to the Repository.

7.6.9 Creating and Starting the Domain

To deploy your Project, you must first create a domain. A domain is an instance of a Logical Host. After the domain is created, the Project is built and then deployed.

Create and Start the Domain

1 Navigate to your **<*JavaCAPS51*>\logicalhost** directory (where <*JavaCAPS51*> is the location of your Sun Java Composite Application Platform Suite installation.

2 Double-click the **domainmgr.bat** file. The **Domain Manager** appears.

3 If you have already created a domain, select your domain in the Domain Manager and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

4 If there are no existing domains, a dialog box indicates that you can create a domain now. Click **Yes**. The **Create Domain** dialog box appears.

5 Make any necessary changes to the **Create Domain** dialog box and click **Create**. The new domain is added to the Domain Manager. Select the domain and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

6 For more information about creating and managing domains see the *Sun SeeBeyond eGate™ Integrator System Administration Guide*.

7.6.10 Building and Deploying the Project

The **Build** process compiles and validates the Project's Java files and creates the Project EAR file.

Build the Project

1 From the Deployment Editor toolbar, click the **Build** icon.

2 If there are any validation errors, a **Validation Errors** pane will appear at the bottom of the Deployment Editor and displays information regarding the errors. Make any necessary corrections and click **Build** again.

3 After the Build has succeeded you are ready to deploy your Project.

Deploy the Project

1 From the Deployment Editor toolbar, click the **Deploy** icon. Click **Yes** when the **Deploy** prompt appears.

2 A message appears when the project is successfully deployed. You can now test your sample.

7.6.11 Running the Sample

To run your deployed sample Project do the following

1 Create any necessary directories (target, input, output, and so forth).

2 From your configured input directory, paste (or rename) the sample input file to trigger the eWay.

3 From your output directory, verify the output data.

Note: *For UNIX, be sure to upload the appropriate LogicalHost*

7.7 The prjBatch_Record Sample Project

The **prjBatch_Record** Project demonstrates the following:

1. The inbound FileIn eWay polls the input directory periodically. When it sees the specified file, it renames the file by adding **.~in** to the end, and triggers the Java Collaboration.
2. The Collaboration generates a String. This String is converted into a record using BatchRecordCreate. The record is appended to the payload of BatchRecordCreate.
3. The operation in Step 2 is repeated 10 times, resulting in 10 records in the BatchRecordCreate payload.
4. The BatchRecordCreate payload is copied to the FileOut payload.
5. The outbound FileOut eWay writes the payload to an output file.
6. The Collaboration takes the BatchRecordCreate payload, and uses BatchRecordParse to parse the records.
7. Each parsed record is copied to the FileOut payload, and the FileOut eWay writes the payload to the output file.
8. The operation in Step 7 is repeated until all 10 records are retrieved.

The following pages provide step by step directions for manually creating the **prjBatch_Record** Project components.

7.7.1 Create a Project

The first step is to create and name a new Project in eGate Enterprise Designer.

1. Start the Enterprise Designer.
2. From the Enterprise Explorer's Project Explorer tab, select your Repository on the Project Explorer tree, right-click the Repository and select **New Project**. A new Project appears on the Project Explorer tree.
3. Rename the Project (for this sample, **prjBatch_Record**).

7.7.2 Creating a Java Collaboration Definition

The next step in the sample is to create a Java Collaboration using the Collaboration Definition Wizard (Java). Once a Collaboration Definition has been created, the Business Rules of the Collaboration are written using the Collaboration Editor.

Create the jcdBatch_Record Collaboration

1. From the Project Explorer, right-click the **prjBatch_Record** Project and select **New > Collaboration Definition (Java)** from the shortcut menu. The **Collaboration Definition Wizard (Java)** appears.

2 Enter a Collaboration Definition name (for this sample **jcdBatch_Record**) and click **Next**.

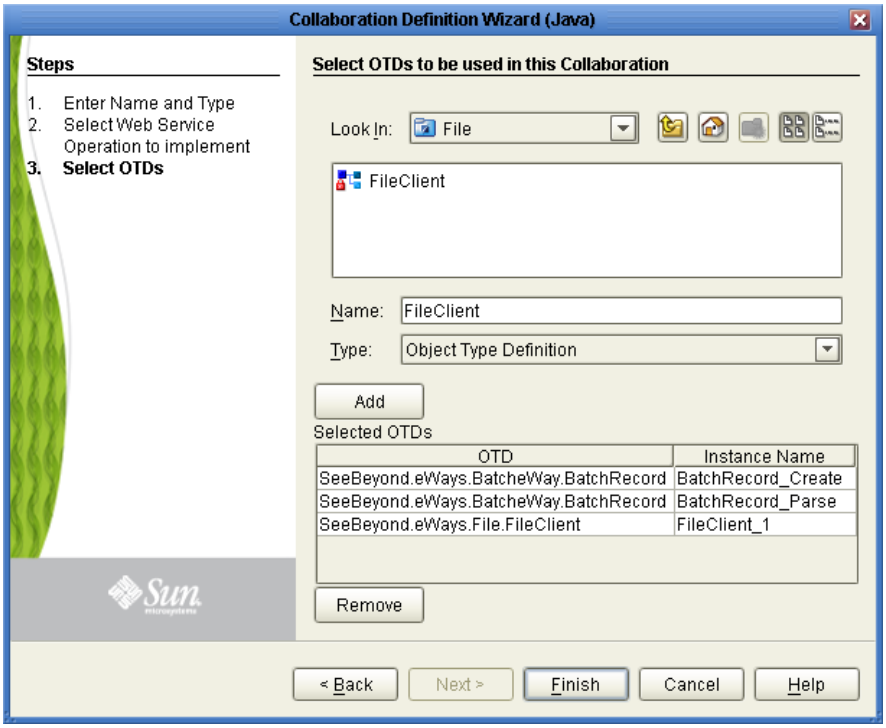
3 For Step 2 of the Wizard, from the Web Services Interfaces selection window, double-click **SeeBeyond** > **eWays** > **File** > **FileClient** > **receive**. The Name field now displays **receive.** Click **Next**.

4 For Step 3 of the Wizard, from the Select OTDs selection window, double-click **SeeBeyond** > **eWays** > **BatcheWay** > **BatchRecord**. The **BatcheWay.BatchRecord** OTD is added to the Selected OTDs field. Double-click the **BatchRecord_1** Instance Name and rename the instance to **BatchRecord_Create**.

5 From the Select OTDs selection window, double-click **SeeBeyond** > **eWays** > **BatcheWay** > **BatchRecord** again. Another **BatcheWay.BatchRecord** OTD is added to the Selected OTDs field. Double-click the **BatchRecord_2** Instance Name and rename the instance to **BatchRecord_Parse**.

6 Click the **Up One Level** button to return to the Repository directory. Double-click **SeeBeyond** > **eWays** > **File** > **FileClient**. The **FileClient** OTD is added to the Selected OTDs field (see Figure 51).

Figure 51 Collaboration Definition Wizard (Java) - Select OTDs



7 Click **Finish**. The Collaboration Editor (Java) opens to the new Collaboration in the right pane of the Enterprise Designer.

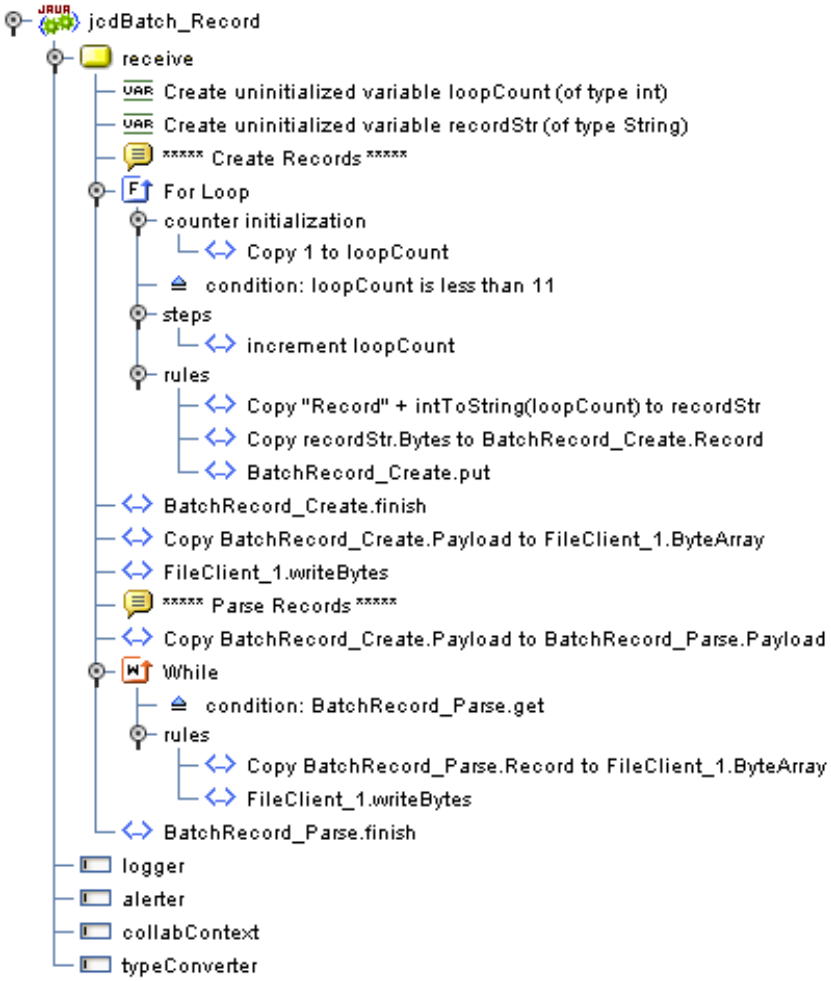
7.7.3 Using the Collaboration Editor (Java)

The **prjBatch_Record** Project uses the **jcdBatch_Record** Collaboration created in the previous section. To complete the Collaboration, use the Collaboration Editor to create the Business Rules.

Create the jcdBatch_Record Collaboration Business Rules

Be careful to open all nodes specified in the directions to connect the correct items. The **jcdBatch_Record** Collaboration contains the Business Rule displayed in **Figure 52 on page 244**.

Figure 52 **jcdBatch_Record** Business Rules



To create the **jcdBatch_Record** Collaboration Business Rules, do the following:

1. From the Project Explorer tree, double-click **jcdBatch_Record**. The Collaboration Editor (Java) opens to the **jcdBatch_Record** Collaboration.
2. To create comments for the Business Rules, from the Business Rules toolbar, click the **comment** icon. The **Enter a Comment** dialog box appears. Enter the comment and click **OK**. The comment is placed on the Business Rules tree under the last selected item. Once the Comment is created, it can be moved by clicking the comment and dragging it up or down the Business Rules tree to a new location.
3. The **Create uninitialized variable loopCount (of type int)** rule defines a loop count integer variable. Create the **Create uninitialized variable loopCount (of type int)** variable:

A From the Business Rules toolbar, click the **Local Variable** icon. The **Create Variable** dialog box appears.

B Enter **loopCount** as the name of the variable. For Type, select **Primitive** of type **int**, and click **OK**. The new variable is added to the Business Rules tree.

4 The **Create uninitialized variable recordStr (of type String)** rule defines String called recordStr. Create the **Create uninitialized variable recordStr (of type String)** variable:

A From the Business Rules toolbar, click the **Local Variable** icon. The **Create Variable** dialog box appears.

B Enter **recordStr** as the name of the variable.

C From the **Create Variable** dialog box Type field, select **Class** and click the ellipsis (...) button. The **Class Browser** dialog box appears.

D From the **Class Browser** dialog box, select **String** in the **All Classes** field, and select **String** in the **String** field. Click **Select** to close the **Class Browser** dialog box and click **OK** to close the **Create Variable** dialog box. The new variable is added to the Business Rules tree.

5 The **For Loop** and its Business Rules create ten records, each incrementing by 1 (as a String). It does a put for each incremented record until the payload contains ten records. Create the **For Loop**:

A From the Business Rules toolbar, click the **For Loop** icon. A **For Loop** is added to the Business Rules tree.

B From the Business Rules tree, select the **counter initialization** under the **For Loop**.

C From the Business Rules Designer's **Math** menu, select **Literal Number**. The Number literal box appears. Double-click the value field of the **Number** literal box and enter **1** as the Literal Number value.

D Map the **1** output node of the **Number** literal box to the **loopCount** variable in the right pane of the Business Rules Designer.

6 Create the **condition: loopCount is less than 11** under the For Loop:

A From the Business Rules tree, select the **condition** under the **For Loop**.

B From the Business Rules Designer's **Comparison** menu, select **Less Than**. The **Less Than** method box appears. Double-click the **number2** value field of the **Less Than** method box and enter **11** as the value.

C Map the **loopCount** variable in the left pane of the Business Rules Designer, to the **number1** input node of the **Less Than** method box.

D Map the **result(boolean)** output node of the **Less Than** method box, to condition variable in the right pane of the Business Rules Designer.

7 Create the **increment loopCount** rule under the For Loop steps:

A From the Business Rules tree, select **steps** under the For Loop.

B From the Business Rules Designer's **Math** menu, select **Increment**. The Increment method box appears.

C Map the **loopCount** variable in the left pane of the Business Rules Designer, to the **number** input node of the **Increment** method box.

8 Create the **Copy "Record" + intToString(loopCount) to recordStr** rule under For Loop rules:

A From the Business Rules tree, select **rules** under the **For** Loop.

B Map the **loopCount** variable in the left pane of the Business Rules Designer, to the **recordStr** variable in the right pane of the Business Rules Designer. The **Number to String Conversion** dialog box appears. Click **OK** to accept the default values. The **intToString** method box appears.

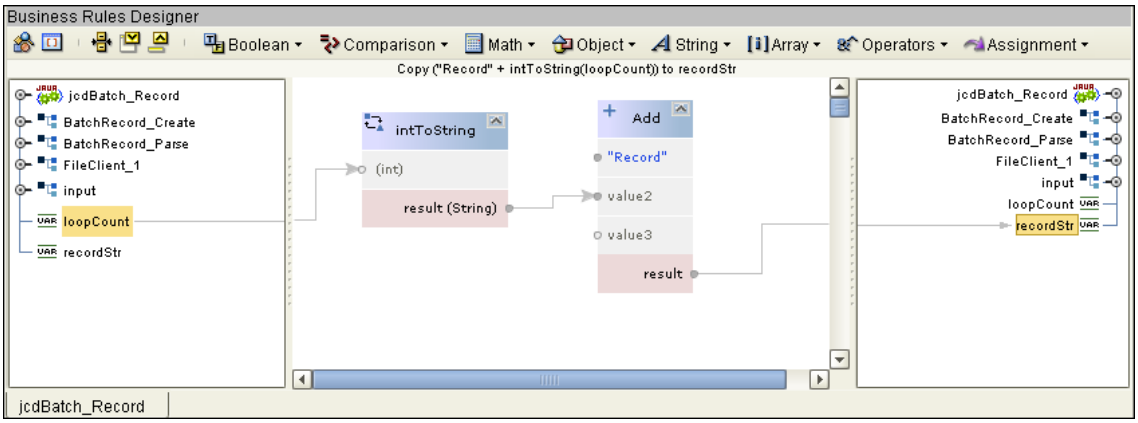
C From the Business Rules Designer canvas, click on the link between the **result(String)** output node of the **intToString** method box and the **recordStr** variable in the right pane of the Business Rules Designer. With the link selected, hit your **Delete** key to break the link.

D From the Business Rules Designer's **String** menu, select **Add**. The **Add** method box appears. Double-click the **value1** value field of the **Add** method box and enter **Record** as the value.

E Map the **result(String)** output node of the **intToString** method box, to the **value2** input node of the **Add** method box.

F Map the **result** output node of the **Add** method box, to the **recordStr** variable in the right pane of the Business Rules Designer (see Figure 53).

Figure 53 Collaboration Editor (Java) - Business Rules



9 Create the **Copy recordStr.Bytes to BatchRecord_Create.Record** rule under **FOR Loop** rules:

A Right-click the **recordStr** variable in the left pane of the Business Rules Designer and select **Browse this type** from the shortcut menu. The **Class Browser** dialog box appears.

B From the **Class Browser** dialog box, select **String** in the **All Classes** field and **getBytes()** in the **String** field. Click **Select**. The getBytes method box appears.

C Map the **result(byte[])** output node of the **getBytes** method box, to **Record** under **BatchRecord_Create** in the right pane of the Business Rules Designer.

10 Create the **BatchRecord_Create.put** rule under FOR Loop rules:

A Right-click **BatchRecord_Create** in the left pane of the Business Rules Designer and select **Select method to call** from the shortcut menu. The method selection window appears.

B Select **put()** from the method selection window. The **put** method box appears.

11 Create the **BatchRecord_Create.finish** rule:

A From the Business Rules tree, select the **For Loop,** and from the Business Rules toolbar click the **rule** icon to create a new rule.

B Right-click **BatchRecord_Create** in the left pane of the Business Rules Designer, and choose **Select method to call** from the shortcut menu. The method selection window appears.

C Select **finish()** from the method selection window. The **finish** method box appears in the Business Rules Designer canvas.

12 The **Copy BatchRecord_Create.Payload to FileClient_1.ByteArray** sets the ByteArray of the File eWay to the payload. Create the **Copy BatchRecord_Create.Payload to FileClient_1.ByteArray** rule:

A Click **rule** on the Business Rules toolbar to add a new rule in the Business Rules pane.

B Map **Payload** under **BatchRecord_Create** in the left pane of the Business Rules Designer, to **ByteArray** under **FileClient_1** in the right pane of the Business Rules Designer.

13 The **FileClient_1.writeBytes** writes the ten records to the payload. Create the **FileClient_1.writeBytes** rule:

A Click **rule** on the Business Rules toolbar to add a new rule in the Business Rules pane.

B Right-click **FileClient_1** in the left pane of the Business Rules Designer, and click **Select method to call** from the shortcut menu. The method selection window appears.

C Select **writeBytes()** from the method selection window. The **writeBytes** method box appears.

14 The **Copy BatchRecord_Create.Payload to BatchRecord_Parse.Payload** rule copies the payload to the BatchRecord_Parse eWay. Create the **Copy BatchRecord_Create.Payload to BatchRecord_Parse.Payload** rule:

A Click **rule** on the Business Rules toolbar to add a new rule in the Business Rules pane.

B Map **Payload** under **BatchRecord_Create** in the left pane of the Business Rules Designer, to **Payload** under **BatchRecord_Parse** in the right pane of the Business Rules Designer.

15 The **While** statement and its Business Rules parse the ten records and publishes each record to the File eWay. Create the **While** statement:

A From the Business Rules toolbar, click the **While** icon. A **While statement** is added to the Business Rules tree.

B From the Business Rules tree, select **condition** under the **While** statement.

C Right-click **BatchRecord_Parse** in the left pane of the Business Rules Designer, and click **Select method to call** from the shortcut menu. The method selection window appears.

D Select **get()** from the method selection window. The **get** method box appears.

E Map the **result(boolean)** output node of the **get** method box to **condition** in the right pane of the Business Rules Designer.

16 Create the **Copy BatchRecord_Parse.Record to FileClient_1.ByteArray** rule under the While statement rules:

A From the Business Rules tree, select **rules** under the **While** statement.

B Map **Record** under **BatchRecord_Parse** in the left pane of the Business Rules Designer, to **ByteArray** under **FileClient_1** in the right pane of the Business Rules Designer.

17 Create the **FileClient_1.writeBytes** rule under the **While** statement rules:

A Click **rule** on the Business Rules toolbar to add a new rule in the Business Rules pane.

B Right-click **FileClient_1** in the left pane of the Business Rules Designer, and click **Select method to call** from the shortcut menu. The method selection window appears.

C Select **writeBytes()** from the method selection window. The **writeBytes** method box appears.

18 Create the **BatchRecord_Parse.finish** rule

A From the Business Rules tree, select the **While** statement, and from the Business Rules toolbar click the **rule** icon to create a new rule.

B Right-click **BatchRecord_Parse** in the left pane of the Business Rules Designer, and choose **Select method to call** from the shortcut menu. The method selection window appears.

C Select **finish()** from the method selection window. The **finish** method box appears in the Business Rules Designer canvas.

19 From the editor's toolbar, click **Validate** to check the Collaboration for errors.

20 Save your current changes to the Repository.

Note: *See the Sun SeeBeyond eGate™ Integrator User's Guide for more information on editing Collaborations.*

7.7.4 Create a Connectivity Map

The Connectivity Map provides a canvas for configuring a Project's components.

1. In Enterprise Explorer's Project Explorer, right-click the new Project (**prjBatch_Record**) and select **New > Connectivity Map** from the shortcut menu.
2. The New Connectivity Map appears and a node for the Connectivity Map is added under the Project, on the Project Explorer tree, labeled **CMap1**.
3. Rename the Connectivity Map to **cmBatch_Record**.

Select the External Applications

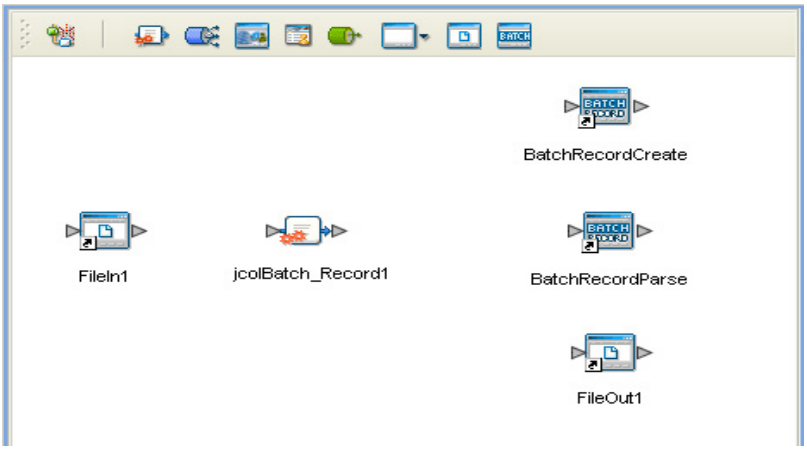
1. Click the **External Application** icon on the Connectivity Map toolbar,
2. Select the following applications for your Project:
   - BatchRecord External Application
   - File External System
3. Icons representing the selected applications are added to the Connectivity Map toolbar.

Populate the Connectivity Map

Add the Project components to the Connectivity Map by dragging the icons from the toolbar to the canvas.

1. For the prjBatch_Record Project, drag the following components onto the Connectivity Map canvas, and name each as displayed in Figure 46:
   - File External Application (rename to **FileIn1**)
   - Service (rename to **jcolBatch_Record1**)
   - BatchRecord External Application (rename to **BatchRecordCreate**)
   - BatchRecord External System (rename to **BatchRecordParse**)
   - File External Application (rename to **FileOut1**)

Figure 54 cmBatch_Record Connectivity Map with Components



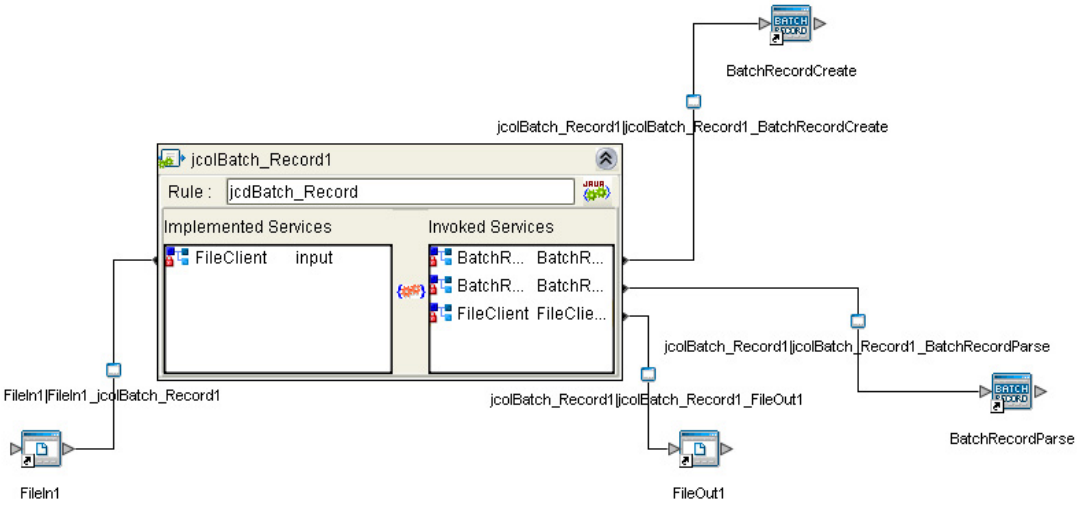
Save your current changes to the Repository.

7.7.5 Binding the Project Components

The components are associated and the bindings are created in the Connectivity Map.

1. From the Project Explorer, double-click **cmBatch_Record** to display the Project's Connectivity Map.
2. Drag and drop the **jcdBatch_Record** Collaboration from the Project Explorer tree onto the **jcolBatch_Record** service. When the Collaboration is successfully associated with the service, the service icon's "gears" turn from red to green.
3. Double-click the **jcolBatch_Record** service. The **jcolBatch_Record** binding dialog box appears with **jcdBatch_Record** as the rule.
4. From the **jcolBatch_Record** binding box, map **FileClient input** (under Implemented Services) to the **FileIn1** application. A link is created between the two components.
5. From the **jcolBatch_Record** binding box, map the **BatchRecord_Create, BatchRecord** (under Invoked Services) to the **BatchRecordCreate** External Application.
6. From the **jcolBatch_Record** binding box, map the **BatchRecord_Parse, BatchRecord** (under Invoked Services) to the **BatchRecordParse** External Application.
7. From the **jcolBatch_Record** binding box, map the **FileClient_1 FileClient** (under Invoked Services) to the **FileOut1** External Application (see Figure 55).

Figure 55 Connectivity Map - Connecting the Project's Components



8 Minimize the **jcolBatch_Record** binding box.

9 Save your current changes to your Repository.

7.7.6 Creating an Environment

Environments include the External Systems, Logical Hosts, Integration Servers and Message Servers used by a Project and contain the configuration information for these components.

1 From the Enterprise Designer's Enterprise Explorer, click the **Environment Explorer** tab.

2 Right-click the Repository and select **New Environment**. Rename the new Environment **envBatch_Record**.

3 Right-click the **envBatch_Record** Environment and select **New > File External System**. Name this External System **esFile**. The **esFile** window is added to the Environment Editor.

4 Right-click **envBatch_Record** and select **New > BatchRecord External System**. Name this External System **esBatchRecord**.

5 Right-click **envBatch_Record** and select **New Logical Host**. The **LogicalHost1** box is added to the Environment and **LogicalHost1** is added to the Environment Editor tree.

6 From the Environment Explorer tree, right-click **LogicalHost1** and select **New > Sun SeeBeyond Integration Server**. A new Integration Server (**IntegrationSvr1**) is added to the Environment Explorer tree under **LogicalHost1**.

7 Save your current changes to the Repository.

7.7.7 Configuring the eWay Properties

The **prjBatch_Record** Project uses four eWays, each represented in the Connectivity Map as a node between an External Application and a Service. For a description of the e

eWay properties and the Properties Editor, see **"Creating and Configuring Batch eWays" on page 27**. The eWay's properties are configured from both the Connectivity Map and the Environment Explorer. To configure the eWays do the following:

Configuring File eWay Connectivity Map Properties

The inbound File eWay properties are set from the Project Explorer's Connectivity Map.

1. Double-click the inbound **FileIn1 eWay, located between the FileIn1** External Application and the **jcdBatch_Record1** service.
2. The Properties Editor opens to the inbound File eWay properties. Modify the configuration for your system, including the settings in Table 92).

Table 92 Inbound File Connectivity Map Properties
Table 93

Inbound File eWay Connectivity Map Properties	
Parameter Settings - Set as directed, otherwise use the default settings	
Input File Name	FileInput.txt

3. From the Connectivity Map, modify the outbound **FileOut1** eWay configuration for your system, including the settings in Table 94).

Table 94 Outbound FileOut1 eWay Properties
Table 95

Outbound File eWay Connectivity Map Properties	
Parameter Settings - Set as directed, otherwise use the default settings	
Output File Name	FileOutput.txt

Modify the File eWay Environment properties

1. From the **Environment Explorer** tree, right-click the **esFile** External System, and select **Properties**. The Properties Editor opens to the File eWay Environment properties.
2. Modify the File eWay Environment properties for your system, including the settings in Table 96, and click **OK**.

Table 96 esFile eWay Environment Properties
Table 97

File eWay Environment Properties	
Inbound File eWay > Parameter Settings - Set as directed, otherwise use the default settings.	
Directory	*An input directory on your system that the inbound File eWays poll and pick up any specified input files.*
Outbound File eWay > Parameter Settings - Set as directed, otherwise use the default settings.	

Table 97

File eWay Environment Properties (Continued)	
Directory	*An output directory to which the outbound File eWay publishes (writes) the output messages.*

Configuring the BatchRecord eWay Properties

Modify the BatchRecordCreate eWay Connectivity Map properties

The **BatchRecord_Create** eWay's specific properties are set from the Connectivity Map.

1. From the **Connectivity Map**, double-click the eWay. The Properties Sheet opens to the eWay's Connectivity Map properties.
2. Modify the configuration for your system, including the settings in Table 98).

Table 98 BatchRecordCreate eWay Connectivity Map Properties
Table 99

BatchRecordCreate eWay Connectivity Map Properties	
General Settings - Set as directed, otherwise use the default settings	
Parse or Create Mode	Create
Record - Set as directed, otherwise use the default settings	
Delimiter on Last Record	No
Record Delimiter	~
Record Type	Delimited

Modify the BatchRecordParse eWay Connectivity Map properties

1. From the **Connectivity Map**, double-click the BatchRecordParse eWay. The Properties Editor opens to the eWay's Connectivity Map properties.
2. Modify the configuration for your system, including the settings in Table 100).

Table 100 BatchRecordParse eWay Connectivity Map Properties
Table 101

BatchRecordParse Connectivity Map Properties	
General Settings - Set as directed, otherwise use the default settings	
Parse or Create Mode	Parse
Record - Set as directed, otherwise use the default settings	
Delimiter on Last Record	No
Record Delimiter	~
Record Type	Delimited

Modify the BatchRecord eWay Environment properties

The BatchRecordCreate and BatchRecordParse eWays share the same Environment Properties. The default BatchRecord eWay Environment property settings can be used for this sample, but they must be accepted.

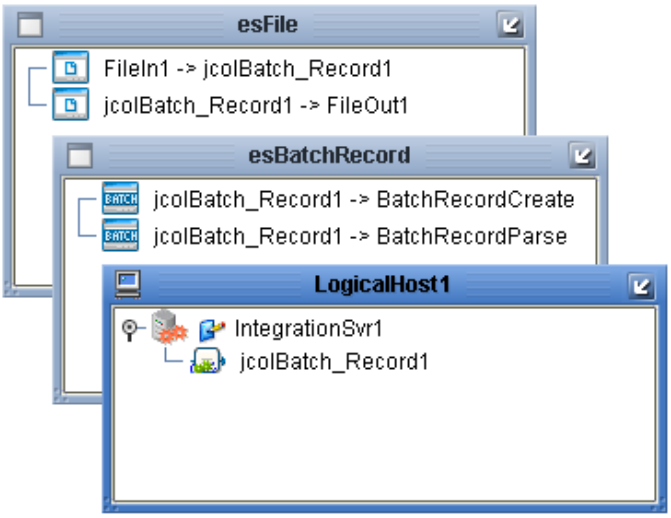
1. From the Environment Explorer, right-click the **esBatchRecord** External System, and select **Properties**. The Properties Editor opens to the BatchRecord eWay Environment properties. Click **OK** to accept the default settings and close the Properties Editor.
2. Save the current changes to your Repository.

7.7.8 Creating the Deployment Profile

A Deployment Profile is used to assign Collaborations and message destinations to the integration server and message server. Deployment Profiles are created using the Deployment Editor.

1. From the Enterprise Explorer's Project Explorer, right-click the Project (**prjBatch_Record**) and select **New** > **Deployment Profile**.
2. Enter a name for the Deployment Profile (for this sample **dpBatch_Record**). Make sure that the selected Environment is **envBatch_Record**. Click **OK**.
3. Click the **Automap** icon. The Project's components are automatically mapped to their system window as seen in **Figure 56 on page 254**.

Figure 56 dpBatchRecord Deployment Profile



4. If any of your Project components did not successfully map to an external system, open each of your eWay's configuration properties (Connectivity Map and Environment) and click **OK** to close and save the current configuration, then click **Automap** again.
5. Save your changes to the Repository.

7.7.9 Creating and Starting the Domain

To deploy your Project, you must first create a domain. A domain is an instance of a Logical Host. After the domain is created, the Project is built and then deployed.

Create and Start the Domain

1. Navigate to your **<*JavaCAPS51*>\logicalhost** directory (where <*JavaCAPS51*> is the location of your Sun Java Composite Application Platform Suite installation.
2. Double-click the **domainmgr.bat** file. The **Domain Manager** appears.
3. If you have already created a domain, select your domain in the Domain Manager and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.
4. If there are no existing domains, a dialog box indicates that you can create a domain now. Click **Yes**. The **Create Domain** dialog box appears.
5. Make any necessary changes to the **Create Domain** dialog box and click **Create**. The new domain is added to the Domain Manager. Select the domain and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

   For more information about creating and managing domains see the *Sun SeeBeyond eGate™ Integrator System Administration Guide*.

7.7.10 Building and Deploying the Project

The **Build** process compiles and validates the Project's Java files and creates the Project EAR file.

Build the Project

1. From the Deployment Editor toolbar, click the **Build** icon.
2. If there are any validation errors, a **Validation Errors** pane will appear at the bottom of the Deployment Editor and displays information regarding the errors. Make any necessary corrections and click **Build** again.
3. After the Build has succeeded you are ready to deploy your Project.

Deploy the Project

1. From the Deployment Editor toolbar, click the **Deploy** icon. Click **Yes** when the **Deploy** prompt appears.
2. A message appears when the project is successfully deployed. You can now test your sample.

Note: *Projects can also be deployed from the Enterprise Manager. For more information about using the Enterprise Manager to deploy, monitor, and manage your projects, see the Sun SeeBeyond eGate™ Integrator System Administration Guide.*

7.7.11 Running the Sample

To run your deployed sample Project do the following

1. Create any necessary directories (input and output).
2. From your configured input directory, paste (or rename) the sample input file to trigger the eWay.
3. From your output directory, verify the output data.

Note: *For UNIX, be sure to upload the appropriate LogicalHost*

7.8 The prjBatch_Secure Project

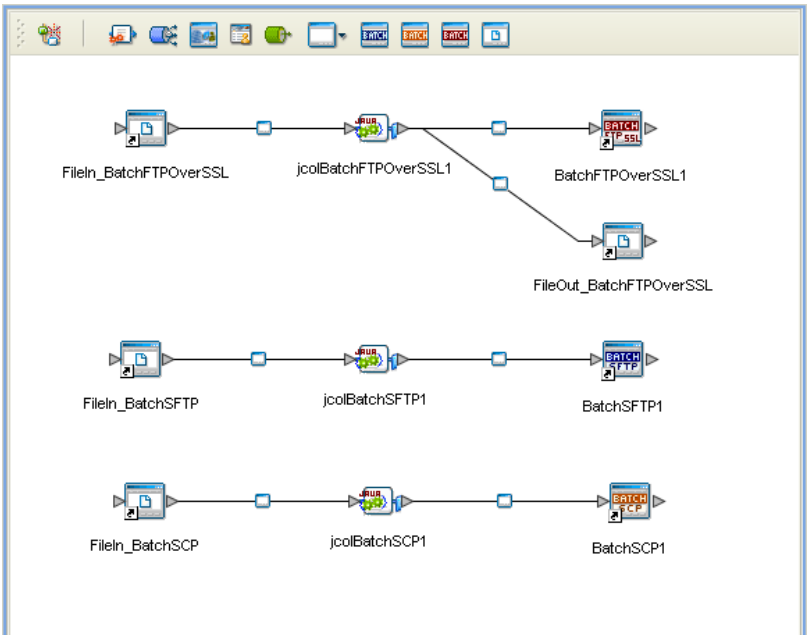
This section provides an overview of the **prjBatch_Secure** Project, and describes how to run the imported project. The sample is created in the same manner as the previous samples in this chapter.

The **prjBatch_Secure** Project contains three scenarios, as displayed in the **cmBatch_Secure** Connectivity Map, that employ three Batch OTDS to enable SSL or SSH secure file transfers.

The prjBatch_Secure Project Components

The prjBatch_Secure Project uses four External Applications, seven component eWays, and three Java Collaborations, as seen in the Project's Connectivity Map (see Figure 57).

Figure 57 cmBatchSecurity Connectivity Map



The Batch eWays perform the following functions

- The inbound File eWays (**FileIn_BatchFTPOverSSL**, **FileIn_BatchSFTP**, and **FileIn_BatchSCP**) act as triggers for the Project. The File eWays poll an external directory for a specific input file. When the input file is present, the File eWay triggers the Collaboration.
- **BatchFTPOverSSL**: is used by the Collaboration to:
  - get files from a remote directory and publish them to a local directory
  - get files from a local directory and publish them to the remote directory
  - download files from the remote directory to a local directory
  - upload files from a local directory to a remote directory
  - create and list remote directories
  - delete remote directories and list the results

  These functions can be seen in the **jcdBatchFTPOverSSL** Java Collaboration Definition (see **Figure 58 on page 259**).
- **BatchSFTP**: is used by the Collaboration to:
  - get files from a remote directory and publish them to a local directory
  - create a new remote directory
  - change directories to the new directory and publish a file to that directory
  - rename the published file
  - delete an unwanted file from the remote directory

  These functions can be seen in the **jcdBatchSFTP** Java Collaboration Definition (see **Figure 59 on page 260**).
- **BatchSCP**: is used by the Collaboration to:
  - recursively get a file from the remote directory and publish the file to a local directory
  - recursively put a file from the local directory to a remote directory

  These functions can be seen in the **jcdBatchSCP** Java Collaboration Definition (see **Figure 60 on page 261**).

7.8.1 The prjBatch_Secure Project Collaboration Definitions

The prjBatch_Secure Project includes three Java Collaborations, one for each of the three scenarios provided in the **cmBatch_Secure** Connectivity Map. The **jcdBatchFTPOverSSL**, **jcdBatchSCP**, and **jcdBatchSFTP** Java Collaboration Definitions are each created using the Collaboration Definition Wizard (Java).

The jcdBatchFTPOverSSL Java Collaboration

1. From the Project Explorer, right-click the **prjBatch_Secure** Project and select **New > Collaboration Definition (Java)** from the shortcut menu. The **Collaboration Definition Wizard (Java)** appears.

2 Enter a Collaboration Definition name (for this sample **jcdBatchFTPOverSSL**) and click **Next**.

3 For Step 2 of the Wizard, from the Web Services Interfaces selection window, double-click **SeeBeyond** > **eWays** > **File** > **FileClient** > **receive**. The Name field now displays **receive.** Click **Next**.

4 For Step 3 of the Wizard, from the Select OTDs selection window, double-click **SeeBeyond** > **eWays** > **BatcheWay > BatchFTPOverSSL**. The **BatchFTPOverSSL** OTD is added to the Selected OTDs field.

5 Click the **Up One Level** button to return to the Repository directory. Double-click **SeeBeyond** > **eWays** > **File** > **FileClient**. The **FileClient** OTD is added to the Selected OTDs field.

6 Click **Finish**. The Collaboration Editor (Java) opens to the new Collaboration in the right pane of the Enterprise Designer.

The jcdBatchSCP Java Collaboration

To create the **jcdBatchSCP** Collaboration, follow the procedures in the above section, replacing the Collaboration name with **jcdBatchSCP**, and selecting only the **BatchSCP** OTD for Step 4 of the Wizard.

The jcdBatchSFTP Java Collaboration

To create the **jcdBatchSFTP** Collaboration, follow the procedures in the above section, replacing the Collaboration name with **jcdBatchSFTP**, and selecting only the **BatchSFTP** OTD for Step 4 of the Wizard.

Collaboration Business Rules

The Java Collaboration Business Rules for the Project are separated into three Java Collaborations as seen from the Java Collaboration Editor: **jcdBatchFTPOverSSL** - **Figure 58 on page 259**, **jcdBatchSCP** - **Figure 60 on page 261**, and **jcdBatchSFTP** - **Figure 59 on page 260**.

Figure 58 jcdBatchFTPOverSSL Collaboration Definition

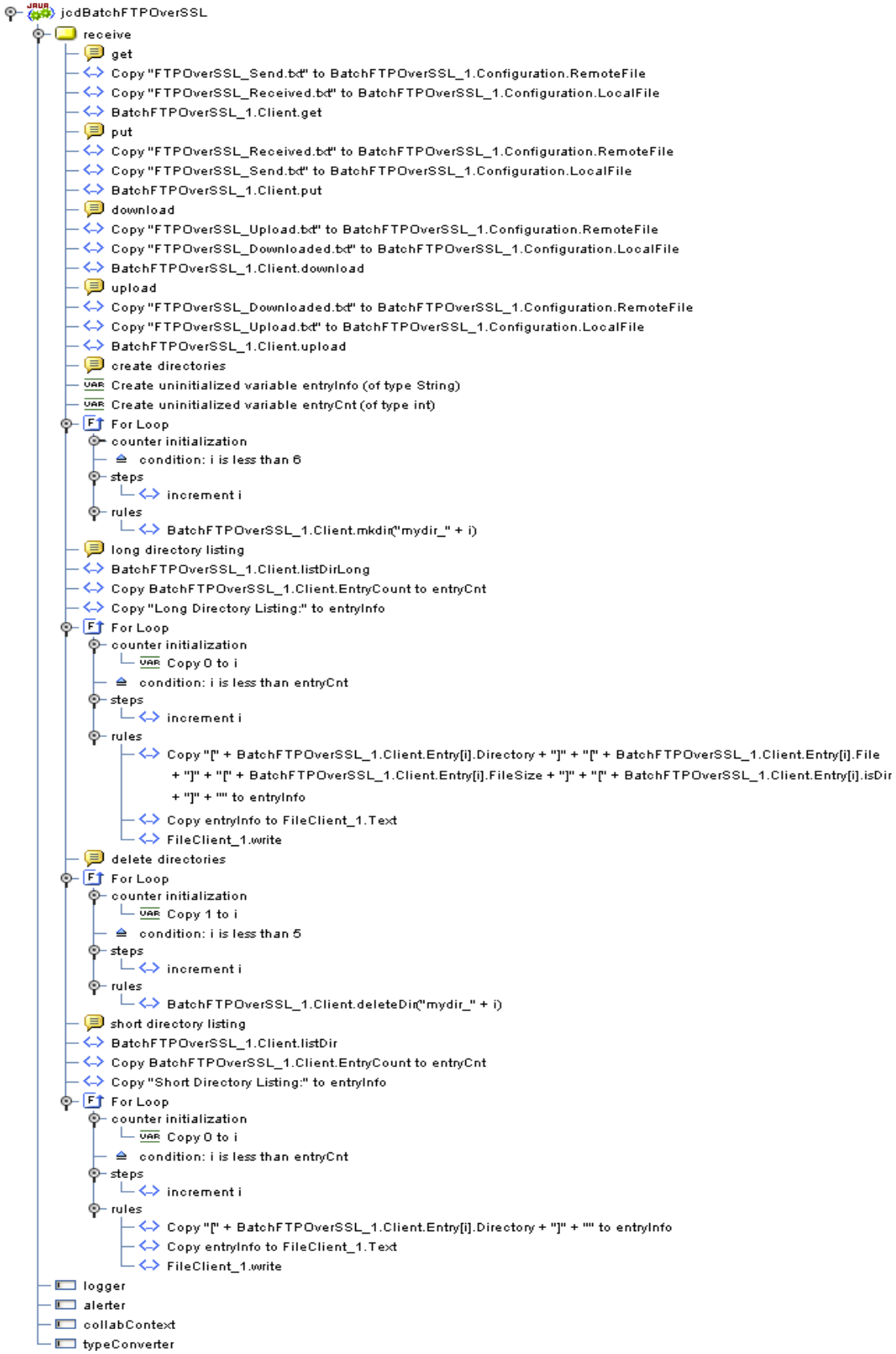


Figure 59 jcdBatchSCP Collaboration Definition

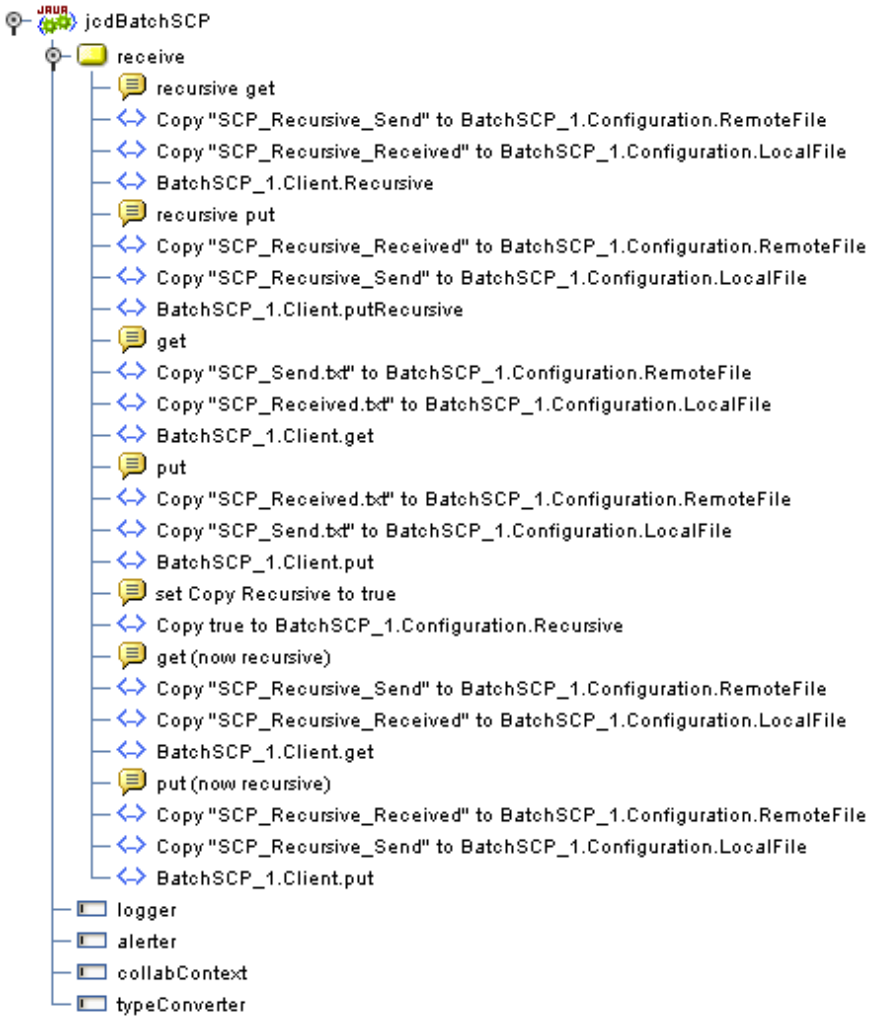


Figure 60 jcdBatchSFTP Collaboration Definition

7.8.2 Completing the prjBatch_Secure Project

Import the **prjBatch_Secure** Project as directed in **"Importing a Sample Project" on page 210**. After the sample Project has been imported and appears in your Project Explorer tree, create the Project's Environment.

7.8.3 Creating an Environment

Environments include the External Systems, Logical Hosts, Integration Servers and Message Servers used by a Project and contain the configuration information for these components.

1. From the Enterprise Designer's Enterprise Explorer, click the **Environment Explorer** tab.
2. Right-click the Repository and select **New Environment**. A new Environment is added to the Environment Explorer tree.
3. Rename the new Environment to **envBatch_Secure**.
4. Right-click **envBatch_Secure** and select **New > BatchFTPOverSSL External System**. Name this External System **esBatchFTPOverSSL**. The **esBatchFTPOverSSL** window is added to the Environment Editor.
5. Right-click **envBatch_Secure** and select **New > BatchSCP External System**. Name this External System **esBatchSCP**. The **esBatchSCP** window is added to the Environment Editor.
6. Right-click **envBatch_Secure** and select **New > BatchSFTP External System**. Name this External System **esBatchSFTP**. The **esBatchSFTP** window is added to the Environment Editor.

7 Right-click **envBatch_Secure** and select **New > File External System**. Name the External System **esFile**. Click **OK**. The **esFile** window is added to the Environment Editor.

8 Right-click **envBatch_Secure** and select **New > Logical Host**. The **LogicalHost1 box** is added to the Environment and **LogicalHost1** is added to the Environment Editor tree.

9 From the Environment Explorer tree, right-click **LogicalHost1** and select **New > Sun Integration Server**. A new Integration Server (**IntegrationSvr1**) is added to the Environment Explorer tree under **Localhost1**.

10 Save your current changes to the Repository.

7.8.4 Configuring the eWay Properties

The **prjBatch_Secure** Project uses seven component eWays, each represented in the Connectivity Map as a node between an External Application and the Service.

The **File, BatchFTPOverSSL**, **BatchSFTP**, and **BatchSCP** eWays contain properties which are configured from the Connectivity Map, and Environment Explorer.

Configuring the File eWay Connectivity Map properties

1 From the Connectivity Map, double-click the inbound **FileIn_BatchFTPOverSSL** eWay (between the **FileIn_BatchFTPOverSSL** External Application and the **jcolBatchFTPOverSSL1** service.

2 The Properties Editor opens to the inbound File eWay properties. Modify the configuration for your system, including the settings in Table 102.

Table 102 Inbound FileIn_BatchFTPOverSSL File eWay Properties
Table 103

Inbound File eWay Connectivity Map Properties	
Parameter Settings - Set as directed, otherwise use the default settings	
Input File Name	FileIn_BatchFTPOverSSL.txt

3 From the Connectivity Map, modify the inbound **FileIn_BatchSFTP** eWay (between the **FileIn_BatchSFTP** External Application and the **jcolBatchSFTP1** service (see Table 104).

Table 104 Inbound FileIn_BatchSFTP File eWay Properties
Table 105

Inbound File eWay Connectivity Map Properties	
Parameter Settings - Set as directed, otherwise use the default settings	
Input File Name	FileIn_BatchSFTP.txt

4 From the Connectivity Map, modify the inbound **FileIn_BatchSCP** eWay (between the **FileIn_BatchSCP** External Application and the **jcolBatchSCP1** service (see **Table 106 on page 263**).

Table 106 Inbound FileIn_BatchSCP File eWay Properties
Table 107

Inbound File eWay Connectivity Map Properties	
Parameter Settings - Set as directed, otherwise use the default settings	
Input File Name	FileIn_BatchSCP.txt

5 From the Connectivity Map, modify the Outbound **FileOut_BatchFTPOverSSL** eWay (between the **FileOut_BatchFTPOverSSL** External Application and the jcolBatchFTPOverSSL1 service (see Table 108).

Table 108 Outbound FileOut_BatchFTPOverSSL File eWay Properties
Table 109

Outbound File eWay Connectivity Map Properties	
Parameter Settings - Set as directed, otherwise use the default settings	
Output File Name	FileOut_BatchFTPOverSSL.dat

Modify the File eWay Environment properties

1 From the **Environment Explorer** tree, right-click the **esFile** External System, and select **Properties**. The Properties Editor opens to the File eWay Environment properties.

2 Modify the File eWay Environment properties for your system, including the settings in Table 110, and click **OK**.

Table 110 File eWay Environment Properties
Table 111

File eWay Environment Properties	
Inbound File eWay > Parameter Settings - Set as directed, otherwise use the default settings.	
Directory	*An input directory on your system that the inbound File eWays poll and pick up any specified input files.*
Outbound File eWay > Parameter Settings - Set as directed, otherwise use the default settings.	
Directory	*An output directory to which the outbound File eWay publishes (writes) the output messages.*

Modify the BatchFTP eWay Connectivity Map Properties

1 From the Connectivity Map, double-click the **BatchFTPoverSSL1** eWay, located between the **BatchFTPOverSSL1** External Application and the **jcolBatchFTPOverSSL1** service. The eWay Properties Editor appears.

2 Modify the **BatchFTPoverSSL** eWay Connectivity Map properties for your system, including the settings in Table 112, and click **OK**.

Table 112 BatchFTPoverSSL eWay Connectivity Map Properties
Table 113

BatchFTPoverSSL eWay Connectivity Map Properties	
FTP and SSL Settings - Set as directed, otherwise use the default settings	
Local Directory	*The local directory name and path for files that are sent to or received from a remote system. The default is X:/Batch51*
Target File Name	*The name of the remote file.*

Modify the BatchFTPoverSSL eWay Environment Properties

1 From the **Environment Explorer** tree, right-click the BatchFTPoverSSL External System (**esBatchFTPoverSSL** in this sample), and select **Properties**. The Properties Editor opens to the BatchFTPoverSSL eWay Environment properties.

2 Modify the BatchFTPoverSSL eWay Environment properties for your system, including the settings in Table 114, and click **OK**.

Table 114 BatchFTPoverSSL eWay Environment Explorer Properties
Table 115

BatchFTPoverSSL eWay Environment Properties	
FTP and SSL Settings - Set as directed, otherwise use the default settings.	
Host Name	*Host name or IP of the FTP server*
Server Port	*Port number to use to connect to the FTP server*
User Name	*User ID used to log into the system*
Password	*Password required to log into the system*

Configure the eWay properties for your system. For more information about the various properties, see **"Batch eWay Properties" on page 31**.

Create the Sample Directories

The **prjBatch_Secure** sample Project uses a number of directories. Create the directories associated eWay configuration properties:

- **Target Directory** (Inbound File eWay Environment Properties): **select a name for your target directory.**
- **Output Directory** (Outbound File eWay Environment Properties): **select a name for your output directory.**

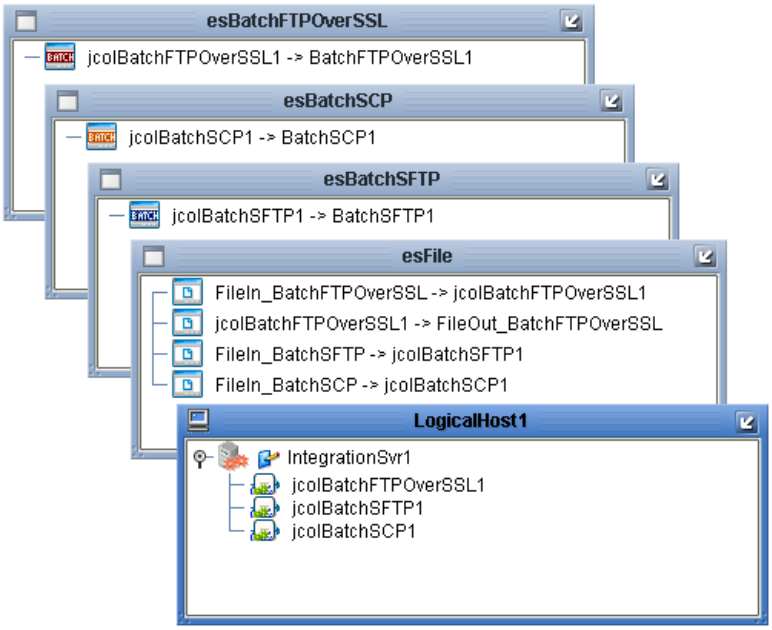
Create these directories on your system, or change the properties to reference your existing directories.

7.8.5 Creating the Deployment Profile

A Deployment Profile is used to assign Collaborations and message destinations to the integration server and message server. Deployment profiles are created using the Deployment Editor.

1. From the Enterprise Explorer's Project Explorer, right-click the Project (**prjBatch_Secure**) and select **New** > **Deployment Profile**.
2. Enter a name for the Deployment Profile (for this sample **dpBatch_Secure**). Make sure that the selected Environment is **envBatch_Secure**. Click **OK**.
3. Click the **Automap** icon. The Project's components are automatically mapped to their system window as seen in Figure 61.

Figure 61 dpBatch_Secure Deployment Profile



4. If any of your Project components did not successfully map to an external system, open each of your eWay's configuration properties (Connectivity Map and Environment) and click **OK** to close and save the current configuration, then click **Automap** again.
5. Save your changes to the Repository.

7.8.6 Creating and Starting the Domain

To deploy your Project, you must first create a domain. A domain is an instance of a Logical Host. After the domain is created, the Project is built and then deployed.

Create and Start the Domain

1. Navigate to your **<*JavaCAPS51*>\logicalhost** directory (where <*JavaCAPS51*> is the location of your Sun Java Composite Application Platform Suite installation.

2 Double-click the **domainmgr.bat** file. The **Domain Manager** appears.

3 If you have already created a domain, select your domain in the Domain Manager and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

4 If there are no existing domains, a dialog box indicates that you can create a domain now. Click **Yes**. The **Create Domain** dialog box appears.

5 Make any necessary changes to the **Create Domain** dialog box and click **Create**. The new domain is added to the Domain Manager. Select the domain and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

6 For more information about creating and managing domains see the *Sun SeeBeyond eGate™ Integrator System Administration Guide*.

7.8.7 Building and Deploying the Project

The **Build** process compiles and validates the Project's Java files and creates the Project EAR file.

Build the Project

1 From the Deployment Editor toolbar, click the **Build** icon.

2 If there are any validation errors, a **Validation Errors** pane will appear at the bottom of the Deployment Editor and displays information regarding the errors. Make any necessary corrections and click **Build** again.

3 After the Build has succeeded you are ready to deploy your Project.

Deploy the Project

1 From the Deployment Editor toolbar, click the **Deploy** icon. Click **Yes** when the **Deploy** prompt appears.

2 A message appears when the project is successfully deployed. You can now test your sample.

Note: *Projects can also be deployed from the Enterprise Manager. For more information about using the Enterprise Manager to deploy, monitor, and manage your projects, see the Sun SeeBeyond eGate™ Integrator System Administration Guide.*

7.8.8 Running the Sample

To run your deployed sample Project do the following

1 Create any necessary directories (target, input, output, and so forth).

2 From your configured input directory, paste (or rename) the sample input file to trigger the eWay.

3 From your output directory, verify the output data.

Note: *For UNIX, be sure to upload the appropriate LogicalHost*

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

W