

SUN SEEBEYOND

**eWAY™ TCP/IP HL7 ADAPTER
USER'S GUIDE**

Release 5.1.1



Copyright © 2006 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved. Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries. U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms. This distribution may include materials developed by third parties. Sun, Sun Microsystems, the Sun logo, Java, Sun Java Composite Application Platform Suite, SeeBeyond, eGate, eInsight, eVision, eTL, eXchange, eView, eIndex, eBAM, eWay, and JMS are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon architecture developed by Sun Microsystems, Inc. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd. This product is covered and controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. Nuclear, missile, chemical biological weapons or nuclear maritime end uses or end users, whether direct or indirect, are strictly prohibited. Export or reexport to countries subject to U.S. embargo or to entities identified on U.S. export exclusion lists, including, but not limited to, the denied persons and specially designated nationals lists is strictly prohibited.

Copyright © 2006 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, Etats-Unis. Tous droits réservés. Sun Microsystems, Inc. détient les droits de propriété intellectuels relatifs à la technologie incorporée dans le produit qui est décrit dans ce document. En particulier, et ce sans limitation, ces droits de propriété intellectuels peuvent inclure un ou plus des brevets américains listés à l'adresse <http://www.sun.com/patents> et un ou les brevets supplémentaires ou les applications de brevet en attente aux Etats - Unis et dans les autres pays. L'utilisation est soumise aux termes de la Licence. Cette distribution peut comprendre des composants développés par des tierces parties. Sun, Sun Microsystems, le logo Sun, Java, Sun Java Composite Application Platform Suite, Sun, SeeBeyond, eGate, eInsight, eVision, eTL, eXchange, eView, eIndex, eBAM et eWay sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd. Ce produit est couvert à la législation américaine en matière de contrôle des exportations et peut être soumis à la réglementation en vigueur dans d'autres pays dans le domaine des exportations et importations. Les utilisations, ou utilisateurs finaux, pour des armes nucléaires, des missiles, des armes biologiques et chimiques ou du nucléaire maritime, directement ou indirectement, sont strictement interdites. Les exportations ou réexportations vers les pays sous embargo américain, ou vers des entités figurant sur les listes d'exclusion d'exportation américaines, y compris, mais de manière non exhaustive, la liste de personnes qui font objet d'un ordre de ne pas participer, d'une façon directe ou indirecte, aux exportations des produits ou des services qui sont régis par la législation américaine en matière de contrôle des exportations et la liste de ressortissants spécifiquement désignés, sont rigoureusement interdites.

Version 20060620125602

Contents

Chapter 1

Introducing the TCP/IP HL7 eWay	8
About the TCP/IP HL7 eWay	8
TCP/IP HL7 Features	9
TCP/IP HL7 eWay Components	9
What's New in This Release	10
What's in This Document	10
TCP/IP HL7 eWay Javadoc	11
Intended Audience	11
Scope of the Document	11
Text Conventions	11
Sun Microsystems, Inc. Web Site	12
Documentation Feedback	12

Chapter 2

Installing the TCP/IP HL7 eWay	13
TCP/IP HL7 eWay System Requirements	13
TCP/IP eWay Components	13
Installing the TCP/IP HL7 eWay	14
Installing the TCP/IP HL7 eWay on an eGate Supported System	14
Adding the eWay to an Existing Suite Installation	15
Installing eWay Enterprise Manager plug-ins	15
TCP/IP HL7 eWay Alert Codes	16
Importing the TCP/IP HL7 eWay Sample Projects	17
Increasing the Enterprise Designer Heap Size	18
ICAN 5.0 Project Migration Procedures	19

Chapter 3

TCP/IP HL7 eWay Overview	22
The TCP/IP HL7 eWay Architecture	22

TCP/IP HL7 Resource Adapter	22
HL7 Collaborations	22
Generic HL7 OTDs	23
Sun SeeBeyond Composite Application Platform Suite Functionality	23
Modes and Roles	24
Standard Mode	24
Delayed ACK Mode	25
Inbound Functionality	27
Inbound eWay Data Flow	27
Standard Inbound Message Mode Data Flow and Architecture	27
Inbound Receiver Message Mode	28
Message Verification	29
Acknowledgment Processing	29
eWay Generates HL7 Acknowledgment	29
eGate Sends HL7 Acknowledgement	30
Canned HL7 NAK	30
Recourse Actions	30
Outbound Functionality	30
Outbound eWay Data Flow	31
Outbound Standard Messaging Mode	31
Outbound eWay Roles for Delayed ACK Scenarios	32
Outbound Delayed ACK Role	32
Outbound Forwarder Role	33
Message Verification	33
Acknowledgment Processing	33
eWay Generates HL7 Acknowledgment	33
eGate Sends HL7 Acknowledgement	33
Canned HL7 NAK	34
Recourse Action	34
General Functionality	35
Non-blocking I/O	35
HL7 Sequence Numbering Protocol	35
Failed Message Handling	35
Recourse Actions	36
Stopping the Collaboration with a Fatal Alert	36
TCP/IP HL7 eWay Operation	37
Direction	37
Connection Type	37
Connected as a TCP/IP HL7 Client	37
Connected as a TCP/IP HL7 Server	37
Lower Layer Protocol	37
MLLP	38
HLLP	38
HL7 Acknowledgment Level	39
Journaling	39
Error Queues	39
Alerts and Monitoring	39
Support for HL7 Version 2.5 SFT Segments	39
Delayed Acknowledgements	40

Chapter 4

Configuring TCP/IP HL7 eWay Properties	41
Creating and Configuring the TCP/IP HL7 eWay	41
Selecting TCP/IP HL7 as the External Application	41
Modifying the TCP/IP HL7 eWay Properties	42
Using the Properties Editor	43
TCP/IP HL7 eWay Configuration Properties	44
TCP/IP HL7 Inbound eWay Connectivity Map Properties	45
General Inbound Settings	45
TCPIP Inbound Settings	47
TCPIP Inbound Settings - Server Port Binding	51
TCPIP Inbound Settings - Client Connection Establishment	52
TCPIP Inbound Settings - Inbound Connection Management	53
TCPIP Inbound Schedules - Listener Schedule	55
TCPIP Inbound Schedules - Service Schedule	57
HL7 Acknowledgment	59
Lower Layer Protocol	61
Sequence Number Protocol	62
HL7 MSH Segment	63
HL7 SFT Segment	66
Communication Control	68
HL7 Recourse Action	70
TCP/IP HL7 Outbound eWay Connectivity Map Properties	73
General Outbound Settings	74
TCPIP Outbound Settings	75
TCPIP Outbound Settings - Client Connection Establishment	80
TCPIP Outbound Settings - Server Port Binding	82
HL7 Acknowledgment	83
Lower Layer Protocol	85
Sequence Number Protocol	86
HL7 MSH Segment	87
HL7 SFT Segment	90
Communication Control	92
HL7 Recourse Action	94
TCP/IP HL7 eWay Environment Explorer Properties	97
HL7 Inbound eWay - TCPIP Inbound Settings	97
HL7 Inbound eWay - MDB Pool Settings	98
HL7 Inbound eWay - Sequence Number Protocol	100
HL7 Outbound eWay - TCPIP Outbound Settings	101
HL7 Outbound eWay - Connection Pool Settings	102
HL7 Outbound eWay - Sequence Number Protocol	104

Chapter 5

Working with TCP/IP HL7 Collaborations	105
The TCP/IP HL7 eWay Collaborations	105
Inbound Collaboration Overview	106

Outbound Collaboration Overview	110
HL7 Outbound Test Collaboration	114
Creating a Copy of a Project	115
Exporting a Project	115
Importing a Project	115
Customizing Predefined Collaborations	115
Creating copies of the HL7 Collaborations	116
Adding an OTD to an Existing Collaboration	116

Chapter 6

Working With HL7 OTDs	118
The OTD Editor	118
Generic HL7 OTDs	118
Viewing an OTD using the OTD Editor	119
Modifying an OTD Using the OTD Editor	120
Editing an OTD's Root Properties	121
Root Node Properties	121
Editing the OTD Delimiters	123
Changing HL7 Standard Encoding Characters	124
Specifying Delimiters	124
Delimiter Properties	125
Delimiter Bytes	126
Terminator Mode Property	126
Precedence	127
OTD Properties	128
Node Properties	128
Element Properties	130
Field Properties	131
Specifying the Node Type	132
Node Management	133
Using the OTD Tester	134

Chapter 7

Working with the TCP/IP HL7 eWay Projects	136
TCP/IP HL7 Project Overview	136
Project Components	136
TCP/IP HL7 eWay Sample Projects	137
Completing a Project	138
Creating an Environment	139
Create the Environment for the prjHL7Outbound Sample	139
Creating Environments for the HL7 Inbound and Outbound Samples	140
Configuring the Properties	141
HL7Inbound HL7 eWay Sample	141

HL7ForwardMSG (Inbound) HL7 eWay Sample	141
HL7Outbound HL7 eWay Sample	141
HL7Forward (Outbound) HL7 eWay Sample	141
HL7OutboundDelayedAck HL7 eWay Sample	142
Configuring the Integration Server	142
Creating the Deployment Profile	142
Creating Deployment Profiles for the HL7 Inbound and Outbound Samples	144
Creating and Starting the Domain	144
Building and Deploying the Project	145
Running the Samples	145
Index	146

Introducing the TCP/IP HL7 eWay

This document describes how to install, configure, and implement the Sun SeeBeyond eWay™ TCP/IP HL7 Adapter, in a typical Sun SeeBeyond Composite Application Platform Suite environment.

This chapter provides a brief overview of operations and components, general features, and system requirements of the Sun SeeBeyond eWay™ TCP/IP HL7 Adapter (also referred to as the TCP/IP HL7 through this document).

What's in This Chapter

- [About the TCP/IP HL7 eWay](#) on page 8
- [What's New in This Release](#) on page 10
- [What's in This Document](#) on page 10
- [Sun Microsystems, Inc. Web Site](#) on page 12

1.1 About the TCP/IP HL7 eWay

The Sun SeeBeyond eWay™ TCP/IP HL7 Adapter enables the eGate Integrator system to exchange data with an external TCP/IP application, using the HL7 data protocol. eGate with the TCP/IP HL7 eWay utilizes J2EE™ Architecture to provide:

- Macro functionality, providing ease of use and productivity.
- Prebuilt Standards-compliant Inbound and Outbound Template Collaborations that work “as is” or can be modified for your specific needs.
- Customization of functionality, by configuring the eWay properties and/or modifying the Collaborations.
- Journaling and error messaging to JMS queues and topics. This is in addition to eGate’s standard alert and debug logging.
- Support for HL7 Standard versions 2.1, 2.2, 2.3, 2.3.1, 2.4, and 2.5.

Note: Throughout this document the term “JMS queue” is used in the generic sense and actually denotes JMS queues or topics.

1.1.1 TCP/IP HL7 Features

The TCP/IP HL7 eWay includes the following features:

- Bi-directional, including Client or Server mode in either direction (to or from eGate).
- Handles both HL7 HLLP and MLLP protocols and envelopes.
- Provides a wide variety of recourse action configurations.
- Non-blocking I/O.
- Recovery and retry logic.
- Debug levels.
- Error logging.
- Journaling of HL7 messages and associated ACKs
- HL7 Acknowledgement levels
- Fully supports the HL7 sequence numbering protocol.
- Full support for HL7 ACK, NAK generation and validation.
- Supports Delayed ACK in both directions.

1.1.2 TCP/IP HL7 eWay Components

The TCP/IP HL7 eWay incorporates three components:

- The HL7 TCP/IP Resource Adapter (commonly referred to as the eWay OTD) that implements the lower layer HL7 protocol over TCP/IP.
- Default inbound and outbound Collaborations that implement the HL7 messaging protocol, sequence numbering and recourse actions.
- Generic HL7 OTDs that provide the structures necessary to parse and create the data messages and ACKs used by the protocol.

The TCP/IP HL7 Object Type Definition (OTD) enables the creation of HL7 interfaces capable of running over TCP/IP, and also utilizes the common eWay services available in eGate. The TCP/IP HL7 eWay works hand in hand with the Sun SeeBeyond HL7 OTD Libraries, versions 2.1 through 2.5.

The TCP/IP HL7 eWay properties allow the user to easily configure the operation of the TCP/IP HL7 eWay. These properties are adopted into the OTD's functions (see [“Configuring TCP/IP HL7 eWay Properties” on page 41](#) for more information on the TCP/IP HL7 eWay properties).

The OTD handles all of the lower-layer protocol. The OTD's behavior is customized using the eWay configuration properties. These eWay properties are used by the resource adapter, but are also accessed and used by the prebuilt Collaborations.

1.2 What's New in This Release

The Sun SeeBeyond eWay™ TCP/IP HL7 Adapter version 5.1.1 includes the following changes and new features:

- **Version Control:** An enhanced version control system allows you to effectively manage changes to the eWay components.
- **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.
- **Added support to read configuration from LDAP at runtime.** 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.

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

The Sun SeeBeyond eWay™ TCP/IP HL7 Adapter User's Guide includes the following chapters:

- **Chapter 1 "Introducing the TCP/IP HL7 eWay"** provides a brief description of the Sun SeeBeyond eWay™ TCP/IP HL7 Adapter and an overview of the TCP/IP HL7 eWay User's Guide.
- **Chapter 2 "Installing the TCP/IP HL7 eWay"** provides directions for installing the TCP/IP HL7 eWay and accessing the accompanying documentation and sample Projects.
- **Chapter 3 "TCP/IP HL7 eWay Overview"** explains the basic functionality and operations of TCP/IP HL7 eWay.
- **Chapter 4 "Configuring TCP/IP HL7 eWay Properties"** describes how to configure the TCP/IP HL7 eWay to run in your environment and provides a description of the inbound and outbound eWay properties.
- **Chapter 5 "Working with TCP/IP HL7 Collaborations"** provides an overview and description of the structure and functionality of the Inbound and Outbound Collaborations provided as part of the TCP/IP HL7 eWay.
- **Chapter 6 "Working With HL7 OTDs"** describes how to view, test, and modify OTDs using the OTD Editor.
- **Chapter 7 "Working with the TCP/IP HL7 eWay Projects"** describes how to use the TCP/IP HL7 eWay template Projects that are installed automatically as part of the TCP/IP HL7 eWay.

1.3.1 TCP/IP HL7 eWay Javadoc

The TCP/IP HL7 eWay Javadoc is an API document in HTML format that provides information on the various methods available with the TCP/IP HL7 eWay. The Javadoc is available, along with the *Sun SeeBeyond eWay™ TCP/IP HL7 Adapter User's Guide* and Readme, by uploading **HL7eWayDocs.sar**. The Javadoc can then be downloaded from the **Documentation** tab of the Sun SeeBeyond Composite Application Platform Suite Installer.

To access the full Javadoc, extract the Javadoc to an easily accessible folder, and double click the **index.html** file.

1.3.2 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.

1.3.3 Scope of the Document

This user's guide provides a description of the Sun SeeBeyond eWay™ TCP/IP HL7 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 and functions.

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	<ul style="list-style-type: none">▪ Click OK.▪ On the File menu, click Exit.▪ Select the eGate.sar file.
Monospaced	Command line arguments, code samples; variables are shown in <i>bold italic</i>	<code>java -jar <i>filename</i>.jar</code>
Blue bold	Hypertext links within document	See Text Conventions on page 11
<u>Blue underlined</u>	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

Installing the TCP/IP HL7 eWay

This Chapter describes how to install the Sun SeeBeyond eWay™ TCP/IP HL7 Adapter, as well as the accompanying documentation and sample Projects.

What's in This Chapter

- [TCP/IP HL7 eWay System Requirements](#) on page 13
- [TCP/IP eWay Components](#) on page 13
- [Installing the TCP/IP HL7 eWay](#) on page 14
- [Importing the TCP/IP HL7 eWay Sample Projects](#) on page 17
- [ICAN 5.0 Project Migration Procedures](#) on page 19

2.1 TCP/IP HL7 eWay System Requirements

The TCP/IP HL7 eWay Readme contains the latest information on:

- Supported Operating Systems
- System Requirements
- External System Requirements
- The TCP/IP HL7 eWay Readme is uploaded with the eWay's documentation file (HL7eWayDocs.sar) and can be accessed from the Documentation tab of the Sun Java Integrator Suite Installer. Refer to the TCP/IP HL7 eWay Readme for the latest requirements before installing the TCP/IP HL7 eWay.

2.2 TCP/IP eWay Components

The TCP/IP HL7 eWay consists of the following components:

- The **TCP/IP HL7 eWay Adapter**: installed from the Suite installer.
- The **jcdHL7Inbound** and **jcdHL7Outbound Java Collaboration Definitions**: downloaded from the Suite Installer and imported to the Enterprise Designer.

The **jcdHL7Inbound** and **jcdHL7Outbound** Collaborations are designed to work “as is” for HL7 compliant interfaces, and can be configured for your specific needs using

only the property configuration files. For more information regarding these component HL7 Collaboration Definitions, see [“The TCP/IP HL7 eWay Collaborations” on page 105](#).

Note: *TCP/IP HL7 eWay Projects require the jcdHL7Inbound or jcdHL7Outbound Java Collaborations for full functionality. These Collaborations are imported with the sample Projects and can be copied to your new projects as needed. To import the TCP/IP HL7 sample Projects, you must first install the TCP/IP HL7 eWay and the File eWay.*

2.3 Installing the TCP/IP HL7 eWay

The Sun SeeBeyond 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.*

2.3.1 Installing the TCP/IP HL7 eWay on an eGate 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, 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 (located under eWay):

- ♦ **FileeWay** (the File eWay is used by the TCP/IP HL7 sample Projects)
- ♦ **HL7eWay**

To upload the TCP/IP HL7 eWay user's guide, help file, Javadoc, and readme, select the following (located under Documentation):

- ♦ **HL7eWayDocs**

- 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 To install the HL7 eWay Inbound and Outbound Collaboration Projects, see **“Importing the TCP/IP HL7 eWay Sample Projects” on page 17.**

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**.

When prompted, restart the IDE (Integrated Development Environment) to complete the installation.

See the *Sun SeeBeyond eGate™ Integrator User’s Guide* for more information on incorporating the eWay into an eGate Project.

2.4 Installing eWay Enterprise Manager plug-ins

The **Sun SeeBeyond Enterprise Manager** is a Web-based interface that allows you to monitor and manage your Java Integration Suite applications. The Enterprise Manager requires an eWay specific “plug-in” for each of your installed eWay types. 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 SeeBeyond 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 SeeBeyond 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**. This should include the **eWays Base Enterprise Manager Plug-In** and the **HL7 eWay Enterprise Manager Plug-In**. The application plug-ins are installed and deployed.
- 2 From the **Sun SeeBeyond Composite Application Platform Suite Installer**:
 - A From the **Sun SeeBeyond Composite Application Platform Suite Installer's** **Download** tab, select the Plug-Ins you require and save them to a temporary directory. This should include the **eWays Base Enterprise Manager Plug-In** and the **HL7 eWay Enterprise Manager Plug-In**.
 - 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.

TCP/IP HL7 eWay Alert Codes

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

The TCP/IP HL7 eWay utilizes the default eWay alert codes. The alert Message Code displays as **EWAY-ERROR**, followed by a specific dynamic description of the error condition, as displayed in [TCP/IP HL7 eWay Alert](#) on page 17.

Figure 1 TCP/IP HL7 eWay Alert

↕ Name	Value
Date	Wed May 31 13:56:17 PDT 2006
ID	118
Environment	Environment1
Physical Host	Sample.stc.com:18000
Logical Host	LogicalHost1
Server	IntegrationSvr1
Project	Project1
Deployment	Deployment1
Component	HL71
Severity	CRITICAL
Type	EWAY
Status	Unobserved
State	Running
Message Code	EWAY-ERROR
Details	TCPIEwayConnection.initialize(): Failed to establish/configure a physical TCP/IP connection. Please make sure the configuration settings are correct and the server is running. Got exception [java.lang.Exception: TCPIEwayConnection.initialize(): Tried <3> times. Still failed to connect to TCP/IP destination {localhost:7777}. Got exception [java.net.ConnectException: Connection refused: connect].].
Close	

For information on Managing and Monitoring alert codes and logs, see the *Sun SeeBeyond eGate Integrator System Administration Guide*.

2.5 Importing the TCP/IP HL7 eWay Sample Projects

The TCP/IP HL7 eWay's sample Projects, **prjHL7Inbound** and **prjHL7Outbound** contain the **jcdHL7Inbound** and **jcdHL7Outbound** Java Collaboration Definitions. These Collaborations are important components of the TCP/IP HL7 eWay, and can be copied and used as-is in your HL7 eWay Projects. To import the sample eWay Projects to the Enterprise Designer do the following:

- 1 Install the TCP/IP HL7 eWay. After the eWay has been added to your Suite installation, the TCP/IP HL7 Inbound and Outbound Collaboration Projects are available for download from the Sun Java Composite Application Platform Suite Installer's **Downloads** tab.
- 2 Extract the **HL7 eWay Inbound Collaboration Projects** and **HL7 eWay Outbound Collaboration Projects** to a local directory.
- 3 From the Enterprise Designer, save all unsaved work before importing the Projects.
- 4 From the Project Explorer pane, right-click your Repository and select **Import** from the shortcut menu. The **Import Manager** appears.
- 5 Browse to the directory to which you extracted the sample Project files. Select a sample file (for example, **prjHL7Inbound.zip**) and click **Import**. Repeat this process

to import each of the sample Projects. After all of the sample Projects have been successfully imported, click **Close**.

- 6 Before an imported sample Project can be run you must do the following:
 - ♦ Create an Environment (see [“Creating an Environment” on page 139](#))
 - ♦ Configure the eWays for your specific system (see [“Configuring the Properties” on page 141](#))
 - ♦ Create a Deployment Profile (see [“Creating the Deployment Profile” on page 142](#))
 - ♦ Create and start a domain (see [“Creating and Starting the Domain” on page 144](#))
 - ♦ Build and deploy the Project (see [“Building and Deploying the Project” on page 145](#))

2.6 Increasing the Enterprise Designer Heap Size

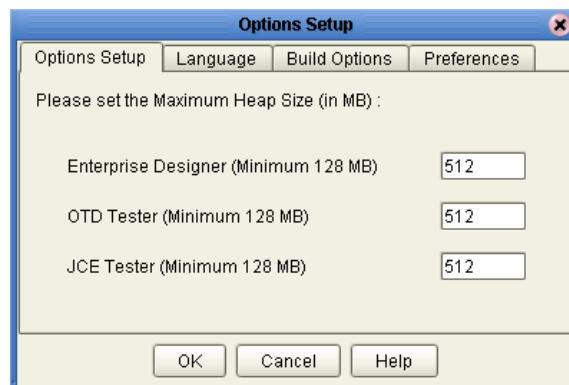
If you are also installing the **Sun SeeBeyond HL7 OTD Library**, we recommend that you increase the Enterprise Designer’s heap memory size. If the heap size is not sufficient it may result in an **OutOfMemoryError** message.

Increasing the heap size from the Enterprise Designer

To increase the heap size from the Enterprise Designer do the following:

- 1 From the Enterprise Designer’s toolbar, click **Tools** and select **Options**. The **Options Setup** dialog box appears.
- 2 Increase the configured heap size for the Enterprise Designer, OTD Tester, and JCE Tester to 512 MB as displayed in [Figure 2 on page 18](#). Click **OK**.

Figure 2 Options Setup - Heap Size



- 3 Close and restart the Enterprise Designer to allow your changes to take effect.

Increasing the heap size from the heapSize.bat file

If an **OutOfMemoryError** message occurs while you are trying to open the Enterprise Designer, the heap size settings may be changed before starting the Enterprise Designer. You can increase the heap size values found in the **heapSize.bat** file.

- 1 Go to the following directory and file:
`<eGate Install Directory>/edesigner/bin/heapSize.bat`
- 2 From the heapSize.bat file code, change the following heap size values to read as follows:
 - ♦ `set eDesigner_heap_size=512`
 - ♦ `set OTDTester_heap_size=512`
 - ♦ `set JCETester_heap_size=512`
- 3 Save the file, and start the Enterprise Designer.

2.7 ICAN 5.0 Project Migration Procedures

This section describes how to transfer your current ICAN 5.0 Projects to Sun Java Composite Application Platform Suite, version 5.1.1. Only Projects developed on ICAN version 5.0.2 and above can be migrated successfully to the Sun Java Composite Application Platform Suite. 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

- 7 Install the Sun Java Composite Application Platform Suite, 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 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 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.
- 12 Close the Import Manager after the Project is successfully imported.

Deploy the Project

- 13 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-protect 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**.
- 14 If your imported Project includes File eWays, these must be reconfigured in your Environment prior to deploying the Project. To reconfigure your File eWays, do the following:
 - A The Environment File External System properties can now accommodate both inbound and outbound eWays. If your previous 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 The Directory property has been relocated from the Connectivity Map Properties to the Environment Properties. Set the inbound and outbound Directory values, and click **OK**.
- 15 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.*

TCP/IP HL7 eWay Overview

This chapter explains the basic functionality and operations of TCP/IP HL7 eWay.

Chapter Topics

- [“The TCP/IP HL7 eWay Architecture” on page 22](#)
- [“Inbound Functionality” on page 27](#)
- [“Outbound Functionality” on page 30](#)
- [“General Functionality” on page 35](#)
- [“TCP/IP HL7 eWay Operation” on page 37](#)

3.1 The TCP/IP HL7 eWay Architecture

The TCP/IP HL7 eWay’s functionality is a result of the TCP/IP HL7 Resource Adapter (RA) combined with the predefined Inbound and Outbound HL7 Collaborations and the generic HL7 OTDs.

TCP/IP HL7 Resource Adapter

The TCP/IP HL7 eWay **Resource Adapter** communicates with external HL7 system, establishes and maintains the TCP/IP socket, manages message enveloping, maintains the sequence numbering file, and provides the HL7 protocol state to the Collaboration. The RA (Resource Adapter) is configured from the eWay Properties Editor.

HL7 Collaborations

The **Inbound** and **Outbound HL7 Collaborations** provide message validation, sequence numbering, ACK and NAK generation, and recourse actions. The predefined HL7 Collaborations are designed to implement the HL7 standard protocol and inter-operate with similar standard compliant systems by simply changing the eWay property configuration.

If a system does not conform to the HL7 specification, the Collaborations can be modified for that transaction by changing the Java code using the Collaboration Editor. The Collaborations Java code is designed to be available and “transparent” so that the user can easily reference the predefined Java code to see how it currently handles HL7 transactions and make the appropriate modifications.

The Enterprise Designer's Collaboration Editor enables the user to edit or create Java code to modify a Collaboration for their specific needs. In many cases this code can be created graphically (*drag and drop*), using the Collaboration Editor's Business Rules Designer. In situations where user is changing the code of a prebuilt Collaboration, it is recommended that the user duplicate the Collaboration first, and then modify it.

The Collaborations are designed to target "one unit of work" at a time, meaning the resolution of one message at a time. Once the current message transaction is resolved, the Collaboration is free to process the next HL7 message. The general form of the Collaborations is a state machine. Based on the state of the connection, the Collaboration performs the appropriate action.

Additional Collaborations can be added to a Project to increase message flow.

Generic HL7 OTDs

The generic HL7 OTDs are version agnostic structures used to send and receive HL7 messages, acknowledgements, and NAKS. They provide the Collaboration with only the essential fields for implementing the HL7 protocol. If a user wishes to do version or message type specific functionality, then they may choose to add the appropriate OTD to the Collaboration.

Sun SeeBeyond Composite Application Platform Suite Functionality

The TCP/IP HL7 eWay also takes advantage of the Sun SeeBeyond Composite Application Platform Suite facilities to provide journaling and error messaging, as well as monitoring, alerting, and logging. Journaling and error messages are sent to JMS queues which allow flexibility for post-processing. For example, invalid messages and their negative acknowledgements (NAKs), are sent to a JMS queue. This JMS queue can then be setup to allow the invalid HL7 messages to be viewed, corrected, and resubmitted automatically to the same eWay, using a Sun SeeBeyond eVision Web application.

Error Queues

Each Collaboration automatically sends invalid messages that either have incorrect data or format, or are deemed unacceptable to a JMS error queue. The error generated by the message and, if appropriate, the NAK associated with it, are written as JMS properties of that message for processing later. The user has the option of sending errors to one common queue, or to a specified queue for each eWay, or some combination of both.

Journaling Queues

When journaling is enabled, the HL7 message and its related acknowledgement (ACK) are written to a JMS journal queue.

The number of JMS error queues or journal queues used by a Project is up to the user. From the JMS queue, these messages can be accessed and written to file, sent to a database, sent to a Web application for processing, and so forth.

Monitoring

The Enterprise Manager provides a real-time picture of the eWay's state and status. The monitoring facilities for the eWay display the following status:

- eWay up or down
- Connected to external
- Current sequence number
- Date and time of the last transaction
- eWay properties

Alerts

Alerts are sent from the RA and the Collaborations when conditions are identified that endanger or stop the interface. Users can add their own custom alert messages to the Collaborations.

Logging

Log messages are written from both the RA and the Collaborations. Users can also configure their own log messages. The level is set from the Enterprise Manager

For more information on monitoring, alerting and logging, see the *Sun SeeBeyond eGate™ Integrator System Administration Guide* and the *Sun SeeBeyond eGate™ Integrator User's Guide*.

3.2 Modes and Roles

The HL7 eWay and eGate are capable of operating in two modes: **Standard** and **Delayed ACK**. Standard mode being the typical message exchange in HL7. Standard mode meaning, send an HL7 message, receive an HL7 ACK (or the opposite of this flow). Delayed ACK mode is where the exchange of a message requires two acknowledgements: one to confirm the message was received and the other from the external system that actually received the message, to verify that it was received.

In these two modes, the HL7 eWay and eGate may have a number of roles that they play within certain scenarios; that is, certain components can fulfill different responsibilities within a protocol. For example, the outbound Collaboration can fulfill 2 roles in the Delayed ACK mode: one as the "sender of messages" that expects 2 delayed ACKS, and another as the "forwarder of ACKS" from the external system.

Note: *Delayed ACK mode is deprecated as of HL7 version 2.2 and has been removed from the HL7 Standard as of version 2.5.*

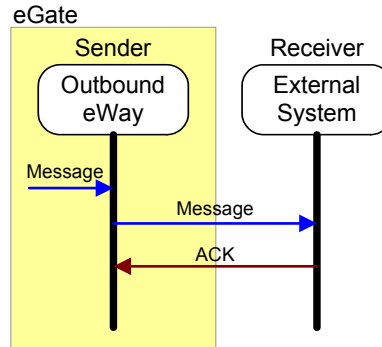
Standard Mode

In Standard mode, the HL7 eWay can assume two roles: Sender and Receiver. These are implemented in the outbound and inbound Collaborations respectively.

HL7 eWay Sender Role

The outbound Collaboration is the implementation of the Sender and the RA is configured for an outbound data flow as displayed in [Figure 3 on page 25](#).

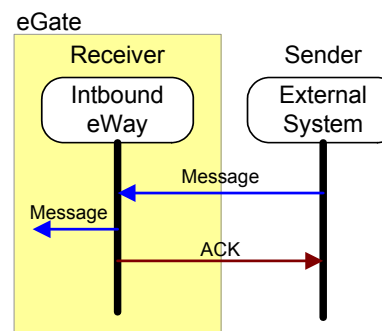
Figure 3 HL7 eWay Sender Role



HL7 eWay Receiver Role

The inbound Collaboration is the implementation of the Receiver role in conjunction with the RA being configured for inbound direction as displayed in [Figure 4](#).

Figure 4 HL7 eWay Sender Role



Delayed ACK Mode

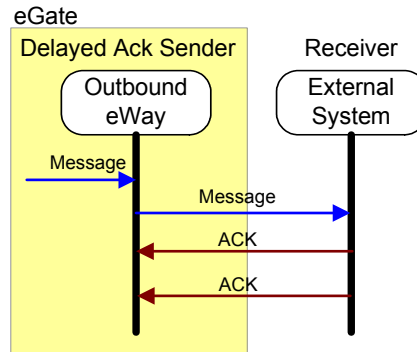
Delayed ACK mode is an extension to the basic or standard HL7 message exchange mode, where there is some middleware component between the **Sender** and the **Receiver**. The sending system expects to receive the ACK from the receiving system in addition to the middleware component. In this mode, eGate can assume two roles in the protocol: as the **Sender** and as the **Receiver**.

eGate Sender Role

In this role, eGate acts as the **Sender** in the exchange. The HL7 RA is configured for the outbound direction, and the HL7 outbound Collaboration is configured so that the **eGate Sends App Ack** property is set to **True**. This parameter is used even though eGate (HL7 eWay) technically does not send an Application ACK. Rather, it receives the Application ACK and provides the compliment behavior to the ACK sent by the inbound configured eWay. The two are related in the protocol.

The purpose of this role is for eGate to act as if it is a system that requires the Delayed ACKs so that it can communicate with a system that operates in the Delayed ACK Receiver role as displayed in [Figure 5 on page 26](#).

Figure 5 HL7 eWay Sender Role

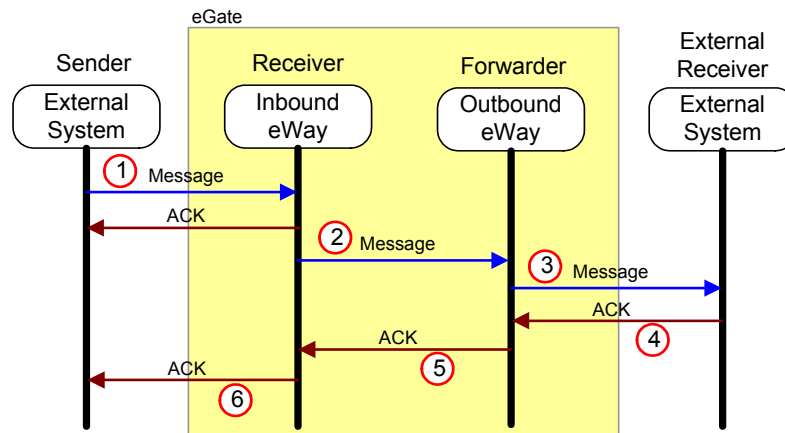


An example implementation of this role is available in the **HL7Outbound** sample Project on page 138.

eGate Receiver and Forwarder Role

To perform this role, eGate is configured with two instances of the HL7 eWay, each performing its own role as **Receiver** and **Forwarder**.

Figure 6 HL7 eWay Sender and Forwarder Role



Note: For Delayed ACK, the Receiver and Forwarder must be on the same integration server.

The following steps convey the process displayed in Figure 6.

- 1 The **Receiver** accepts the HL7 message. It then returns the first ACK, with an MSH - 5, value "D" (for the Delayed ACK), to the **Sender**.
- 2 The **Receiver** sends the message to the **Forwarder**.
- 3 The **Forwarder** sends the message to the receiving **External System**.
- 4 The receiving **External System** sends an ACK to the **Forwarder**.

- 5 The **Forwarder** receives the ACK from the **External** and forwards it on to the **Receiver**.
- 6 The **Receiver** then forwards the ACK, with an **MSH - 5**, value “F” indicating that it is the receiving External System’s ACK, to the **Sender**.

In the **Receiver** role, the HL7 RA is configured for **inbound**, and the inbound Collaboration is used with the parameter, **eGate Sends App Ack**, set to **True**. In the **Forwarder** role, the RA is set to **outbound**, the parameter, **Forward External Acks to eGate**, is set to **True**, and the **outbound Collaboration** is selected.

This configuration table presents the necessary parameters used to configure the eWay to assume the Delayed ACK roles.

Table 2 eWay Delayed Ack Configuration

Role	Direction	eGate Sends App Ack Property	Forward External Acks to eGate
Sender	Out	True	N/A
Receiver	In	True	N/A
Forwarder	Out	N/A	True

An example of the receiver role is provided in the **prjHL7Inbound** sample Project on page 138. The **prjHL7Outbound** Project provides a sample implementation of the Forwarder role.

3.3 Inbound Functionality

The Inbound TCP/IP eWay Project, **prjHL7Inbound**, provides a sample implementation of an inbound flow using the eWay. It can be configured for “standard inbound” mode or in “forward message” mode.

3.3.1 Inbound eWay Data Flow

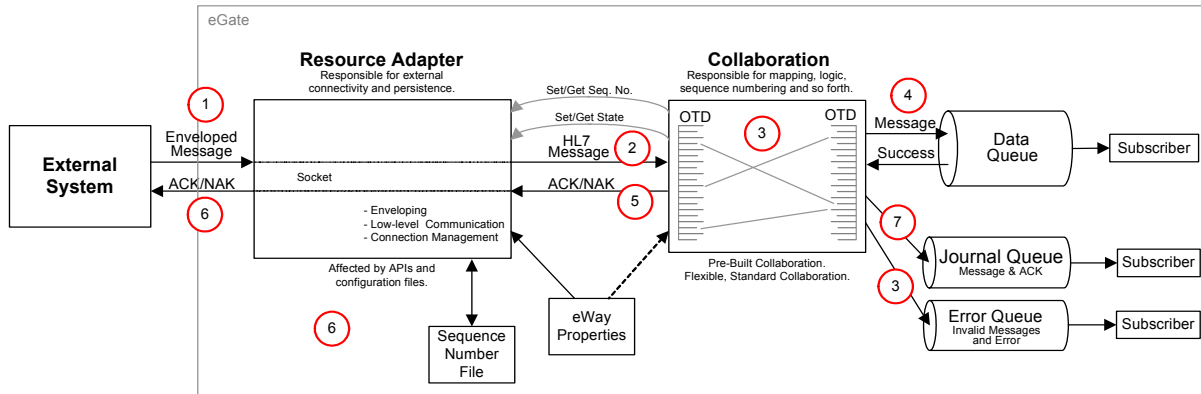
The inbound TCP/IP HL7 eWay Project receives HL7 messages from an external system, sends an acknowledgement of the message to the external, provides sequence numbering, writes the HL7 message to a JMS data queue, and also writes the HL7 message and ACK to a JMS journal queue. Any error messages and NAKs are sent to a JMS error queue.

The HL7 data is processed so that all the fields in the MSH segment of the message are stored in an internal structure to generate an HL7 response. Non-HL7 data, including HL7 acknowledgments, automatically generate warnings in the eWay’s log file and send an HL7 NAK to the external system.

Standard Inbound Message Mode Data Flow and Architecture

Figure 7 on page 28 illustrates the flow of data and architecture for an inbound eWay. A description of the diagram follows.

Inbound eWay



- 1 The external system sends the HL7 message to the eWay.
- 2 The Collaboration receives the HL7 message.
- 3 The Collaboration validates the message (if validate is enabled). If it fails, the Collaboration takes the configured recourse action. If the recourse action is stripped and the maximum number of retries has been exceeded, the message and error are written to the Error Queue.
- 4 The Collaboration writes the message to the Data Queue.
- 5 The Collaboration then creates the appropriate ACK and sends it to the RA.
- 6 The RA envelopes the ACK and sends it to the External System.
- 7 If Journaling is enabled, the message and its ACK are written to the Journal Queue.

Inbound Receiver Message Mode

```
sequenceDiagram
    participant Sender as Sender External
    participant Inbound as Inbound eWay
    participant Outbound as Outbound eWay
    participant External as External System

    Sender->>Inbound: 1 Message
    Inbound->>Sender: 2 ACK
    Inbound->>Outbound: 3 Message
    Outbound->>Inbound: 6 ACK
    Outbound->>External: 4 Message
    External->>Outbound: 5 ACK
    Outbound->>Sender: 7 ACK
```

The Inbound Receiver Message mode is used when the Delayed ACK is configured to fulfill the role of the Receiver in the Delayed ACK scenario. It accepts the message and acknowledges the External and then forwards the message to the component fulfilling the Forwarder role. It then accepts the ACK from the Forwarder and passes it on to the External that sent the message.

The following steps convey the Inbound Forward Message Role displayed in Figure 8.

- 1 The Sender External, sends an HL7 message to the Inbound eWay, which is configured as a Receiver (the **eGate Sends App Acks** is enabled).

- 2 The Inbound eWay receives the HL7 message and returns the first Acknowledgement to the External with an **MSA - 5**, value “**D**” for Delayed Acknowledgement. The External receives the ACK, validates the ACK (verifying that it is a Delayed ACK), and waits for another ACK.
- 3 The Inbound eWay creates a JMS message with the HL7 message as the payload, creates a “reply to” destination, and forwards the HL7 message to the Outbound Forwarder (to a JMS destination).
- 4 The Outbound Forwarder gets the HL7 message and forwards the message to the External System.
- 5 The External System receives the HL7 message and returns the HL7 ACK message to the Outbound Forwarder.
- 6 The Outbound Forwarder gets the HL7 ACK message and sends it to the Inbound Receiver eWay using the “reply to” destination.
- 7 The Receiver External reads the HL7 ACK message and forwards the second HL7 ACK message with an **MSA - 5**, value “**F**” to the Sender External. The Sender External then takes the appropriate action: for example, journaling the HL7 message and the HL7 ACK.

3.3.2 Message Verification

Message verification begins with reading the message from the external system. The message is expected to match the MLLP envelope, since both HLLP and MLLP envelopes have the Start of Block (SOB), End of Data (EOD), and a Carriage Return (CR) in common.

If a message fails the read verification, it is considered bad data. If read by an inbound eWay, this failure causes the eWay to generate a Canned HL7 NAK. (An outbound eWay ignores the message and logs a warning, reporting the nature of the problem to the log file.)

An HLLP envelope needs further verification as to whether it is data or a NAK, as well as the Block Checksum and Block Size. The eWay behaves as described above if the HLLP envelope verification fails.

After stripping the message envelope, the RA hands the de-enveloped message to the inbound Collaboration where it is parsed into the generic event OTD. This ensures that the general form and MSH segment are valid. If the MSH property is set, the Collaboration verifies that the fields specified in the HL7 segment section are the same as those of the received MSH, otherwise, a NAK is returned.

3.3.3 Acknowledgment Processing

eWay Generates HL7 Acknowledgment

In this scenario, the eWay generates an HL7 ACK after receiving and successfully storing the message in a queue; otherwise, it generates an HL7 NAK. The HL7 ACK or NAK is placed in the proper envelope and sent to the external system.

eGate Sends HL7 Acknowledgement

In this scenario, the eWay acts as a receiver in a Delayed ACK scenario, as described in [Inbound Receiver Message Mode](#) on page 28.

Canned HL7 NAK

A Canned HL7 NAK is created when a read error occurs, or when an message cannot be identified as an HL7 message. The initial test ensures that the message conforms to the lower-layer protocol. The Resource Adapter uses the MSH section parameters to create an appropriate NAK.

Recourse Actions

Recourse actions can be configured for the inbound eWay for the following conditions:

- [Action on Max Failed Read Retry](#) in response to the configured maximum number of canned negative acknowledgments that the eWay sends (see [Max Empty Read Retry](#) on page 92).
- [Action on Max Nak Received](#) in response to the configured maximum number of times the eWay attempts to read data from the external system after the read/receive operation returns nothing (see [Max NAK Receive Retry](#) on page 93).
- [Action on Max Nak Sent](#) in response to the configured maximum number of negative acknowledgments and Canned Naks the eWay sends (see [Max NAK Send Retry](#) on page 69 and [Max Canned NAK Send Retry](#)).
- [Action on Max No Response](#) in response to the configured maximum number of response timeouts the eWay allows while waiting for data from the external system (see [Action on No Response](#) on page 70).
- [Action on Nak Received](#) in response to receiving an HL7 Application NAK from the external system.
- [Action on No Response](#) in response to the configured amount of time (in milliseconds) that the eWay waits for a response from the external system (see [Time To Wait For A Response](#) on page 92).

See [Recourse Actions](#) on page 36 for more information on the available Recourse Actions.

3.4 Outbound Functionality

The Outbound TCP/IP eWay Project, **prjHL7Outbound**, can be implemented in a “Standard Outbound” mode, or in two “Forward message” modes: outbound Delayed ACK or Outbound Forwarder.

3.4.1 Outbound eWay Data Flow

In the outbound direction, the eWay receives HL7 messages from a JMS queue. Each Event is verified to ensure it contains HL7 data only. Legitimate HL7 data coming from eGate is enveloped into its configured format and sent to the external system.

A message in the JMS queue triggers the outbound Collaboration. The outbound Collaboration is provided with an HL7 message to send to the external system

The eWay waits for a configurable number of milliseconds (see [Time To Wait For A Response](#) on page 92) for an incoming HL7 ACK or NAK from the external system. After receiving an HL7 response from the external system, the eWay strips the message from its envelope and verifies its integrity.

Any non-HL7 acknowledgment received from the external system causes the eWay to resend the same message. If the incoming response is an HL7 ACK or NAK, the eWay may do either of the following, as dictated by its configuration:

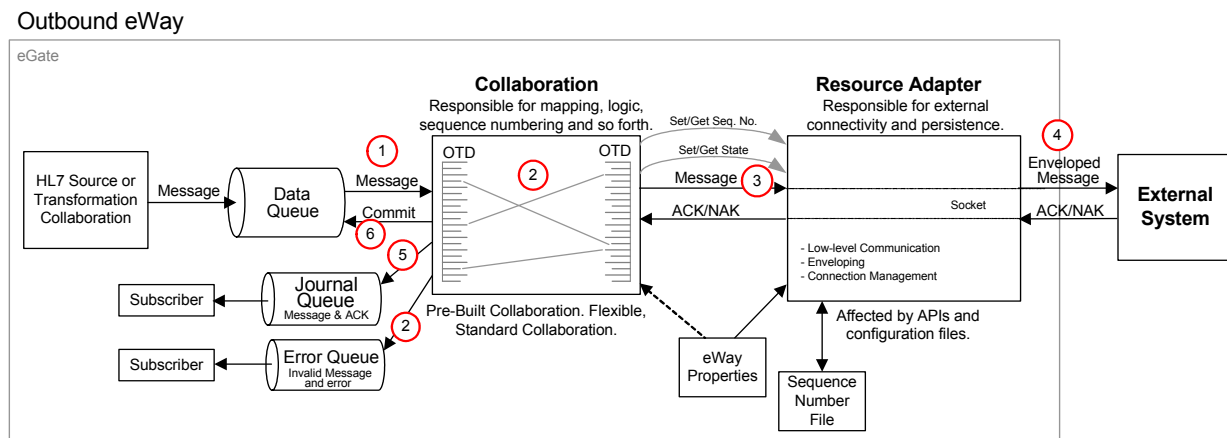
- Recourse Action on NAK received.
- Recourse Action on Max NAK received.

If journaling is set, these messages and their ACKs are placed in the journal file.

Outbound Standard Messaging Mode

Figure 9 illustrates the flow of data for an outbound eWay.

Figure 9 Data Flow for the Outbound TCP/IP HL7 eWay



The following steps convey the process displayed in [Figure 9 on page 31](#).

- 1 An HL7 message triggers the Collaboration. The outbound Collaboration is designed to accept the HL7 Messages.
- 2 The Collaboration maps the received message into the Generic Event OTD and validates the MSH segment. If validation is enabled, the Collaboration checks the MSH segment of the outbound messages against MSH values configured in the eWays Properties file. If the validation fails or the message cannot be parsed, the

message and its error are written to the Error Queue. Note that the HL7 message is always checked for structural correctness.

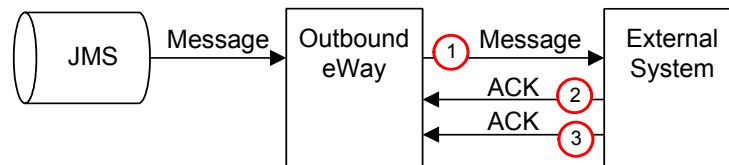
- 3 The Collaboration sends the message to the RA.
- 4 The RA envelopes the message and sends it to the External System and waits for an ACK.
- 5 The Collaboration receives and validates the ACK, and then journals the ACK and the HL7 message (if journaling is enabled). If the Collaboration receives a NAK, the NAK and the HL7 message are sent to the error queue.
- 6 Finally, the Collaboration commits the JMS receive.

3.4.2 Outbound eWay Roles for Delayed ACK Scenarios

The outbound eWay can fulfill two roles in a Delayed ACK scenario.

Outbound Delayed ACK Role

Figure 10 Outbound Delayed ACK Role



The outbound Delayed Acknowledgement mode is used to communicate with an external system that is configured to receive messages in a delay ACK way; that is, it receives two ACKs: one to confirm the message was received, and one from the application that accepts the message.

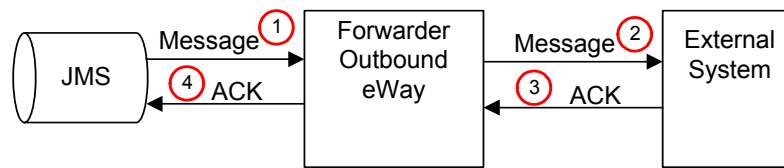
For Delayed ACK mode, the process is similar to that of the standard outbound mode, except that it receives two ACKs. The initial ACK comes from the receiving system.

The following steps convey the Outbound Delayed Acknowledgement Role process displayed in Figure 10.

- 1 The outbound eWay, which is configured as Delayed Acknowledgement role, receives a message from JMS, and sends the message to the External System.
- 2 The External System receives the message and returns the first Acknowledgement to the outbound eWay with an **MSA - 5**, value **"D"** for Delayed Acknowledgement. The outbound eWay receives the ACK, validates the ACK (verifying that it is a Delayed ACK), and waits for another ACK.
- 3 The outbound eWay receives another HL7 ACK message (the second) and validates that the second HL7 ACK message is an **MSA - 5**, with a value of **"F."** If the second ACK is valid, the eWay commits the message, otherwise it resends the message.

Outbound Forwarder Role

Figure 11 Outbound Forwarder Role



The Outbound Forward Message role is used in conjunction with the with the inbound eWay, which is also configured to handle delayed ACKs. No validation is preformed: the eWay acts as a “pass-through.”

The following steps convey the Outbound Forwarder Role displayed in Figure 11.

- 1 Data is received by the Collaboration, from the JMS queue.
- 2 The Collaboration extracts the JMS property “reply to” destination from the message, but does no validation, and sends the message to the External System.
- 3 The eWay receives the ACK from the External System.
- 4 The Collaboration sends the ACK to the temporary topic that was contained in the “reply to.”

Note: For Delayed ACK, the Receiver and Forwarder must be on the same integration server.

3.4.3 Message Verification

The only verification that the outbound eWay does is to ensure that the message parses into the generic Event OTD, and that the MSH uses the correct fields. The acknowledge is verified to ensure that the sent message is valid.

3.4.4 Acknowledgment Processing

eWay Generates HL7 Acknowledgment

In this scenario, the eWay generates an HL7 ACK after receiving and successfully storing the message in a queue; otherwise, it generates an HL7 NAK. The HL7 ACK or NAK is placed in the proper envelope and sent to the external system.

eGate Sends HL7 Acknowledgement

In this scenario, the eWay acts as a sender in a Delayed ACK scenario, as described in [Delayed ACK Mode](#) on page 25.

Canned HL7 NAK

A Canned HL7 NAK is created when a read error occurs, or when an message cannot be identified as an HL7 message. The initial test ensures that the message conforms to the lower-layer protocol. The Resource Adapter uses the MSH section parameters to create an appropriate NAK.

Recourse Action

Recourse action can be configured for the outbound eWay for the following conditions:

- **Action on Max Failed Read Retry** in response to the configured maximum number of times the eWay attempts to read data from the external system after the read/receive operation returns nothing (see **Max Empty Read Retry** on page 92).
- **Action on Max Nak Received** in response to the configured maximum number of negative acknowledgments the eWay receives (see **Max NAK Receive Retry** on page 93). This action is also used in cases where HL7 message validation failed prior to the sending of the HL7 message to the external system.
- **Action on Max No Response** in response to the configured maximum number of response timeouts the eWay allows, while waiting for data from the external system (see **Max No Response** on page 92).
- **Action on Nak Received** in response to eWay receiving an HL7 Application NAK from the external system.
- **Action on No Response** in response to the configured the amount of time (in milliseconds) that the eWay waits for a response from the external system (see **Time To Wait For A Response** on page 92).

See **Recourse Actions** on page 36 for more information on the available Recourse Actions.

Note: *The TCP/IP HL7 eWay includes internal counters that keep track of all error conditions.*

3.5 General Functionality

This section explains the eWay's general functionality and features.

3.5.1 Non-blocking I/O

The non-blocking I/O feature will prevent the eWay from locking up when an attempt to read or write data blocks. This will enable the eWay to continue its operation in case of any communication errors.

If the read attempt fails for a configurable number of times, the eWay will exit or reset its connection to the external system, based on its configuration.

In the event of a failed write, it is able to resume its write operation to pick up where it previously left off until the entire message is successfully sent.

Without this feature, there is a possibility for the eWay to lock up when a read or write failure occurs, and to be unresponsive to all external messages, including requests from the user or the eGate monitor (for status).

3.5.2 HL7 Sequence Numbering Protocol

The eWay can be configured to use HL7 sequence numbering. The negotiation and incrementation of this number is automatically performed by the eWay. For more details on HL7 sequence numbering, refer to Appendix C (Lower Layer Protocols) of the *HL7 Standard* for the HL7 version you are using.

The sequence number file will open at start-up of the eWay. If the sequence number file does not exist, one is created and populated with a zero sequence number.

The sequence number file is updated on the inbound eWay when the eWay generates the HL7 ACK (this process is transparent to the user), and when the outbound eWay receives the HL7 ACK from the external system.

If you want to change the sequence number at runtime, you must suspend the eWay, edit and save the sequence number file, and reactivate the eWay.

To force the eWay to re-synchronize its sequence number with the external system, you must suspend the eWay, edit the file so that it contains a -1, and re-activate the eWay.

The minimum HL7 sequence number is 1. The maximum HL7 sequence number is 2 billion. A sequence number of "0" is used to start a session.

If the sequence numbers between the eWay and the external cannot be reconciled during start or when exchanging messages, the eWay will shutdown and wait for human intervention as dictated by the HL7 Standard.

3.5.3 Failed Message Handling

The eWay may be configured to send failed or Skipped messages (destined for the external system) to a JMS-based error queue. Messages that fail validation are also written to the error queue. Note that the inbound mode of the eWay will not write

messages that fail the MLLP and HLLP validation. These are automatically NAKed and not passed to the Collaboration, but they are logged to the eWay's log file.

The failed or skipped message is written to JMS and the error type and message are written as the JMS properties:

- **Error:** the actual error message or NAK
- **Error Type:** the type of error: for example HL7_NAK_error or HL7_Validation_error.

Skipped messages are those which get continuously NAKed by the external system and thus are Skipped (if the eWay is configured as such).

If the eWay is configured for any other recourse action other than skip, the message remains in the queue.

3.5.4 Recourse Actions

The TCP/IP HL7 eWay Recourse Actions include Reset, Resend, Skip Message, and Exit

On **Reset**, the eWay drops its connection and then attempts to reconnect.

On **Resend**, the sequence number file and journal file are opened again (provided the newly loaded configuration parameters are set for sequence numbering and journaling).

On **Skip Message**, the eWay remains connected, but writes the message to an error queue.

On **Exit**, the eWay closes its journal file and sequence number file (provided these were configured for use). The eWay terminates its connection with the external system and shuts down. This allows the user to modify these files and resolve any errors. Once the corrections are made, the eWay can be reactivated from the Enterprise Manager.

Stopping the Collaboration with a Fatal Alert

When the **Exit** Recourse Action is triggered it logs the error that caused the action. It also shuts down the Collaboration, which in turn causes the HL7 message to roll back and sends an alert to the Enterprise Manager.

The **Exit** Recourse Action calls the fatal alerter in the Collaboration:

```
alerter.fatal('error message', 'HL7');
```

The argument "**error message**" is the user-configured alert message. The argument "**HL7**" is the source component (this must be "HL7").

Note: The `alerter.fatal("error msg", "HL7")` method is only applicable to the packaged TCP/IP HL7 Collaborations.

The **Exit** Recourse Action should be applied to any error condition which requires human intervention to correct an error. Once the error condition is resolved, the Collaboration can be restarted from the Enterprise Manager.

3.6 TCP/IP HL7 eWay Operation

This section explains the basic elements of the TCP/IP HL7 eWay's general operation. These elements are:

- **Direction**
- **Connection Type**
- **Lower Layer Protocol**
- **HL7 Acknowledgment Level** on page 39
- **Journaling** on page 39
- **Error Queues** on page 39
- **Alerts and Monitoring** on page 39
- **Support for HL7 Version 2.5 SFT Segments** on page 39
- **Delayed Acknowledgements** on page 40

3.6.1 Direction

The TCP/IP HL7 eWay can be configured as either **HL7 Inbound** or **HL7 Outbound**. This option is determined automatically by the eWay's binding (link) in the Connectivity Map.

3.6.2 Connection Type

The connection type indicates how the eWay establishes a TCP/IP connection. The role can be as a Client, where the RA connects to the external, or as a Server, where the RA waits for a connection.

Connected as a TCP/IP HL7 Client

As TCP/IP HL7 Client, the eWay connects to external server (host/port) and establishes a connection (in active mode).

Connected as a TCP/IP HL7 Server

The eWay waits and listens to a specific port for incoming connection requests from an external client. Once a request is received, the eWay accepts the request and establishes a connection (in passive mode).

3.6.3. Lower Layer Protocol

This section describes the two supported envelope types used in the HL7 protocol:

- **HLLP** (Hybrid Lower Layer Protocol)
- **MLLP** (Minimal Lower Layer Protocol)

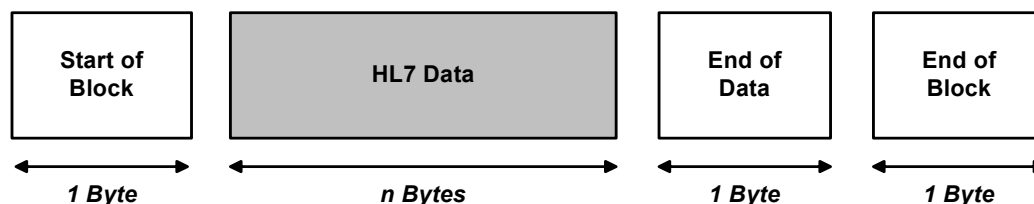
Both envelope types use the following configuration parameters:

- **Start Block Character** on page 61
- **End Data Character** on page 61
- **End Block Character** on page 61

MLLP

The MLLP envelope structure is displayed in Figure 12.

Figure 12 MLLP Envelope Structure

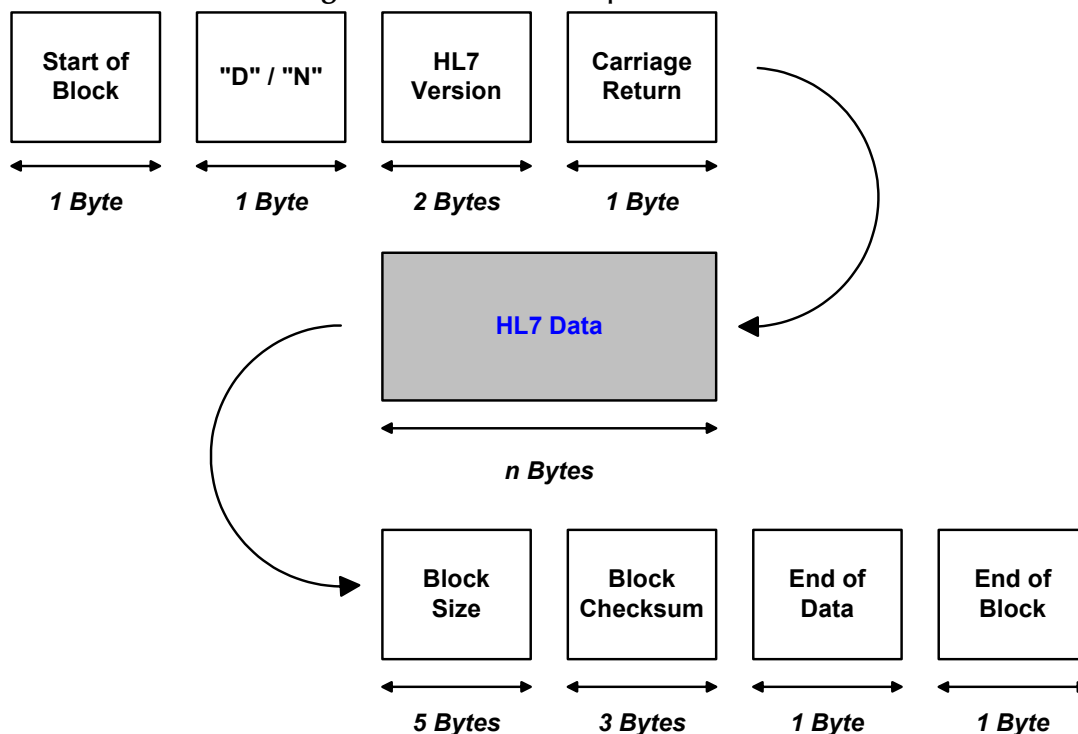


This envelope consists of a **Start of Block** component, a **Data** component, an **End of Data** component, and an **End of Block** component. The size of the **HL7 Data** field is determined by the length of the data (number of bytes between start and end), with a maximum size of 99999 Bytes.

HLLP

The HLLP envelope structure is displayed in [Figure 13 on page 38](#).

Figure 13 HLLP Envelope Structure



This envelope consists of a **Start of Block** component, a **'D'** (Data) or **'N'** (NAK) indicator, an **HL7 Version** component, a **Carriage Return**, a **Data** component, a **Block Size** component, a **Block Checksum** component, an **End of Data** component, and an **End of Block** component. The size of the **HL7 Data** field is determined by the length of the data (number of bytes between start and end), with a maximum size of 99999 Bytes.

3.6.4 HL7 Acknowledgment Level

The eWay supports the sending and receiving of both HL7 acknowledgement types:

- Application acknowledgment: The acknowledgement is sent when the message is successfully received.
- Commit (accept) acknowledgment: The acknowledgement is sent after the message is successfully and functionally processed by one receiving system.

3.6.5 Journaling

The eWay provides the option to journal successfully received or sent messages and their corresponding ACKs. The messages are sent to a JMS queue or topic, as configured by the user, and the ACKs are stored as a JMS property, `HL7_ACK`, of that message.

It is expected that, when enabled, the journal queue has at least one or multiple subscribers that process the contents of the queue so that it remains manageable. For example, the Batch eWay or a database eWay could periodically consume the messages by writing them to a file or a database.

3.6.6 Error Queues

The eWay provides a mechanism to store failed or stripped messages in a JMS queue or topic. The advantage of this is that the messages are then saved in a form readily usable by the other data flows, that can automatically process these messages or make them available to some type of human intervention or message repair, using tools like the JMS queue editor or an eVision application.

3.6.7 Alerts and Monitoring

If the eWay loses the connection to the external system in any direction or connection type, due to a crash, shutdown, or suspension (including recourse actions), an alert is generated. The monitor's status of that eWay is changed to "down" and the eWay's icon is encased in a red warning box. The monitor also displays the number of messages it has processed along with the date and time of the last message sent.

3.6.8 Support for HL7 Version 2.5 SFT Segments

HL7 version 2.5 adds a new SFT segment to every message. The eWay not only sends and receives messages with the new segment, it can automatically create and populate them, using information from the eWay properties, for the outbound message and the

ACK sent from the inbound mode. This feature is only available when the Version ID property is set as 2.5.

3.6.9 Delayed Acknowledgements

The eWay supports Delayed Acknowledgements in either direction and in a number of roles. This functionality is described in detail in [Outbound eWay Roles for Delayed ACK Scenarios](#) on page 32

Configuring TCP/IP HL7 eWay Properties

This chapter explains how to configure the TCP/IP HL7 eWay properties.

What's in This Chapter

- [Creating and Configuring the TCP/IP HL7 eWay](#) on page 41
- [TCP/IP HL7 eWay Configuration Properties](#) on page 44
- [TCP/IP HL7 Inbound eWay Connectivity Map Properties](#) on page 45
- [TCP/IP HL7 Outbound eWay Connectivity Map Properties](#) on page 73
- [TCP/IP HL7 eWay Environment Explorer Properties](#) on page 97

4.1 Creating and Configuring the TCP/IP HL7 eWay

All eWays contain a set of parameters with properties unique to that eWay type. After the eWays are established and a TCP/IP HL7 External System is created in the Project's Environment, the eWay parameters can be modified for your specific system. The TCP/IP HL7 eWay parameters are modified from these locations:

- **Connectivity Map:** These parameters most commonly apply to a specific component eWay, and may vary from other eWays (of the same type) in the Project.
- **Environment Explorer:** These parameters are commonly global, applying to all eWays (of the same type) in the Project. The saved properties are shared by all eWays in the TCP/IP HL7 External System window.
- **Collaboration:** Many TCP/IP HL7 eWay properties can also be set from your Collaboration, in which case the settings will override the corresponding properties in the eWay's configuration file. Any properties that are not overridden retain their configured default settings.

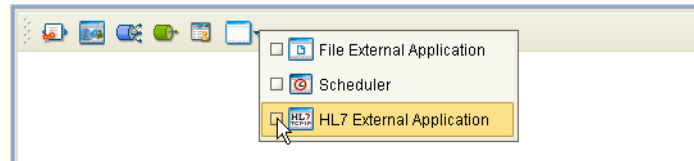
4.1.1 Selecting TCP/IP HL7 as the External Application

To create a TCP/IP HL7 eWay you must first create a TCP/IP HL7 External Application in your Connectivity Map. TCP/IP HL7 eWays are located in the link between a TCP/IP HL7 External Application and a Service. Services are containers for Java Collaborations, Business Processes, eTL processes, and so forth.

To create the TCP/IP HL7 External Application

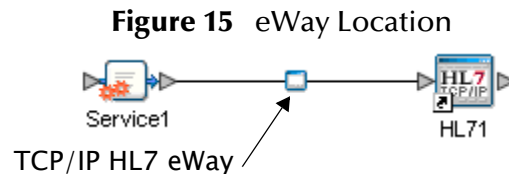
- 1 From the Connectivity Map toolbar, click the **External Applications** icon.
- 2 Select the **TCP/IP HL7 External Application** from the menu (see [Figure 14 on page 42](#)). The selected TCP/IP HL7 External Application icon appears on the Connectivity Map toolbar.

Figure 14 External Applications Selection Menu



- 3 Drag the new **TCP/IP HL7 External Application** from the toolbar onto the Connectivity Map canvas. This represents an external TCP/IP HL7 system.

From the Connectivity Map, you can associate (bind) the External Application with the Service to establish an eWay (see [Figure 15](#)).



When TCP/IP HL7 is selected as the External Application, it automatically applies the default TCP/IP HL7 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**.

4.1.2 Modifying the TCP/IP HL7 eWay Properties

A Project's eWay properties can be modified after the eWays have been established in the Connectivity Map and the Environment has been created.

Modifying the TCP/IP HL7 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 **TCP/IP HL7 eWay Connectivity Map Properties** (inbound or outbound, depending on the link with the service). Make any necessary modifications and click **OK** to save the settings.

Modifying the TCP/IP HL7 eWay (Environment Explorer) Properties

- 1 From the Environment Explorer tree, right-click the **TCP/IP HL7 External System**. Select **Properties** from the shortcut menu (or double-click the **TCP/IP HL7 External System**). The **Properties Editor** opens with the TCP/IP HL7 eWay Environment properties.
- 2 Make any necessary modifications to the Environment properties of the TCP/IP HL7 eWays, and click **OK** to save the settings.

4.1.3 Using the Properties Editor

Modifications to the eWay configuration properties are made from the TCP/IP HL7 eWay Properties Editor.

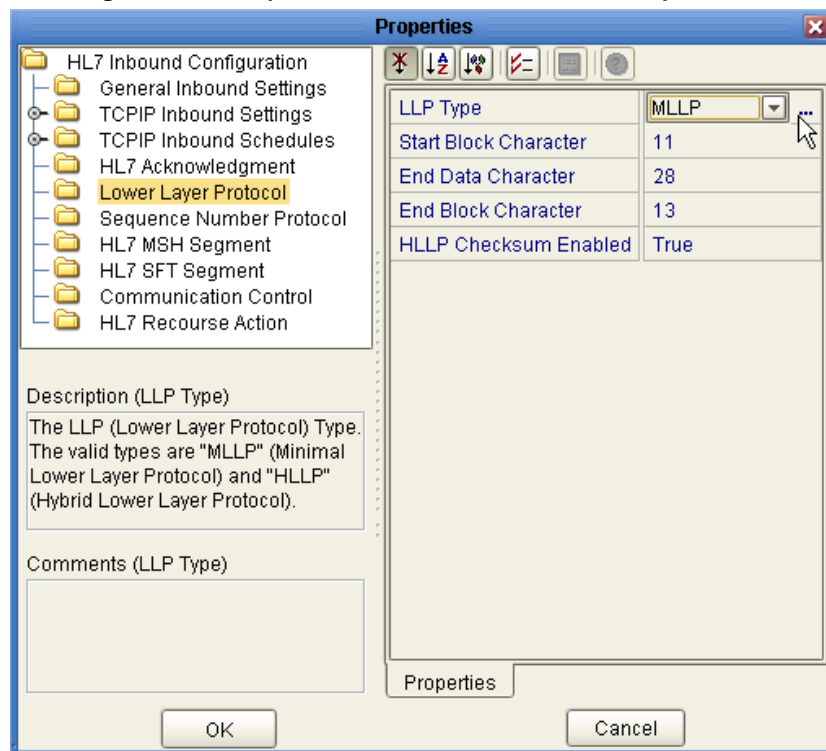
To modify the default eWay configuration properties do the following:

- 1 From the prjHL7Inbound Project's cmHL7Inbound Connectivity Map, double-click the inbound TCP/IP HL7 eWay to open the Properties Editor to the TCP/IP HL7 Inbound eWay's Connectivity Map default properties.

An eWay has two different sets of parameters: those specific to that particular eWay (accessed from the Connectivity Map), and those that are common to all of the Projects all eWays of this type (accessed from the Environment Explorer tree). In addition, the TCP/IP HL7 eWay properties will differ depending on whether the eWay is an Inbound or Outbound eWay.

- 2 From the upper-right pane of the Properties Editor, select a directory under Root Configuration directory. The parameters contained in that directory are now displayed in the Properties pane of the Properties Editor. For example, clicking on the **Lower Layer Protocol** directory displays the editable parameters in the right pane, as shown in Figure 16.

Figure 16 Properties Editor: TCP/IP HL7 Properties



- 3 Click on any property field to make it editable. For example, click on the **LLP Type** parameter to edit its value. If a parameter's value is true/false or multiple choice, the field reveals a submenu of property options.

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.

- 4 A description of each parameter is displayed in the **Description** pane when that parameter is selected, providing an explanation of any required settings or options.
- 5 The **Comments** pane provides an area for recording notes and information regarding the currently selected parameter. This is saved for future referral.
- 6 After modifying the configuration properties, click **OK** to save the changes and close the Properties Editor.

4.2 TCP/IP HL7 eWay Configuration Properties

The TCP/IP HL7 eWay includes four following sets of properties:

- [TCP/IP HL7 Inbound eWay Connectivity Map Properties](#) on page 45
- [TCP/IP HL7 Outbound eWay Connectivity Map Properties](#) on page 73
- [TCP/IP HL7 eWay Environment Explorer Properties](#) on page 97

4.3 TCP/IP HL7 Inbound eWay Connectivity Map Properties

The TCP/IP HL7 Server (inbound) eWay configuration parameters, accessed from the Connectivity Map, are organized into the following sections:

- [General Inbound Settings](#) on page 45
- [TCPIP Inbound Settings](#) on page 47
- [TCPIP Inbound Settings - Server Port Binding](#) on page 51
- [TCPIP Inbound Settings - Client Connection Establishment](#) on page 52
- [TCPIP Inbound Settings - Inbound Connection Management](#) on page 53
- [TCPIP Inbound Schedules - Listener Schedule](#) on page 55
- [TCPIP Inbound Schedules - Service Schedule](#) on page 57
- [HL7 Acknowledgment](#) on page 59
- [Lower Layer Protocol](#) on page 61
- [Sequence Number Protocol](#) on page 62
- [HL7 MSH Segment](#) on page 63
- [HL7 SFT Segment](#) on page 66
- [Communication Control](#) on page 68
- [HL7 Recourse Action](#) on page 70

4.3.1 General Inbound Settings

The **General Inbound Settings** section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 3.

Table 3 Connectivity Map - General Inbound Settings

Name	Description	Required Value
Max Data Size	Specifies the maximum amount of data that the programs can hold internally. The valid range is from 1 to 2GB (which is the maximum value of a Java integer).	A number indicating the maximum amount of data. The valid range is from 1 to 2147483647 (bytes). The configured default is 2147483647 .

Table 3 Connectivity Map - General Inbound Settings (Continued)

Name	Description	Required Value
Scope Of State	<p>Defines the scope of State object, which is an OTD node. The options for this parameter are:</p> <ul style="list-style-type: none"> ▪ Resource Adapter Level: The State has the same life cycle as the resource adapter. ▪ Connection Level: The State has the same life cycle as the connection. ▪ OTD Level: The State has the same life cycle as the OTD object. <p>This scope represents the life cycle of the State.</p>	<p>Select one of the following:</p> <ul style="list-style-type: none"> ♦ Resource Adapter Level ♦ Connection Level ♦ OTD Level <p>The configured default is Resource Adapter Level.</p>
Dedicated Session Mode	<p>Specifies whether the server Dedicated Session Mode is enabled or disabled. When the server Dedicated Session Mode is enabled, the current client's request exclusively holds the server port to which it connects. The next client's request to the same port is blocked or rejected until the previous request concludes and releases the connection.</p>	<p>Select True or False. True indicates that Dedicated Session Mode is enabled.</p> <p>The configured default is False.</p>

4.3.2 TCPIP Inbound Settings

The **TCPIP Inbound Settings** section presents the java Socket and ServerSocket options. This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 4.

Table 4 Connectivity Map - TCPIP Inbound Settings

Name	Description	Required Value
Connection Type	<p>Specifies how the eWay establishes the TCP/IP connection:</p> <ul style="list-style-type: none"> ▪ Client: The eWay connects to an external server (host/port) to establish the connection. The eWay is in active mode. ▪ Server: The eWay waits/listens on a certain port for an incoming connection request from an external client. Once the request is received, the eWay accepts the request and establishes the connection. The eWay is in passive mode. 	Client or Server . Server is the default setting. Unless you specifically require Client mode, leave this value as the default: Server .
ServerSO Timeout	Sets or gets the value of the SO_TIMEOUT for ServerSocket , in milliseconds. Used for ServerSocket.accept() . When this option is set to a non-zero timeout, calling accept() for this ServerSocket will block for the configured length of time. If the timeout expires, a java.net.SocketTimeoutException (or java.net.InterruptedIOException) is thrown, but the ServerSocket remains valid. Enable this option prior to entering the blocking operation. This parameter is only used when the Connection Type is set as Server	<p>The a number indicating the Server SO Timeout in milliseconds.</p> <p>The timeout must be greater than zero (0). A timeout of zero is interpreted as an infinite timeout.</p> <p>The configured default is 10000 (10 seconds).</p>
Server Socket Factory Implementation Class Name	Specifies the name of the Java class that implements the server socket factory. This class is used to create the server socket. If you have provided your own server socket implementation, enter the name of the Java class that contains this implementation here. The factory implementation class must implement the following interface: <code>com.stc.connector.tcpip.model.factory.TCPIPConnectionFactory</code>	<p>A valid Java class name; the default is:</p> <pre>com.stc.connector.tcpip.model.factory.TCPIPConnectionFactoryImpl</pre>

Table 4 Connectivity Map - TCPIP Inbound Settings (Continued)

Name	Description	Required Value
Keep Alive	<p>Specifies whether the client's SO_KEEPALIVE option is enabled or disabled. When the option is set for a TCP socket and no data has been exchanged across the socket in either direction for 2 hours, TCP automatically sends a KEEPALIVE probe to the peer (the actual value is implementation dependent). This probe is a TCP segment to which the peer must respond. One of three responses is expected:</p> <ol style="list-style-type: none"> 1 The peer responds with the expected ACK. The application is not notified (since everything is OK). TCP will send another probe following another 2 hours of inactivity. 2 The peer responds with an RST, which tells the local TCP that the peer host has crashed and rebooted. The socket is closed. 3 There is no response from the peer. The socket is closed. The purpose of this option is to detect if the peer host has crashed. This is used for the accepted client Socket. 	<p>True or False. True indicates that server SO_KEEPALIVE option is enabled.</p> <p>Note: For some properties, the server socket itself does not have direct properties settings associated with it. Instead, the properties map to the accepted client socket.</p>
Receive Buffer Size	<p>Sets or gets the value of the SO_RCVBUF option for the current socket, that is the buffer size used by the operating system for input on this socket. It provides an estimate of the size of the underlying buffers used by the platform for incoming network I/O. When used in set, this is a suggestion for the kernel from the application regarding the size of buffers to use for the data to be received over the socket. When used in get, this must return the actual size of the buffer used by the platform when receiving data on this socket.</p>	<p>A number indicating the receive buffer size.</p> <p>The configured default is 8192.</p>
Send Buffer Size	<p>Sets or gets the value of the SO_SNDBUF option for the current socket, that is the buffer size used by the operating system for output on this socket. It provides an estimate of the size of the underlying buffers used by the platform for outgoing network I/O. When used in set, this is a suggestion for the kernel from the application regarding the size of buffers to use for the data to be sent over the socket. When used in get, this must return the actual size of the buffer used by the platform when sending out data on this socket.</p>	<p>A number indicating the send buffer size.</p> <p>The configured default is 8192.</p>

Table 4 Connectivity Map - TCPIP Inbound Settings (Continued)

Name	Description	Required Value
SoLinger	<p>Specifies whether the eWay performs a linger-on-close timeout. This option disables/enables immediate return from a close() of a TCP Socket. This parameter is used in conjunction with SoLinger Timeout.</p> <ul style="list-style-type: none"> ▪ True with SoLinger Timeout set to a non-zero integer timeout: This means that a close() will block, pending the transmission and acknowledgement of all data written to the peer, at which point the socket is closed gracefully. Upon reaching the linger timeout, the socket is closed forcefully with a TCP RST. ▪ True with SoLinger Timeout set to a timeout of zero: Indicates that a forceful close is done immediately. See “SoLinger Timeout” on page 49. 	True or False. True enables the SO_Linger option.
SoLinger Timeout	<p>Specifies the server's SoLinger time-out in seconds. SoLinger Timeout is used in conjunction with SoLinger (see “SoLinger” on page 49) to configure the “linger-on-close” timeout.</p> <p>When SoLinger is set to true (enabled), the SoLinger Timeout value indicates the following:</p> <ul style="list-style-type: none"> ▪ A non-zero integer means that calling close() will block, pending the transmission and acknowledgement of all data written to the peer, at which point the socket is closed gracefully. Upon reaching the linger timeout, the socket is closed forcefully with a TCP RST. If the specified timeout value exceeds 65,535 it will be reduced to 65,535. ▪ A zero integer indicates that a forceful close is done immediately. 	<p>An integer between -1 and 65535. The default is -1 seconds, which indicates that the SoLinger option is disabled (set as false).</p> <p>Zero (0) indicates that SoLinger immediately performs a forceful close. An integer of 1 to 65535 indicates the number of seconds for the time-out.</p>

Table 4 Connectivity Map - TCPIP Inbound Settings (Continued)

Name	Description	Required Value
SoTimeout	<p>Sets or gets the value of the SoTimeout in milliseconds. Used for the accepted client socket. With this option set to a non-zero timeout, calling read() on the InputStream associated with this socket will block for only the configured length of time. If the timeout expires, a java.io.InterruptedIOException (or java.net.SocketTimeoutException) is thrown, but the Socket remains valid.</p> <p>Enable this option prior to entering the blocking operation.</p>	<p>The SoTimeout value in milliseconds. The configured default is 10000 (10 seconds).</p> <p>The timeout must be greater than 0 (zero). A timeout of zero is interpreted as an infinite timeout.</p>
TcpNoDelay	<p>Specifies whether the server's TcpNoDelay option (that is, Nagle's algorithm) is enabled or disabled.</p> <ul style="list-style-type: none"> ▪ True: Indicates that the server allows data packets that are less than the maximum transfer unit (MTU) size to be sent out immediately over the network. A setting of True may improve performance for higher-speed networks. ▪ False: Indicates that the server does not allow data packets that are less than the MTU size be sent out immediately over the network. <p>This is used for the accepted client socket.</p>	<p>True or False.</p> <p>The configured default is False.</p>

4.3.3 TCPIP Inbound Settings - Server Port Binding

The **TCPIP Inbound Settings - Server Port Binding** section defines the parameters used for controlling the server port binding. This section is only used when the **Connection Type** is set as **Server**.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 5.

***Note:** This section is only used when the Connection Type is set as Server.*

Table 5 Connectivity Map - TCPIP Inbound Settings - Server Port Binding

Name	Description	Required Value
Max Binding Retry	Specifies the maximum number of times the eWay will attempt to bind to the specified TCP/IP port on the localhost.	An integer indicating the number of bind attempts. The configured default is 3 .
Retry Binding Interval	Specifies the length of time (in milliseconds) the eWay waits between attempts to bind to the specified TCP/IP port on the localhost.	An integer indicating the length of time in milliseconds that the eWay waits between attempts. The configured default is 30000 (30 seconds).

4.3.4 TCPIP Inbound Settings - Client Connection Establishment

The **TCPIP Inbound Settings - Client Connection Establishment** section defines some configuration parameters used for controlling the connection establishment. This section is only used when the **Connection Type** is set as **Client**.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 6.

***Note:** This section is only used when the Connection Type is set as Client.*

Table 6 Connectivity Map - TCPIP Inbound Settings - Client Connection Establishment

Name	Description	Required Value
Time to Wait Before Attempting Connection	Specifies the length of time (in milliseconds) that the eWay waits before attempting to connect to the external system.	A number indicating the amount of time (in milliseconds) to wait before attempting to connect to an external system. The configured default is 30000 (30 seconds).

4.3.5 TCPIP Inbound Settings - Inbound Connection Management

The **TCPIP Inbound Settings - Inbound Connection Management** section defines the parameters used for inbound Server Connection Management. For example, the connection pool and the life cycle of the accepted connection.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 7.

Table 7 Connectivity Map - TCPIP Inbound Settings - Inbound Connection Management

Name	Description	Required Value
Max Connection Pool Size	Specifies the maximum number of concurrent connections allowed for the specific listener/monitor which is listening on, or monitoring a specified TCP/IP port. This represents the capability or availability of this server's services. Each connect-request from a client gains one concurrent connection. This parameter also represents the maximum number of clients who can concurrently connect to this server's services, and get served by the specific listener/monitor at the same time.	A number indicating the maximum number of concurrent connections available from a listener/monitor for a specific TCP/IP port. 0 indicates that there is no limit. The configured default is 50 .
Scope Of Connection	Specifies the scope of the accepted connection which is used by the eWay. The two options are: <ul style="list-style-type: none"> ▪ Resource Adapter Level: The resource adapter will close the connection upon closure request (by way of <code>ClosureCommandMessage</code>), so the connection may "keep alive" during multiple executions of the Collaboration. Collaboration Level: The connection is closed once the Collaboration has been executed, so the connection has the same life cycle as the Collaboration.	Select Resource Adapter Level or Collaboration Level . The configured default value is Resource Adapter Level .
Close Notification	Specifies the close notification value. When the server receives a notification with content that matches this parameter's value, the server safely closes the connection and cancels any corresponding schedules.	A String indicating the trigger value that notifies the server to close the connection. The configured default is QUIT .

Table 7 Connectivity Map - TCPIP Inbound Settings - Inbound Connection Management

Name	Description	Required Value
Idle Timeout	Specifies the length of time (in milliseconds) for inactivity of the requestor (client). The eWay attempts to detect activity on client side (the other side of the connection). If no client activity (no i/o request comes over the connection from the client) for a specified time period, then the connection is closed from the server side to release the resource. The value is in milliseconds. If you want to disable this IdleTimeout checking, just specify 0 for this parameter.	An integer that indicates the length of time in milliseconds. A value of 0 disables IdleTimeout. The configured default is 60000 .

4.3.6 TCPIP Inbound Schedules - Listener Schedule

This section configures the scheduler used by the inbound TCP/IP Server. The server waits for a new client connection establishment request. These parameters are used to configure the listener/monitor that listens on the specified port.

Two J2EE schedulers are available (see [“Scheduler” on page 55](#)):

- **Timer Service:** available for J2EE. This scheduler is configured using the **At Fixed Rate**, **Delay**, and **Period** properties.
- **Work Manager:** available for J2EE (JCA 1.5 and above). This scheduler is configured using the **Delay** and **Period** properties.

Both schedulers provide the functionality required by the inbound TCP/IP Server.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 8.

Table 8 Connectivity Map - TCPIP Inbound Schedules - Listener Schedule

Name	Description	Required Value
Scheduler	Specifies the scheduler type for this inbound communication. There are two options: <ul style="list-style-type: none"> ▪ Timer Service: The task is scheduled through the J2EE Timer Service. Timer Service is supported by J2EE. ▪ Work Manager: The task is scheduled through the J2EE Work Manager. Work Manager is supported by J2EE (JCA 1.5 and above). 	Select Timer Service or Work Manager . If your container doesn't support JCA Work Manager, select Timer Service.
Schedule Type	This property configuration, though visible from the Properties Editor, is disabled. The only available schedule type is Repeated, indicating that the task is scheduled for repeated execution at regular intervals defined by the Period property in this section (see “Period” on page 55).	This field is not enabled.
Delay	Applies to both the Timer Service or the Work Manager . Specifies, in milliseconds, the length of delay time before the task is executed.	An integer indicating the length of time before the task is executed, in milliseconds.
Period	Applies to both the Timer Service or the Work Manager . Specifies the regular interval, in milliseconds, between successive task executions.	An integer indicating the length of time between successive task executions, in milliseconds.

Table 8 Connectivity Map - TCPIP Inbound Schedules - Listener Schedule (Continued)

Name	Description	Required Value
At Fixed Rate	<p>Specific to the Timer Service configuration only. Specifies whether a Fixed-Rate execution or Fixed-Delay execution is used.</p> <ul style="list-style-type: none"> ▪ Fixed-Rate: A fixed-rate execution means that each execution is scheduled relative to the scheduled time of the initial execution. If an execution is delayed for any reason (such as garbage collection or other background activity), two or more executions will occur in rapid succession to “catch up.” In the long run, the frequency of execution will be exactly the reciprocal of the specified period (assuming the system clock underlying <code>Object.wait(long)</code> is accurate). ▪ Fixed-Delay: A fixed-delay execution means that each execution is scheduled relative to the actual time of the previous execution. If an execution is delayed for any reason (such as garbage collection or other background activity), subsequent executions will be delayed as well. As a result, the frequency of execution will generally be slightly lower than the reciprocal of the specified period, assuming the system clock underlying <code>Object.wait(long)</code> is accurate. 	True or False. True indicates that a fixed-rate execution is used. False indicates that a fixed-delay execution is used.

4.3.7 TCPIP Inbound Schedules - Service Schedule

This section configures the scheduler used by the TCP/IP Server that executes the business tasks (Collaboration Rules) over the existing connection. This scheduler affects the actual Business Rules defined by the user.

Two J2EE schedulers are available (see [“Scheduler” on page 55](#)):

- **Timer Service:** available for J2EE. This scheduler is configured using the **At Fixed Rate, Delay, Period,** and **Schedule Type**, properties.
- **Work Manager:** available for J2EE (JCA 1.5 and above). This scheduler is configured using the **Delay, Period,** and **Schedule Type**, properties.

Both schedulers provide the functionality required by the inbound TCP/IP Server.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 9.

Table 9 Connectivity Map - TCPIP Inbound Schedules - Server Schedule

Name	Description	Required Value
Scheduler	Specifies the scheduler type for this inbound communication. There are two options: <ul style="list-style-type: none"> ▪ Timer Service: The task is scheduled through the J2EE Timer Service. Timer Service is supported by J2EE. ▪ Work Manager: The task is scheduled through the J2EE Work Manager. Work Manager is supported by J2EE (JCA 1.5 and above). 	Select Timer Service or Work Manager . If your container doesn't support JCA Work Manager, select Timer Service.
Schedule Type	Applies to both the Timer Service or the Work Manager . Specifies whether the task is scheduled to occur once or be repeated. <ul style="list-style-type: none"> ▪ OneTime: The task will be scheduled for one-time execution. ▪ Repeated: The task will be scheduled for repeated execution at regular intervals defined by Period property in this section (see “Period” on page 55). 	Select OneTime or Repeated .
Delay	Applies to both the Timer Service or the Work Manager . Specifies, in milliseconds, the length of delay time before the task is executed.	An integer indicating the length of time, in milliseconds, before the task is executed.
Period	Applies to both the Timer Service or the Work Manager . Specifies the regular interval, in milliseconds, between successive task executions. This is used for the Repeated schedule type (see “Schedule Type” on page 55)	An integer indicating the length of time, in milliseconds, between successive task executions.

Table 9 Connectivity Map - TCPIP Inbound Schedules - Server Schedule (Continued)

Name	Description	Required Value
At Fixed Rate	<p>Specific to the Timer Service configuration only. Specifies whether a Fixed-Rate execution or Fixed-Delay execution is used. This is used for the "Repeated" schedule type by the "Timer Service" scheduler.</p> <ul style="list-style-type: none"> ▪ Fixed-Rate: A fixed-rate execution means that each execution is scheduled relative to the scheduled time of the initial execution. If an execution is delayed for any reason (such as garbage collection or other background activity), two or more executions will occur in rapid succession to "catch up." In the long run, the frequency of execution will be exactly the reciprocal of the specified period (assuming the system clock underlying <code>Object.wait(long)</code> is accurate). ▪ Fixed-Delay: A fixed-delay execution means that each execution is scheduled relative to the actual time of the previous execution. If an execution is delayed for any reason (such as garbage collection or other background activity), subsequent executions will be delayed as well. As a result, the frequency of execution will generally be slightly lower than the reciprocal of the specified period, assuming the system clock underlying <code>Object.wait(long)</code> is accurate. 	True or False. True indicates that a fixed-rate execution is used. False indicates that a fixed-delay execution is used.

4.3.8 HL7 Acknowledgment

Specifies how the application acknowledgment Events are handled. This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 10.

Table 10 Connectivity Map - HL7 Acknowledgment

Name	Description	Required Value
Acknowledgment Level	<p>Specifies whether the external application sends an HL7 application Acknowledgement after successfully receiving a message, or after the message has been successfully committed to the application database. The options are:</p> <ul style="list-style-type: none"> ▪ A: Application acknowledgment. The acknowledgement is sent when the message is successfully received. ▪ C: Commit (accept) acknowledgment. The acknowledgement is sent after the message is successfully and functionally processed by one receiving system. 	<p>A or C.</p> <p>The configured default is A.</p>
eGate Sends App Acks	<p>Specifies whether the HL7 application acknowledgment sent to the external system is generated by the eWay or forwarded from eGate.</p> <ul style="list-style-type: none"> ▪ True indicates that eGate receives or creates the HL7 application acknowledgment and sends it to the eWay, which in turn forwards it to the external system. ▪ False indicates that the eWay creates and sends the HL7 application acknowledgment directly to the external system. <p>This parameter is used for inbound Collaboration code.</p>	<p>True or False.</p> <p>The configured default is False.</p>

Table 10 Connectivity Map - HL7 Acknowledgment (Continued)

Name	Description	Required Value
Forward External Acks to eGate	<p>Specifies whether the HL7 application acknowledgment is forwarded to eGate. When an HL7 application acknowledgment is received, it is sometimes necessary to forward the contents of the HL7 application acknowledgment to eGate (as data).</p> <ul style="list-style-type: none"> ▪ True indicates that eWay forwards HL7 application acknowledgments from the external system to eGate for processing. ▪ False indicates that HL7 application acknowledgments from the external system are not forwarded to eGate by the eWay. <p>This parameter is used for inbound Collaboration code.</p>	<p>True or False.</p> <p>The configured default is False.</p>
Timeout For Delayed Ack	<p>Specifies the timeout value for delayed ACK in milliseconds.</p> <p>This parameter is used for the inbound Collaboration code.</p>	<p>A number indicating the timeout in milliseconds.</p> <p>The configured default is 30000 (30 seconds).</p>

4.3.9 Lower Layer Protocol

Provides Lower Layer Protocol (LLP) configuration settings.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 11.

Table 11 Connectivity Map - Lower Layer Protocol

Name	Description	Required Value
LLP Type	Specifies the LLP (Lower Layer Protocol) type. The valid types are: <ul style="list-style-type: none"> ▪ MLLP (Minimal Lower Layer Protocol) ▪ HLLP (Hybrid Lower Layer Protocol) For more information on the available envelope types see “Lower Layer Protocol” on page 37 .	MLLP or HLLP . MLLP is the configured default value.
Start Block Character	Specifies the Start Block Character (the first envelope marker character in the HL7 envelope) as a decimal ascii number.	A decimal within the range of 1 to 127 . Unless there is a conflict, the value should be ASCII VT (decimal 11). The default value is 11 .
End Data Character	Specifies the End Data Character (The second to the last envelope marker character in the HL7 envelope) as a decimal ASCII number. The allowed range is 1 to 127.	A decimal within the range of 1 to 127 . Unless there is a conflict, the value should be ASCII FS (decimal 28). The default value is 28 .
End Block Character	Specifies the End Block Character (the last envelope marker character in the HL7 envelope) as a decimal ASCII number.	A decimal within the range of 1 to 127 . To be strictly compliant with the HL7 Standard, this parameter MUST be set to a Carriage Return (decimal 13). The default value is 13 .
HLLP Checksum Enabled	Specifies whether the HLLP (Hybrid Lower Level Protocol) Checksum is enabled or disabled.	True or False . True indicates that the HLLP Checksum is enabled.

4.3.10. Sequence Number Protocol

Provides sequence number protocol configuration settings. HL7 sequence numbering is used to help prevent duplication of data.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 12.

Table 12 Connectivity Map - Sequence Number Protocol

Name	Description	Required Value
Sequence Number Enabled	Specifies whether Sequence Number Protocol is enabled or disabled. HL7 sequence numbering is used to help prevent duplication of data. True indicates that sequence numbering is enabled.	True or False . The configured default is True .

4.3.11 HL7 MSH Segment

Provides HL7 MSH Header segment configuration settings.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 13.

Table 13 Connectivity Map - HL7 MSH Segment

Name	Description	Required Value
Field Separator	<p>Specifies the separator between the segment ID and the first real field. This value defines the character that is used as a separator for the rest of the message.</p> <p>This is the first field in the HL7 MSH segment (MSH-01).</p>	<p>Field Separator character value as a decimal ASCII number. The allowed range is 1 to 127.</p> <p>The default setting is 124 which is the character ' '.</p>
Encoding Characters	<p>Specifies four encoding characters in the following order:</p> <ul style="list-style-type: none"> Component separator Repetition separator Escape character Subcomponent separator <p>This is the second field in the HL7 MSH segment (MSH-02).</p>	<p>HL7 encoding characters in the respective order.</p> <p>The configured default is: ^~\& (ASCII 94, 126, 92, and 38, respectively).</p>
Sending Application	<p>Specifies the sending application among other applications within the network enterprise. The network enterprise consists of the applications that participate in the exchange of HL7 messages within the enterprise.</p> <p>This is the third field in the HL7 MSH segment (MSH-03).</p>	<p>User defined value for the HL7 sending application.</p> <p>The Configured default is Sun SeeBeyond HL7 eWay.</p>
Sending Facility	<p>Specifies (further identifies) the sending application among multiple identical instances of the application running on behalf of different organizations.</p> <p>This is the forth field in the HL7 MSH segment (MSH-04).</p>	<p>User defined value for the HL7 sending facility.</p> <p>The Configured default is Sun SeeBeyond HL7 eWay.</p>

Table 13 Connectivity Map - HL7 MSH Segment (Continued)

Name	Description	Required Value
Receiving Application	<p>Specifies the receiving application among other applications within the network enterprise. The network enterprise consists of the applications that participate in the exchange of HL7 messages within the enterprise.</p> <p>This is the fifth field in the HL7 MSH segment (MSH-05).</p>	<p>User defined value for the HL7 receiving application.</p> <p>The Configured default is Sun SeeBeyond HL7 eWay.</p>
Receiving Facility	<p>Specifies (further identifies) the receiving application among multiple identical instances of the application running on behalf of different organizations.</p> <p>This is the sixth field in the HL7 MSH segment (MSH-06).</p>	<p>User defined value for the HL7 receiving facility.</p> <p>The Configured default is Sun SeeBeyond HL7 eWay.</p>
Security	<p>Specifies the implemented application level security features.</p> <p>This is the eighth field in the HL7 MSH segment (MSH-08).</p>	<p>Under development by HL7.</p>
Processing ID	<p>Specifies the sub-component Processing ID of MSH-11. MSH-11 is used to indicate whether a message is processed as defined in the HL7 Application (level 7) Processing rules.</p>	<p>Requires one of the following:</p> <ul style="list-style-type: none"> ▪ D - for Debugging ▪ P - for Production ▪ T - for Training <p>In some cases there may be an additional sub-component "Processing Mode" following the initial value. The configured default is P.</p>
Version ID	<p>Specifies the particular HL7 version to be matched by the receiving system to its own version.</p> <p>This is the 12th field in the HL7 MSH segment (MSH-12).</p>	<p>The HL7 Standard version value as displayed in HL7 Table 0104 - Version ID.</p> <p>The configured default value is 2.5.</p>

Table 13 Connectivity Map - HL7 MSH Segment (Continued)

Name	Description	Required Value
Country Code	<p>Specifies a code that indicates the country of origin for the message (see HL7 Table 0399). Used to specify default elements in a message, such as currency.</p> <p>This is the 17th field in the HL7 MSH segment (MSH-17).</p>	<p>The Country Code value uses the 3-character (alphabetic) form of ISO 3166.</p> <p>The default value is USA.</p>
Character Set	<p>Specifies the character set(s) in use by the messages (see HL7 Table 0211). If the field is left blank, the character set in use is understood to be the 7-bit ASCII set.</p> <p>This is the 18th field in the HL7 MSH segment (MSH-18).</p>	<p>The configured default is 8859/1 (printable 7-bit ASCII character set).</p> <p>See HL7 Table 0211 for available values and descriptions.</p>
Principal Language of Message	<p>Specifies the principal language of the message. Codes come from ISO 639.</p> <p>This is the 19th field in the HL7 MSH segment (MSH-19).</p>	<p>The 2-character ISO 639 alphabetic code.</p>
Alternate Character Set Handling Scheme	<p>Specifies the value for the alternate character set handling scheme to be used when any alternative character sets are used and a special handling scheme is necessary (see HL7 Table 0356).</p> <p>This is the 20th field in the HL7 MSH segment (MSH-20).</p>	<p>Available values include ISO 2022-1994, 2.3, or <null> (blank). Leaving the field blank indicates that no character set switching will occur.</p>
Conformance Statement ID	<p>The Conformance Statement ID (Message Profile Identifier in V2.5) is a unique identifier that applies to a query's Conformance Statement, or as a Message Profile Identifier, asserts constancy with a message profile (grammar, syntax, usage, and so forth).</p> <p>This is the 21st field in the HL7 MSH segment (MSH-21).</p>	<p>An HL7 Conformance Statement ID value or leave blank.</p>
Validate MSH	<p>Specifies whether to validate the MSH segment of the data message (for inbound) and the MSH segment of the ACK (for outbound).</p> <p>This parameter is used for inbound Collaboration code.</p> <p>Note: This property does not affect structural validation of the whole HL7 message itself. Structural validation is always performed.</p>	<p>True or False. True indicates that the Collaboration validates the MSH segment.</p> <p>The configured default is True.</p>

4.3.12 HL7 SFT Segment

Provides the HL7 SFT Segment configuration settings. The SFT segment is available starting with HL7 version 2.5. This segment provides additional information about one or more software products used as sending applications. The primary purpose of this segment is for diagnostic use. There may be additional uses per site-specific agreements.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 14.

Table 14 Connectivity Map - HL7 SFT Segment

Name	Description	Required Value
Enable	Specifies whether the SFT optional segment is enabled in the ACK. Note: If enable is set to true, and the HL7 version is not configured as 2.5, the eWay will error upon startup.	True or False. True indicates that the SFT segment is enabled in the ACK. Note: This property only applies to HL7 version 2.5.
Software Vendor Organization	Specifies HL7 segment SFT-01 , the name of the company that publishes and/or distributes the sending software that created the transaction. The purpose of this field, along with the remaining fields in this segment, is to provide a more complete profile of the sending applications. The Software Vendor Organization field identifies the vendor who is responsible for maintaining the application.	The name of the sending software publisher or vendor. The configured default is Sun Microsystems, Inc. Note: This property only applies to HL7 version 2.5.
Software Certified Version or Release Number	Specifies HL7 segment SFT-02 , the Software Certified Version or Release Number . The latest software version number or release number for the sending system, helps to provide a more complete profile of the application that is sending or receiving HL7 messages. Version numbers are important in identifying the specific release of an application. In some situations, the receiving application validates the software certified version or release number against a list of “certified” versions or releases of the particular software. This helps determine whether the sending application adheres to specific Business Rules required by the receiving application. Alternatively, the software may perform different processing, depending on the version of the sending software.	The software certified version or release number. The configured default is 5.1.1. Note: This property only applies to HL7 version 2.5.

Table 14 Connectivity Map - HL7 SFT Segment (Continued)

Name	Description	Required Value
Software Product Name	Specifies HL7 segment SFT-03 , the name of the software product that submitted the transaction. The software product name is a key component for identifying the sending application.	<p>The sending software product name.</p> <p>The default value is Sun SeeBeyond TCP/IP HL7 eWay Intelligent Adapter.</p> <p>Note: This property only applies to HL7 version 2.5.</p>
Software Binary ID	Specifies HL7 segment SFT-04 , the Software Binary ID. This property is available starting with HL7 version 2.5. Software Binary IDs are issued by a vendor for each unique software version instance. These IDs are used to differentiate between differing versions of the same software. Identical Primary IDs indicate that the software is identical at the binary level, but configuration settings may differ.	<p>The unique Software Binary ID.</p> <p>The configured default is 5.1.1.</p> <p>Note: This property only applies to HL7 version 2.5.</p>
Software Product Information	<p>Specifies HL7 segment SFT-05, software product identification information. This may include a description of the software application, configuration settings, modifications made to the software, and so forth.</p> <p>This field can contain any additional information about the sending application, with the transaction it has submitted. The information is used for diagnostic purposes and may provide greater flexibility for identifying the application software.</p>	<p>Information that may help to identify the specific sending software. This field should only be used when performing diagnostics.</p> <p>The default value is "It is a JCA adapter for HL7 over TCP/IP connection."</p> <p>Note: This property only applies to HL7 version 2.5.</p>
Software Install Date	Specifies HL7 segment SFT-06 , the Software Install Date. This is the date on which the submitting software was installed at the sending site. the software install date on its own can often provide key information about the behavior of the application. This is necessary for providing a more complete profile of the sending application.	<p>The date of installation for the sending application software.</p> <p>Note: This property only applies to HL7 version 2.5.</p>

4.3.13 Communication Control

The **Communication Control** section controls data transferring (sending/receiving) over the TCP/IP connection.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 15.

Table 15 Connectivity Map - Communication Control

Name	Description	Required Value
Time To Wait For A Response	<p>Specifies the amount of time (in milliseconds) that the eWay waits for a response from the external system before taking recourse action (see “Action on No Response” on page 70). Any data from the external system is considered a response.</p> <p>This property corresponds to the initial read/receive operation timeout. Once a response is received, the following read/receive operation uses the SoTimeout specified timeout (see “SoTimeout” on page 50). Value 0 is interpreted as an infinite timeout.</p>	<p>An integer indicating the length of time in milliseconds that the eWay waits for a response to arrive from the external system. A value of 0 (zero) is interpreted as an infinite timeout.</p> <p>The configured default is 30000 (30 seconds).</p>
Max Empty Read Retry	<p>Specifies the maximum number of times the eWay attempts to read data from the external system after the read/receive operation returns nothing. This applies to the read or receive operation after a response starts to arrive. Empty Read means that a timeout occurs on the read/receive operation, which takes the SoTimeout parameter in the TCPIP Server Base Settings section as the applied timeout setting (see “SoTimeout” on page 50). The corresponding recourse action is specified by the “Action on Max Failed Read Retry” on page 71</p>	<p>A number indicating the maximum number or retries.</p> <p>The configured default is 5.</p>
Max No Response	<p>Specifies the maximum number of response timeouts the eWay allows, while waiting for data from the external system, before taking recourse action (see “Action on Max No Response” on page 70).</p> <p>This parameter is used for the inbound Collaboration code. This parameter is only used by outbound eWays and works in conjunction with the Resend option of the Recourse Action parameter Action on No Response (see “Action on No Response” on page 70). It configures the eWay to resend the last message for the specified maximum number of times before the subsequent recourse action is taken.</p>	<p>An integer indicating the appropriate number of timeouts that may occur before taking recourse action.</p> <p>The configured default is 30.</p>

Table 15 Connectivity Map - Communication Control (Continued)

Name	Description	Required Value
Max NAK Receive Retry	<p>Specifies the maximum number of negative acknowledgments the eWay receives before taking recourse action (see “Action on Max Nak Received” on page 72).</p> <p>This parameter is used for the inbound Collaboration code.</p>	<p>A number indicating the appropriate maximum number of NAKs received before taking recourse action.</p> <p>The default value is 30.</p>
Max NAK Send Retry	<p>Specifies the maximum number of negative acknowledgments the eWay sends before taking recourse action (see “Action on Max Nak Sent” on page 72).</p> <p>This parameter is used for the inbound Collaboration code.</p>	<p>An integer indicating the appropriate maximum number of NAKs sent by the eWay before recourse action is taken.</p> <p>The default value is 30.</p>
Max Canned NAK Send Retry	<p>Specifies the maximum number of canned negative acknowledgments that the eWay sends before taking recourse action (see “Action on Max Nak Sent” on page 72).</p>	<p>The appropriate maximum number of canned NAK to send before taking recourse action. 0 indicates that the eWay will not attempt to create or send a canned NAK.</p> <p>The configured default is 3.</p>
Enable Journaling	<p>Specifies whether message journaling is enabled.</p> <p>This parameter is used for inbound Collaboration code.</p>	<p>True or False. True indicates that journaling is enabled.</p> <p>The configured default is True.</p>

4.3.14 HL7 Recourse Action

The **HL7 Recourse Action** section determines the actions the eWay takes when operations occur outside the configured constraints.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 16.

Table 16 Connectivity Map - HL7 Recourse Action

Name	Description	Required Value
Action on No Response	<p>Specifies the action taken by the eWay when no ACK is received from the external system in the allotted time. The amount of time is determined by the Time To Wait For A Response parameter (see “Time To Wait For A Response” on page 68). The options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Resend: The eWay attempts to resend the message to the external system. The Resend option is only allowed when sequence numbering is in effect. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. <p>This parameter is used for inbound Collaboration code.</p>	<p>Exit, Resend, or Reset.</p> <p>The configured default is Reset.</p>
Action on Max No Response	<p>Specifies the action the eWay takes when it attempts to send a message to the external system the maximum allowed number of times, and does not receive any response (HL7 Application Acknowledgement) from the external system. The maximum number times the eWay sends a message without receiving a response is determined by the Max No Response parameter (see “Max No Response” on page 68). The options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. <p>This parameter is used for inbound Collaboration code.</p>	<p>Exit or Reset.</p> <p>The default value is Reset.</p>

Table 16 Connectivity Map - HL7 Recourse Action (Continued)

Name	Description	Required Value
Action on Max Failed Read Retry	<p>Specifies the action the eWay takes after it has reached the empty read limit set by the Max Empty Read Retry parameter. This parameter is used by inbound eWays only. The recourse options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. <p>This parameter is used for inbound Collaboration code.</p>	<p>Exit or Reset.</p> <p>The configured default is Reset.</p>
Action on Nak Received	<p>Specifies the action taken by the eWay when it receives an HL7 Application NAK from the external system. The options are:</p> <ul style="list-style-type: none"> ▪ Resend: The eWay attempts to resend the message to the external system. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. ▪ Skip Message: The eWay remains connected, but writes the message to an error queue. <p>Note: Do not set both the “Action On NAK Received” and “Action On Max NAK Received” parameters to “Skip Message.”</p> <p>This parameter is used for inbound Collaboration code.</p>	<p>Resend, Reset, or Skip Message.</p> <p>The configured default is Resend.</p>

Table 16 Connectivity Map - HL7 Recourse Action (Continued)

Name	Description	Required Value
Action on Max Nak Received	<p>Specifies the action the eWay takes when the maximum number of HL7 Application NAKs have been received from the external system, as set by the Max NAK Receive Retry parameter (see “Max NAK Receive Retry” on page 69). The options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. ▪ Skip Message: The eWay remains connected, but writes the message to an error queue. This parameter is used for inbound Collaboration code. <p>Note: Do not set both “Action On NAK Received” and “Action On Max NAK Received” parameters to “Skip Message.”</p>	<p>Exit, Reset, or Skip Message.</p> <p>The configured default is Skip Message.</p>
Action on Max Nak Sent	<p>Specifies the action taken by the eWay when it has sent the maximum allowed number of NAKs to the external system, as set by the Max NAK Send Retry parameter (see “Max NAK Send Retry” on page 69). The options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. <p>This parameter is used for inbound Collaboration code.</p>	<p>Exit or Reset.</p> <p>The default value is Exit.</p>

4.4 TCP/IP HL7 Outbound eWay Connectivity Map Properties

The TCP/IP HL7 Server (outbound) eWay configuration parameters, accessed from the Connectivity Map, are organized into the following sections:

- [General Outbound Settings](#) on page 74
- [TCPIP Outbound Settings](#) on page 75
- [TCPIP Outbound Settings - Client Connection Establishment](#) on page 80
- [TCPIP Outbound Settings - Server Port Binding](#) on page 82
- [HL7 Acknowledgment](#) on page 83
- [Lower Layer Protocol](#) on page 85
- [Sequence Number Protocol](#) on page 86
- [HL7 MSH Segment](#) on page 87
- [HL7 SFT Segment](#) on page 90
- [Communication Control](#) on page 92
- [HL7 Recourse Action](#) on page 94

4.4.1 General Outbound Settings

Provides the general HL7 outbound configuration settings. This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 16.

Table 17 Connectivity Map - General Outbound Settings

Name	Description	Required Value
Max Data Size	Specifies the maximum size of data that the programs can hold internally. The valid range is from 1 to 2GB (which is the max value of java integer).	An integer indication the maximum data size in bytes, ranging from 1 to 2147483647 (2GB). The configured default is 2147483647 .
Scope Of State	<p>Specifies It is used to define the scope of State object, which is an OTD node. The valid options for this parameter are:</p> <ul style="list-style-type: none"> ▪ Resource Adapter Level: The State has the same life cycle as the resource adapter. ▪ Connection Level: The State has the same life cycle as the connection. ▪ OTD Level: The State has the same life cycle as the OTD object. <p>This scope represents the life cycle of the State.</p>	Select Resource Adapter Level , Connection Level , or OTD Level . The configured default is Resource Adapter Level .

4.4.2 TCPIP Outbound Settings

Presents the java Socket options. For more information see the JDK Javadoc. This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 18.

Table 18 Connectivity Map - TCPIP Outbound Settings

Name	Description	Required Value
Connection Type	<p>Specifies how the eWay establishes the TCP/IP connection:</p> <ul style="list-style-type: none"> ▪ Client: The eWay connects to an external server (host/port) to establish the connection. The eWay is in active mode. ▪ Server: The eWay waits/listens on a particular port for an incoming connection request from an external client. Once the request is received, the eWay accepts the request and establishes the connection. The eWay is in passive mode. 	<p>Select Client or Server. Unless you specifically require Server mode, leave this value as the default, Client.</p>
ServerSoTimeout	<p>Sets or gets the value of the SoTimeout for the ServerSocket, in milliseconds. Used for ServerSocket.accept().</p> <p>When this option is set to a non-zero timeout, calling accept() for ServerSocket will block for only this period of time. If the timeout expires, a java.net.SocketTimeoutException (or java.net.InterruptedIOException) is thrown, though the ServerSocket remains valid.</p> <p>Enable this option prior to entering the blocking operation. This parameter is only used when the Connection Type is set as Server.</p>	<p>The SoTimeout value in milliseconds. The timeout must be greater than 0 (zero). A timeout of zero is interpreted as an infinite timeout.</p> <p>The configured default is 60000 (60 seconds).</p>

Table 18 Connectivity Map - TCPIP Outbound Settings (Continued)

Name	Description	Required Value
Keep Alive	<p>Specifies whether the client's SO_KEEPALIVE option is enabled or disabled. When the option is set for a TCP socket and no data has been exchanged across the socket in either direction for 2 hours, TCP automatically sends a KEEPALIVE probe to the peer (the actual value is implementation dependent). This probe is a TCP segment to which the peer must respond. One of three responses is expected:</p> <ol style="list-style-type: none"> 1 The peer responds with the expected ACK. The application is not notified (since everything is OK). TCP will send another probe following another 2 hours of inactivity. 2 The peer responds with an RST, which tells the local TCP that the peer host has crashed and rebooted. The socket is closed. 3 There is no response from the peer. The socket is closed. The purpose of this option is to detect if the peer host has crashed. This is used for the accepted client Socket. 	<p>True or False. True indicates that the SO_KEEPALIVE option is enabled.</p>
Receive Buffer Size	<p>Sets or gets the value of the SO_RCVBUF option for the current socket, that is the buffer size used by the operating system for input on this socket. It provides an estimate of the size of the underlying buffers used by the platform for incoming network I/O.</p> <p>When used in set, this is a suggestion for the kernel from the application regarding the size of buffers to use for the data to be received over the socket. When used in get, this must return the actual size of the buffer used by the platform when receiving data on this socket.</p>	<p>A number indicating the receive buffer size.</p> <p>The configured default is 8192.</p>

Table 18 Connectivity Map - TCPIP Outbound Settings (Continued)

Name	Description	Required Value
Send Buffer Size	<p>Sets or gets the value of the SO_SNDBUF option for the current socket, that is the buffer size used by the operating system for output on this socket. It provides an estimate of the size of the underlying buffers used by the platform for outgoing network I/O.</p> <p>When used in set, this is a suggestion for the kernel from the application regarding the size of buffers to use for the data to be sent over the socket. When used in get, this must return the actual size of the buffer used by the platform when sending out data on this socket.</p>	<p>A number indicating the send buffer size.</p> <p>The configured default is 8192.</p>
SoLinger	<p>Specifies whether the eWay performs a linger-on-close timeout. This option disables/enables immediate return from a close() of a TCP Socket. This parameter is used in conjunction with SoLinger Timeout.</p> <ul style="list-style-type: none"> ▪ True with SoLinger Timeout set to a non-zero integer timeout: This means that a close() will block, pending the transmission and acknowledgement of all data written to the peer, at which point the socket is closed gracefully. Upon reaching the linger timeout, the socket is closed forcefully with a TCP RST. ▪ True with SoLinger Timeout set to a timeout of zero: Indicates that a forceful close is done immediately. See “SoLinger Timeout” on page 78. 	<p>True or False. True enables the SO_Linger option.</p>

Table 18 Connectivity Map - TCPIP Outbound Settings (Continued)

Name	Description	Required Value
SoLinger Timeout	<p>Specifies the server's SoLinger time-out in seconds. SoLinger Timeout is used in conjunction with SoLinger (see "SoLinger" on page 77) to configure the "linger-on-close" timeout.</p> <p>When SoLinger is set to true (enabled), the SoLinger Timeout value indicates the following:</p> <ul style="list-style-type: none"> ▪ A non-zero integer means that calling close() will block, pending the transmission and acknowledgement of all data written to the peer, at which point the socket is closed gracefully. Upon reaching the linger timeout, the socket is closed forcefully with a TCP RST. If the specified timeout value exceeds 65,535 it will be reduced to 65,535. ▪ A zero integer indicates that a forceful close is done immediately. 	<p>An integer between -1 and 65535.</p> <p>A value of -1 seconds indicates that the SoLinger option is disabled (set as false).</p> <p>A value of 0 (zero) indicates that SoLinger immediately performs a forceful close.</p> <p>An value of 1 to 65535 indicates the number of seconds for the time-out.</p> <p>The configured default is -1 seconds.</p>
SoTimeout	<p>Sets or gets the value of the SoTimeout in milliseconds. When this option is set to a non-zero timeout, calling read() on the InputStream associated with this socket will block for only this configured length of time.</p> <p>If the timeout expires, a java.io.InterruptedIOException (or java.net.SocketTimeoutException) is thrown, but the Socket remains valid. Enable this option prior to entering the blocking operation.</p>	<p>The SoTimeout value in milliseconds. The timeout must be greater than zero (0). A timeout of zero is interpreted as an infinite timeout.</p> <p>The configured default is 10000 (10 seconds).</p>

Table 18 Connectivity Map - TCPIP Outbound Settings (Continued)

Name	Description	Required Value
TcpNoDelay	<p>Specifies whether the server's TcpNoDelay option (that is, Nagle's algorithm) is enabled or disabled.</p> <ul style="list-style-type: none"> ▪ True: Indicates that the server allows data packets that are less than the maximum transfer unit (MTU) size to be sent out immediately over the network. A setting of True may improve performance for higher-speed networks. ▪ False: Indicates that the server does not allow data packets that are less than the MTU size be sent out immediately over the network. <p>This is used for the accepted client socket.</p>	<p>Select True or False.</p> <p>The configured default is False.</p>
Socket Factory Implementation Class Name	<p>Specifies the name of the Java class that implements the socket factory. This class is used to create the socket. If you have provided your own socket implementation, enter the name of the Java class that contains this implementation here. The factory implementation class must implement the following interface:</p> <p><code>com.stc.connector.tcpip.model.factory.TCPIPSocketFactory</code></p>	<p>A Java class name.</p> <p>The default value is com.stc.connector.tcpip.model.factory.TCPIPSocketFactoryImpl</p>

4.4.3 TCPIP Outbound Settings - Client Connection Establishment

The **TCPIP OUTbound Settings - Client Connection Establishment** section defines configuration parameters used for controlling the connection establishment. These properties are only used when the **Connection Type** is set to **Client**.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 19.

Table 19 Connectivity Map - TCPIP Outbound Settings - Client Connection Establishment

Name	Description	Required Value
Time To Wait Before Attempting Connection	Specifies the length of time (in milliseconds) the eWay waits before attempting to connect to the external system.	An integer indicating the length of time (in milliseconds) that the eWay waits before attempts to connect. The configured default is 0 .
Always Create New Connection	Specifies whether the eWay always attempts to create a new connection when a connection establishment request is received. <ul style="list-style-type: none"> ▪ True indicates that the eWay always attempts to create a new connection without attempting to match an existing connection. ▪ False indicates that the eWay attempts to match an existing connection (managed by the container). 	Select True or False . The configured default is False .
Auto Reconnect Upon Matching Failure	Specifies whether to attempt to re-connect automatically when the eWay gets a matching connection from a container, even though this connection is not valid due to various reasons: for example, the external side of the connection is closed/reset due to the external application's logic. <ul style="list-style-type: none"> ▪ True indicates that the eWay discards the invalid matching connection and automatically attempts to reconnect using a new connection. ▪ False indicates that the eWay does not automatically attempt to reconnect using a new connection: instead, the eWay defers the reconnect control to the user Business Rules. The user must detect this type of failure and act appropriately. 	Select True or False . The configured default is True .

Table 19 Connectivity Map - TCPIP Outbound Settings - Client Connection Establishment

Name	Description	Required Value
Max Connection Retry	Specifies the maximum number of times the eWay attempts to connect to a specific external TCP/IP destination (host/port) before giving up.	An integer indicating the number of times the eWay will attempt to connect. The configured default is 3 .
Retry Connection Interval	Specifies the length of time (in milliseconds) the eWay waits between attempts to connect to a specific external TCP/IP destination (host/port).	An integer indicating the length of time (in milliseconds) that the eWay waits between attempts to connect. The configured default is 30000 (or 30 seconds).

4.4.4 TCPIP Outbound Settings - Server Port Binding

Specifies configuration parameters used for controlling server port binding. These properties are only used when the **Connection Type** is set to **Server**.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 20.

Table 20 Connectivity Map - TCPIP Outbound Settings - Server Port Binding

Name	Description	Required Value
Max Binding Retry	Specifies the maximum number of times the eWay will attempt to bind to the specified TCP/IP port on the localhost before giving up.	An integer indicating the number of times the eWay will attempt to bind to the specified TCP/IP port on the localhost. The configured default is 3 .
Retry Binding Interval	Specifies the amount of time (in milliseconds) that the eWay waits between attempts to bind to the specified TCP/IP port on the localhost.	An integer indicating the length of times, in milliseconds, between attempts to bind to the specified TCP/IP port. The configured default is 30000 (30 seconds).

4.4.5 HL7 Acknowledgment

Provides HL7 acknowledgment configuration settings that control how the application acknowledgment Events are handled.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 21.

Table 21 Connectivity Map - HL7 Acknowledgement

Name	Description	Required Value
Acknowledgment Level	<p>Specifies whether the external application is configured to send an HL7 application acknowledgement after successfully receiving the message or after the message has been successfully committed to the application database. The valid levels are:</p> <ul style="list-style-type: none"> ▪ A: Application acknowledgment. The acknowledgement is sent when the message is successfully received. ▪ C: Commit (accept) acknowledgment. The acknowledgement is sent after the message is successfully and functionally processed by one receiving system. 	<p>A or C.</p> <p>The configured default is A.</p>
eGate Sends App Acks	<p>Used by both the inbound and outbound Collaboration.</p> <ul style="list-style-type: none"> ▪ Inbound: Specifies whether the HL7 application acknowledgment sent to the external system is generated by the eWay or forwarded from eGate. <ul style="list-style-type: none"> ▪ True indicates that the eWay receives the external receiving HL7 application acknowledgment from eGate and sends it to the external system. ▪ False indicates that the eWay creates and sends the HL7 application acknowledgment directly to the external system. ▪ Outbound: Specifies whether the outbound Collaboration is in outbound Delayed ACK role; that is, the outbound eWay is connecting to an external system that communicates as a Delayed ACK receiver and is sending two ACKs to the eWay. <ul style="list-style-type: none"> ▪ True indicates that the eWay is expecting a Delayed ACK (2 ACKS). ▪ False indicates that the eWay does not expect a Delayed ACK. 	<p>True or False.</p> <p>The configured default is False.</p>

Table 21 Connectivity Map - HL7 Acknowledgement (Continued)

Name	Description	Required Value
Forward External Acks to eGate	<p>Specifies whether the HL7 application acknowledgment is forwarded to eGate. When an HL7 application acknowledgment is received, it is sometimes necessary to forward the contents of the HL7 application acknowledgment to eGate (as data).</p> <ul style="list-style-type: none"> ▪ True indicates that eWay forwards HL7 application acknowledgments from the external system to eGate for processing. ▪ False indicates that HL7 application acknowledgments from the external system are not forwarded to eGate by the eWay. <p>This parameter is used for the outbound Collaboration code.</p>	<p>True or False.</p> <p>The configured default is False.</p>
Timeout For Delayed Ack	<p>Specifies the timeout value for delayed ACK in milliseconds.</p> <p>This parameter is used for outbound Collaboration code.</p>	<p>An integer indicating the timeout in milliseconds.</p> <p>The configured default is 30000</p>

4.4.6 Lower Layer Protocol

Provides Lower Layer Protocol (LLP) configuration settings.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 22.

Table 22 Connectivity Map - Lower Layer Protocol

Name	Description	Required Value
LLP Type	Specifies the LLP (Lower Layer Protocol) type. The valid types are: <ul style="list-style-type: none"> ▪ MLLP (Minimal Lower Layer Protocol) ▪ HLLP (Hybrid Lower Layer Protocol) For more information on the available envelope types see	MLLP or HLLP . MLLP is the configured default value.
Start Block Character	Specifies the Start Block Character (the first envelope marker character in the HL7 envelope) as a decimal ascii number.	A decimal within the range of 1 to 127 . Unless there is a conflict, the value should be ASCII VT (decimal 11). The default value is 11 .
End Data Character	Specifies the End Data Character (the second to the last envelope marker character in the HL7 envelope) as a decimal ascii number. The allowed range is 1 to 127 .	A decimal within the range of 1 to 127 . Unless there is a conflict, the value should be ASCII FS (decimal 28). The default value is 28 .
End Block Character	Specifies the End Block Character (the last envelope marker character in the HL7 envelope) as a decimal ascii number.	A decimal within the range of 1 to 127 . To be strictly compliant with the HL7 Standard, this parameter MUST be set to a Carriage Return (decimal 13). The default value is 13 .
HLLP Checksum Enabled	Specifies whether the HLLP (Hybrid Lower Level Protocol) checksum is enabled or disabled.	True or False . True indicates that the HLLP Checksum is enabled. The configured default is True .

4.4.7. Sequence Number Protocol

Provides sequence number protocol configuration settings.

***Note:** Many of the parameters for the eWay are specific to the direction the data is travelling, that is whether the eWay is Inbound or Outbound to eGate.*

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 23.

Table 23 Connectivity Map - Sequence Number Protocol

Name	Description	Required Value
Sequence Number Enabled	Specifies whether Sequence Number Protocol is enabled or disabled. HL7 sequence numbering is used to help prevent duplication of data. True indicates that sequence numbering is enabled.	True or False . The configured default is True .

4.4.8 HL7 MSH Segment

Provides HL7 MSH Header segment configuration settings.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 24.

Table 24 Connectivity Map - HL7 Acknowledgement

Name	Description	Required Value
Field Separator	<p>Specifies the separator between the segment ID and the first real field. This value defines the character that is used as a separator for the rest of the message.</p> <p>This is the first field in the HL7 MSH segment (MSH-01).</p>	<p>Field Separator character value as a decimal ascii number. The allowed range is 1 to 127.</p> <p>The default setting is 124 which is character ' '.</p>
Encoding Characters	<p>Specifies four encoding characters in the following order:</p> <ul style="list-style-type: none"> ▪ Component separator ▪ Repetition separator ▪ Escape character ▪ Subcomponent separator <p>This is the second field in the HL7 MSH segment (MSH-02).</p>	<p>HL7 encoding characters in the respective order.</p> <p>The configured default is: ^~\& (ASCII 94, 126, 92, and 38, respectively).</p>
Sending Application	<p>Specifies the sending application among other applications within the network enterprise. The network enterprise consists of the applications that participate in the exchange of HL7 messages within the enterprise.</p> <p>This is the third field in the HL7 MSH segment (MSH-03).</p>	<p>User defined value for the HL7 sending application.</p> <p>The configured default is Sun SeeBeyond HL7 eWay.</p>
Sending Facility	<p>Specifies (further identifies) the sending application among multiple identical instances of the application running on behalf of different organizations.</p> <p>This is the fourth field in the HL7 MSH segment (MSH-04).</p>	<p>User defined value for the HL7 sending facility.</p> <p>The configured default is Sun SeeBeyond HL7 eWay.</p>

Table 24 Connectivity Map - HL7 Acknowledgement (Continued)

Name	Description	Required Value
Receiving Application	Specifies the receiving application among other applications within the network enterprise. The network enterprise consists of the applications that participate in the exchange of HL7 messages within the enterprise. This is the fifth field in the HL7 MSH segment (MSH-05).	User defined value for the HL7 receiving application. The configured default is Sun SeeBeyond HL7 eWay .
Receiving Facility	Specifies (further identifies) the receiving application among multiple identical instances of the application running on behalf of different organizations. This is the sixth field in the HL7 MSH segment (MSH-06).	User defined value for the HL7 receiving facility. The configured default is Sun SeeBeyond HL7 eWay .
Security	Specifies the implemented application level security features. This is the eighth field in the HL7 MSH segment (MSH-08).	Under development by HL7.
Processing ID	Specifies the sub-component Processing ID of MSH-11. MSH-11 is used to indicate whether a message is processed as defined in the HL7 Application (level 7) Processing rules.	Enter one of the following: <ul style="list-style-type: none"> ▪ D (for Debugging) ▪ P (for Production) ▪ T (for Training) In some cases there may be an additional sub-component "Processing Mode" following the initial value. P is the configured default.
Version ID	Specifies the particular HL7 version to be matched by the receiving system to its own version. This is the 12th field in the HL7 MSH segment (MSH-12).	The HL7 Standard version value as displayed in HL7 Table 0104 - Version ID. The default value is 2.5 .
Country Code	Specifies a code that indicates the country of origin for the message (see HL7 Table 0399). Used to specify default elements in a message, such as currency. This is the 17th field in the HL7 MSH segment (MSH-17).	The Country Code value uses the 3-character (alphabetic) form of ISO 3166. The default value is USA .

Table 24 Connectivity Map - HL7 Acknowledgement (Continued)

Name	Description	Required Value
Character Set	Specifies the character set(s) in use by the messages (see HL7 Table 0211). If the field is left blank, the character set in use is understood to be the 7-bit ASCII set. This is the 18th field in the HL7 MSH segment (MSH-18).	The configured default is 8859/1 (printable 7-bit ASCII character set). See HL7 Table 0211 for available values and descriptions.
Principal Language of Message	Specifies the principal language of the message. Codes come from ISO 639. This is the 19th field in the HL7 MSH segment (MSH-19).	The 2-character ISO 639 alphabetic code.
Alternate Character Set Handling Scheme	Specifies the value for the Alternate character set handling scheme to be used when any alternative character sets are used and a special handling scheme is necessary (see HL7 Table 0356). This is the 20th field in the HL7 MSH segment (MSH-20).	Available values include ISO 2022-1994, 2.3 , or <null> (blank). Leaving the field blank indicates that no character set switching will occur.
Conformance Statement ID	The Conformance Statement ID (Message Profile Identifier in V2.5) is a unique identifier that applies to a query's Conformance Statement, or as a Message Profile Identifier, asserts constancy with a message profile (grammar, syntax, usage, and so forth). This is the 21st field in the HL7 MSH segment (MSH-21).	An HL7 Conformance Statement ID value or leave blank.
Validate MSH	Specifies whether to validate the MSH segment of the data message (for inbound) and the MSH segment of the ACK (for outbound). This parameter is used for outbound Collaboration code. <i>Note: This property does not affect structural validation of the whole HL7 message itself. Structural validation is always performed.</i>	True or False . True indicates that the Collaboration validates the MSH segment. The configured default is True .

4.4.9 HL7 SFT Segment

Provides HL7 SFT Segment configuration settings. The SFT segment is available starting with HL7 version 2.5. This segment provides additional information about one or more software product used as sending applications. The primary purpose of this segment is for diagnostic use. There may be additional uses per site-specific agreements.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 25.

Table 25 Connectivity Map - HL7 SFT Segment

Name	Description	Required Value
Enable	Specifies whether the optional SFT segment is enabled in the ACK message. Note: If enable is set to true, and the HL7 version is not configured as 2.5, the eWay will error upon startup.	True or False. True indicates that the SFT segment is enabled. The configured default is True .
Software Vendor Organization	Specifies HL7 segment SFT-01 , the name of the company that publishes and/or distributes the sending software that created the transaction. The purpose of this field, along with the remaining fields in this segment, is to provide a more complete profile of the sending applications. The Software Vendor Organization field identifies the vendor who is responsible for maintaining the application.	The name of the sending software publisher or vendor. The configured default is Sun Microsystems, Inc. Note: This property only applies to HL7 version 2.5
Software Certified Version or Release Number	Specifies HL7 segment SFT-02 , the Software Certified Version or Release Number : the latest software version number or release number for the sending system. This helps to provide a more complete profile of the application that is sending or receiving HL7 messages. Version numbers are important in identifying the specific _release_ of an application. In some situations, the receiving application validates the software certified version or release number against a list of “certified” versions or releases of the particular software. This helps determine whether the sending application adheres to specific Business Rules required by the receiving application. Alternatively, the software may perform different processing, depending on the version of the sending software.	The software certified version or release number. The configured default is 5.1.1 . Note: This property only applies to HL7 version 2.5.

Table 25 Connectivity Map - HL7 SFT Segment (Continued)

Name	Description	Required Value
Software Product Name	Specifies HL7 segment SFT-03 , the name of the software product that submitted the transaction. The software product name is a key component for identifying the sending application.	<p>The sending software product name. The default value is HL7 eWay.</p> <p>The configured default is Sun SeeBeyond eWay TCP/IP HL7 Adapter.</p> <p>Note: This property only applies to HL7 version 2.5.</p>
Software Binary ID	<p>Specifies HL7 segment SFT-04, the Software Binary ID. This property is available starting with HL7 version 2.5. Software Binary IDs are issued by a vendor for each unique software version instance.</p> <p>These IDs are used to differentiate between differing versions of the same software. Software IDs are issued for each unique software version instance. Identical Primary IDs indicate that the software is identical at the binary level, but configuration settings may differ.</p>	<p>The unique Software Binary ID.</p> <p>The configured default is 5.1.1.</p> <p>Note: This property only applies to HL7 version 2.5.</p>
Software Product Information	<p>Specifies HL7 segment SFT-05, software product identification information. This may include a description of the software application, configuration settings, modifications made to the software, and so forth. This field can contain any additional information about the sending application, with the transaction it has submitted.</p> <p>This information is used for diagnostic purposes and may provide greater flexibility for identifying the application software.</p>	<p>Information that may help to identify the specific sending software. This field should only be used when performing diagnostics.</p> <p>The default value is "It is a JCA adapter for HL7 over TCP/IP connection."</p> <p>Note: This property only applies to HL7 version 2.5.</p>
Software Install Date	Specifies HL7 segment SFT-06 , the Software Install Date. This is the date on which the submitting software was installed at the sending site. the software install date on its own can often provide key information about the behavior of the application. This is necessary for providing a more complete profile of the sending application.	<p>The date of installation for the sending application software.</p> <p>Note: This property only applies to HL7 version 2.5.</p>

4.4.10 Communication Control

Controls data transferring (sending/receiving) over TCP/IP connection.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 26.

Table 26 Connectivity Map - Communication Control

Name	Description	Required Value
Time To Wait For A Response	<p>Specifies the amount of time (in milliseconds) that the eWay waits for a response from the external system before taking recourse action (see “Action on No Response” on page 94). Any data from the external system is considered a response.</p> <p>This property corresponds to the initial read/receive operation timeout. Once a response is received, the subsequent read/receive operation uses the SoTimeout specified timeout (see “SoTimeout” on page 78).</p>	<p>An integer indicating the length of time in milliseconds that the eWay waits for a response from the external system. A value of 0 (zero) is interpreted as an infinite timeout.</p> <p>The configured default is 30000 (30 seconds).</p>
Max Empty Read Retry	<p>Specifies the maximum number of times the eWay attempts to read data from the external system after the read/receive operation returns nothing. This applies to the read or receive operation after a response starts to arrive.</p> <p>Empty Read means that a timeout occurs on the read/receive operation, which takes the SoTimeout parameter in the TCPIP Server Base Settings section as the applied timeout setting (see “SoTimeout” on page 78).</p> <p>The corresponding recourse action is specified by the “Action on Max Failed Read Retry” on page 95.</p>	<p>An integer indicating the maximum number or retries.</p> <p>The configured default is 5.</p>
Max No Response	<p>Specifies the maximum number of response timeouts the eWay allows, while waiting for data from the external system, before taking recourse action.</p> <p>This parameter is only used by outbound eWays (outbound Collaboration code) and works in conjunction with the Resend option of the Recourse Action parameter “Action on Max No Response” on page 94. It configures the eWay to resend the last message for the specified maximum number of times before the subsequent recourse action is taken.</p>	<p>An integer indicating the appropriate number of timeouts that may occur before taking recourse action.</p> <p>The default value is 5.</p>

Table 26 Connectivity Map - Communication Control (Continued)

Name	Description	Required Value
Max NAK Receive Retry	Specifies the maximum number of negative acknowledgments the eWay receives before taking recourse action (see “Action on Max Nak Received” on page 96). This parameter is used for outbound Collaboration code.	An integer indicating the appropriate maximum number of NAKs received before taking recourse action. The default value is 5 .
Max NAK Send Retry	Specifies the maximum number of negative acknowledgments the eWay sends before taking recourse action (see “Action on Max Nak Sent” on page 96). This parameter is used for outbound Collaboration code.	An integer that indicates the appropriate maximum number of NAKs sent by the eWay before recourse action is taken. The default value is 5 .
Max Canned NAK Send Retry	Specifies the maximum number of canned negative acknowledgments that the eWay sends before taking recourse action (see “Action on Max Nak Sent” on page 96).	The appropriate maximum number of canned NAK to send before taking recourse action. 0 indicates that the eWay will not attempt to create or send a canned NAK. The default value is 3 .
Enable Journaling	Specifies whether message journaling is enabled. This parameter is used for outbound Collaboration code.	True or False. True indicates that journaling is enabled. The default value is False .

4.4.11 HL7 Recourse Action

Determines the actions the eWay takes when operations occur outside the configured constraints.

This section of the TCP/IP HL7 inbound eWay Connectivity Map properties contains the top-level parameters displayed in Table 27.

Table 27 Connectivity Map - HL7 Recourse Action

Name	Description	Required Value
Action on No Response	<p>Specifies the action taken by the eWay when no ACK is received from the external system in the allotted time. The amount of time is determined by the Time To Wait For A Response parameter (see “Time To Wait For A Response” on page 92). The options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Resend: The eWay attempts to resend the message to the external system. The Resend option is only allowed when sequence numbering is in effect. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. <p>This parameter is used for outbound Collaboration code.</p>	<p>Exit, Resend, or Reset.</p> <p>The configured default is Reset.</p>
Action on Max No Response	<p>Specifies the action the eWay takes when it attempts to send a message to the external system the maximum allowed number of times, and does not receive any response (HL7 Application Acknowledgement) from the external system. The maximum number times the eWay sends a message without receiving a response is determined by the Max No Response parameter (see “Max No Response” on page 92). The options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. <p>This parameter is used for outbound Collaboration code.</p>	<p>Select Exit or Reset.</p> <p>The default value is Reset.</p>

Table 27 Connectivity Map - HL7 Recourse Action (Continued)

Name	Description	Required Value
Action on Max Failed Read Retry	<p>Specifies the action the eWay takes after it has reached the empty read limit set by the Max Empty Read Retry parameter. The recourse options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. <p>This parameter is used for outbound Collaboration code.</p>	<p>Select Exit or Reset.</p> <p>The configured default is Reset.</p>
Action on Nak Received	<p>Specifies the action taken by the eWay when it receives an HL7 Application NAK from the external system. The options are:</p> <ul style="list-style-type: none"> ▪ Resend: The eWay attempts to resend the message to the external system. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. ▪ Skip Message: The eWay remains connected, but writes the message to an error queue. <p>Note: Do not set both the "Action On NAK Received" and "Action On Max NAK Received" parameters to "Skip Message."</p> <p>This parameter is used for outbound Collaboration code.</p>	<p>Select Resend, Reset, or Skip Message.</p> <p>The configured default is Resend.</p>

Table 27 Connectivity Map - HL7 Recourse Action (Continued)

Name	Description	Required Value
Action on Max Nak Received	<p>Specifies the action the eWay takes when the maximum number of HL7 Application NAKs have been received from the external system, as set by the Max NAK Receive Retry parameter (see “Max NAK Receive Retry” on page 93). The options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. ▪ Skip Message: The eWay remains connected, but writes the message to an error queue. This parameter is used for outbound Collaboration code. <p>Note: Do not set both “Action On NAK Received” and “Action On Max NAK Received” parameters to “Skip Message.”</p>	<p>Select Exit, Reset, or Skip Message.</p> <p>The configured default is Skip Message.</p>
Action on Max Nak Sent	<p>Specifies the action taken by the eWay when it has sent the maximum allowed number of NAKs to the external system, as set by the Max NAK Send Retry parameter (see “Max NAK Receive Retry” on page 93). The options are:</p> <ul style="list-style-type: none"> ▪ Exit: The eWay terminates its connection with the external system and shuts down. ▪ Reset: The eWay closes its connection with the external system and goes through the connection scenario. <p>This parameter is used for outbound Collaboration code.</p>	<p>Select Exit or Reset.</p> <p>The default value is Exit.</p>

4.5 TCP/IP HL7 eWay Environment Explorer Properties

The TCP/IP HL7 eWay configuration parameters accessed from the Environment Explorer tree apply to both the inbound and outbound eWays. The TCP/IP eWay's Environment properties are organized into the following sections:

- [HL7 Inbound eWay - TCPIP Inbound Settings](#) on page 97
- [HL7 Inbound eWay - MDB Pool Settings](#) on page 98
- [HL7 Inbound eWay - Sequence Number Protocol](#) on page 100
- [HL7 Outbound eWay - TCPIP Outbound Settings](#) on page 101
- [HL7 Outbound eWay - Connection Pool Settings](#) on page 102
- [HL7 Outbound eWay - Sequence Number Protocol](#) on page 104

4.5.1 HL7 Inbound eWay - TCPIP Inbound Settings

The **HL7 Inbound eWay - TCPIP Inbound Settings** present the java Socket and ServerSocket options. For more information, please refer JDK javadoc.

This section of the TCP/IP HL7 eWay Environment properties contains the top-level parameters displayed in Table 28.

Table 28 Environment - HL7 Inbound eWay - TCPIP Inbound Settings

Name	Description	Required Value
Host	Specifies the host name or IP address used to establish a TCP/IP connection. This parameter is only used when the Connection Type is set to Client .	The host name or IP address to used to establish a TCPIP connection.
ServerPort	<p>Specifies the port number of the TCP/IP destination. This is dependent on the value set for Connection Type.</p> <ul style="list-style-type: none">▪ If Connection Type is set to Server, it indicates the port number on the local host.▪ If Connection Type is set to Client, it indicates the port number of the external host.	<p>An integer between 0 - 65535, indicating the port number of the TCP/IP destination.</p> <p>The configured default is 8888.</p>
Backlog	<p>Specifies the maximum length of the queue when creating the ServerSocket. The maximum queue length for the incoming connection indications (request to connect) is set to the Backlog parameter. When a connection indication arrives and the queue is full, the connection is refused.</p> <p>This parameter is only used when Connection Type is set to Server.</p>	<p>An integer indicating the maximum length of the queue when creating the ServerSocket.</p> <p>The configured default is 50.</p>

4.5.2 HL7 Inbound eWay - MDB Pool Settings

The **HL7 Inbound eWay - MDB Pool Settings** are specific to the MDB bean pool of **Sun Java System Application Server** or **Sun SeeBeyond Runtime Server** only. The parameter settings in this section will go into **sun-ejb-jar.xml**.

This section of the TCP/IP HL7 eWay Environment properties contains the top-level parameters displayed in Table 29.

Table 29 Environment - HL7 Inbound eWay - TCPIP Inbound Settings

Name	Description	Required Value
Steady Pool Size	<p>Specifies the minimum number of MDB beans to be maintained.</p> <p>When the value is set to a number greater than 0 (zero), the container not only pre-populates the MDB bean pool with the specified number, but also attempts to ensure that there is always this many MDB beans in the free pool. This ensures that there are enough MDB beans in the ready-to-serve state to process user requests.</p> <p>This parameter does not necessarily guarantee that no more than steady-pool-size instances exist at a given time. It only governs the number of instances that are pooled over a long period of time. For example, suppose an idle stateless session container has a fully-populated pool with a steady-pool-size of 10. If 20 concurrent requests arrive for the MDB bean component, the container creates 10 additional instances to satisfy the burst of requests. The advantage of this is that it prevents the container from blocking any of the incoming requests. However, if the activity dies down to 10 or fewer concurrent requests, the additional 10 instances are discarded.</p>	<p>An integer indication the minimum number of MDB beans to be maintained.</p> <p>The configured default is 10.</p>
Max Pool Size	<p>Specifies the maximum number of MDB beans in the pool.</p>	<p>An integer indication the maximum number of MDB beans in the pool. A value of 0 (zero) indicates that the pool is unbounded.</p> <p>The configured default is 60.</p>

Table 29 Environment - HL7 Inbound eWay - TCPIP Inbound Settings (Continued)

Name	Description	Required Value
Pool Idle Timeout in Seconds	<p>Specifies the maximum amount of time (in seconds) that an MDB bean instance can remain idle in the pool. When an MDB has exceeded the configured Pool Idle Timeout, a timer thread removes the unused MDB bean. This property defines the interval at which this thread runs.</p> <p>A value greater than 0 indicates that the container removes or destroys any MDB bean instance that is idle at this specified duration. A value of 0 (zero) specifies that idle MDB beans can remain in the pool indefinitely.</p>	<p>The maximum amount of time (in seconds) that an MDB bean instance can remain idle in the pool.</p> <p>The configured default is 600 (10 minutes)</p>

4.5.3 HL7 Inbound eWay - Sequence Number Protocol

The **HL7 Inbound eWay - Sequence Number Protocol** provides configuration settings used to help prevent duplication of data.

This section of the TCP/IP HL7 eWay Environment properties contains the top-level parameters displayed in Table 30.

Table 30 Environment - HL7 Inbound eWay - Sequence Number Protocol

Name	Description	Required Value
Sequence Number File Location	<p>Specifies the location of the sequence number file (a local directory). This is required when the Sequence Number Protocol is enabled. The sequence number file is a non-volatile directory that stores the sequence number files that are used to persist the HL7 sequence number. This unique base file name is automatically generated according to Project/Collaboration information.</p> <p>For the Inbound eWay the file names are created as follows: <Project name> + <deployment name> + <external application node name> + <Collaboration name> + .seqno</p> <p>For example: prjHL7Inbound_dpIn_eaHL7Inbound_jcdHL7inbound1.seqno</p>	<p>The path and directory where the sequence number file is located.</p> <p>The default setting is: /temp/hl7inbound/seq</p>

4.5.4. HL7 Outbound eWay - TCPIP Outbound Settings

Presents the java Socket options. For more information, please refer JDK javadoc.

This section of the TCP/IP HL7 eWay Environment properties contains the top-level parameters displayed in Table 31.

Table 31 Environment - HL7 Outbound eWay - TCPIP Outbound Settings

Name	Description	Required Value
Host	Specifies the host name or IP address used to establish a TCPIP connection. This parameter is only used when the Connection Type is set to Client .	The host name or IP address used to establish a TCPIP connection. The configured default is localhost
ServerPort	Specifies the port number of the TCP/IP destination. This is dependent on the value set for Connection Type . <ul style="list-style-type: none"> ▪ If Connection Type is set to Server, it indicates the port number on the local host. ▪ If Connection Type is set to Client, it indicates the port number of the external host. 	An integer between 0 - 65535 , indicating the port number of the TCP/IP destination. The configured default is 7777 .
Backlog	Specifies the maximum length of the queue when creating the ServerSocket . The maximum queue length for the incoming connection indications (request to connect) is set to the Backlog parameter. When a connection indication arrives and the queue is full, the connection is refused. This parameter is only used when Connection Type is set to Server .	An integer indicating the maximum length of the queue when creating the ServerSocket . The configured default is 50 .

4.5.5 HL7 Outbound eWay - Connection Pool Settings

The **HL7 Outbound eWay - Connection Pool Settings** are specific to the RA connection pool of **Sun Java System Application Server** (Sun One Application Server) or **Sun SeeBeyond Runtime Server** only. The parameter settings in this section will go into **sun-ra.xml**.

This section of the TCP/IP HL7 eWay Environment properties contains the top-level parameters displayed in Table 32.

Table 32 Environment - HL7 Outbound eWay - Connection Pool Settings

Name	Description	Required Value
Steady Pool Size	<p>Specifies the minimum number of RA connections to be maintained.</p> <p>When the value is set to a number greater than 0 (zero), the container not only pre-populates the RA connection pool with the specified number, but also attempts to ensure that there is always this many RA connections in the free pool. This ensures that there are enough RA connections in the ready-to-serve state to process user requests.</p> <p>This parameter does not necessarily guarantee that no more than steady-pool-size instances exist at a given time. It only governs the number of instances that are pooled over a long period of time. For example, suppose an idle stateless session container has a fully-populated pool with a steady-pool-size of 10. If 20 concurrent requests arrive for the RA connection component, the container creates 10 additional instances to satisfy the burst of requests. The advantage of this is that it prevents the container from blocking any of the incoming requests. However, if the activity dies down to 10 or fewer concurrent requests, the additional 10 instances are discarded.</p>	<p>An integer indication the minimum number of RA connections to be maintained.</p> <p>The configured default is 1.</p>
Max Pool Size	<p>Specifies the maximum number of RA connections in the pool.</p>	<p>An integer indication the maximum number of RA connections in the pool. A value of 0 (zero) indicates that the pool is unbounded.</p> <p>The configured default is 32.</p>

Table 32 Environment - HL7 Outbound eWay - Connection Pool Settings (Continued)

Name	Description	Required Value
Pool Idle Timeout in Seconds	<p>Specifies the maximum amount of time (in seconds) that an RA connections instance can remain idle in the pool. When an RA connection has exceeded the configured Pool Idle Timeout, a timer thread removes the unused RA connection. This property defines the interval at which this thread runs.</p> <p>A value greater than 0 indicates that the container removes or destroys any RA connection instance that is idle at this specified duration. A value of 0 (zero) specifies that idle RA connections can remain in the pool indefinitely.</p>	<p>The maximum amount of time (in seconds) that an RA connections instance can remain idle in the pool.</p> <p>The configured default is 300 (5 minutes)</p>

4.5.6 HL7 Outbound eWay - Sequence Number Protocol

The **HL7 Outbound eWay - Sequence Number Protocol** provides configuration settings used to help prevent duplication of data.

This section of the TCP/IP HL7 eWay Environment properties contains the top-level parameters displayed in Table 33.

Table 33 Environment - HL7 Outbound eWay - Sequence Number Protocol

Name	Description	Required Value
Sequence Number File Location	<p>Specifies the location of the sequence number file (a local directory). This is required when the Sequence Number Protocol is enabled. The sequence number file is a non-volatile directory that stores the sequence number files that are used to persist the HL7 sequence number. This unique base file name is automatically generated according to Project/Collaboration information.</p> <p>For the Outbound eWay the file names are created as follows: <Project name> + <deployment name> + <collab name> + <external application node name> + .seqno</p> <p>For example: prjHL7Outbound_dpOut_jcolHL7Outbound_eaHL7Outbound.seqno</p>	<p>The path and directory where the sequence number file is located.</p> <p>The default setting is: /temp/hl7outbound/seq</p>

Working with TCP/IP HL7 Collaborations

This chapter provides an overview and description of the structure and functionality of the Inbound and Outbound Collaborations provided as part of the TCP/IP HL7 eWay. It also provides information on how to incorporate them into a Project.

Chapter Topics

- [Inbound Collaboration Overview](#) on page 106
- [Outbound Collaboration Overview](#) on page 110
- [Creating a Copy of a Project](#) on page 115
- [Customizing Predefined Collaborations](#) on page 115

5.1 The TCP/IP HL7 eWay Collaborations

The TCP/IP HL7 eWay includes one inbound Collaboration (**jcdHL7Inbound**) and one outbound Collaboration (**jcdHL7Outbound**), provided within the sample Projects for inbound and outbound HL7 messaging. These template/sample Projects, **prjHL7Inbound** and **prjHL7Outbound** are downloaded from the Sun Java™ Composite Application Platform Suite Installer. These Collaborations are designed to work “as is” for HL7 compliant interfaces, and can be configured for your specific needs using only the property configuration files. If an interface requires special functionality, the Collaboration’s Java code is easily accessible for modification, much of which can be created graphically (*drag and drop*), using the Collaboration Editor’s Business Rules Designer.

The Collaborations contain a number of OTDs that extend functionality for HL7 message handling, logging, error messaging, journaling, and sequence numbering. These include both generic HL7 OTDs for HL7 ACK/NAK generation or verification, and the Resource Adapter that communicates to the external system and offers services to the integration server. The Collaboration controls messaging protocol and all business logic.

The Collaborations are designed to target “one unit of work” at a time, meaning the resolution of one message at a time. The basic structure of both Collaborations is a state machine implemented in a Java switch statement. The state machine keeps track of the messaging protocol so that when a Collaboration is invoked, it can retrieve the state of the connection just handed to it by the RA, and then execute the proper actions based on the state machine.

At the end of each action, the state is set for the next execution of the Collaboration. There are three main states:

- **To Establish:** A new or reset connection needs to have an HL7 session established. If sequence numbering is used, the sequence numbers need to be negotiated.
- **Messaging:** This is where the exchange of messages and ACKs is takes place.
- **Shutdown:** This is where any cleanup can happen before the connection is closed, or to close the connection.

Additional Collaborations can be added to a Project to increase message flow.

Note: *The TCP/IP HL7 Inbound Collaboration publishes received data as a Byte message in JMS using the `sendBytes()` method. However, the HL7 Outbound Collaboration expects a Text message from JMS. The eWay is not designed for the HL7 Outbound Collaboration to subscribe to a JMS data queue created by the HL7 Inbound Collaboration directly. HL7 Inbound and Outbound Collaborations are designed to communicate through an HL7 eWay TCP/IP connection.*

5.1.1 Inbound Collaboration Overview

The Inbound HL7 Collaboration, **jcdHL7inbound**, contains OTDs for the HL7 Resource Adapter, JMS Data, HL7 ACK, JMS Journal, and JMS Error, as well as the Generic HL7 Event. The Collaboration works with its own internal code and the Properties Configuration files.

Inbound Collaboration - Part 1

The inbound Collaboration is triggered by an HL7 message received from an external system or an outbound HL7 Client. The Collaboration calls the HL7 User Collaboration Rule by executing **receive()**.

The receive method is the entry point to the HL7 User Collaboration, with the following signature: public void

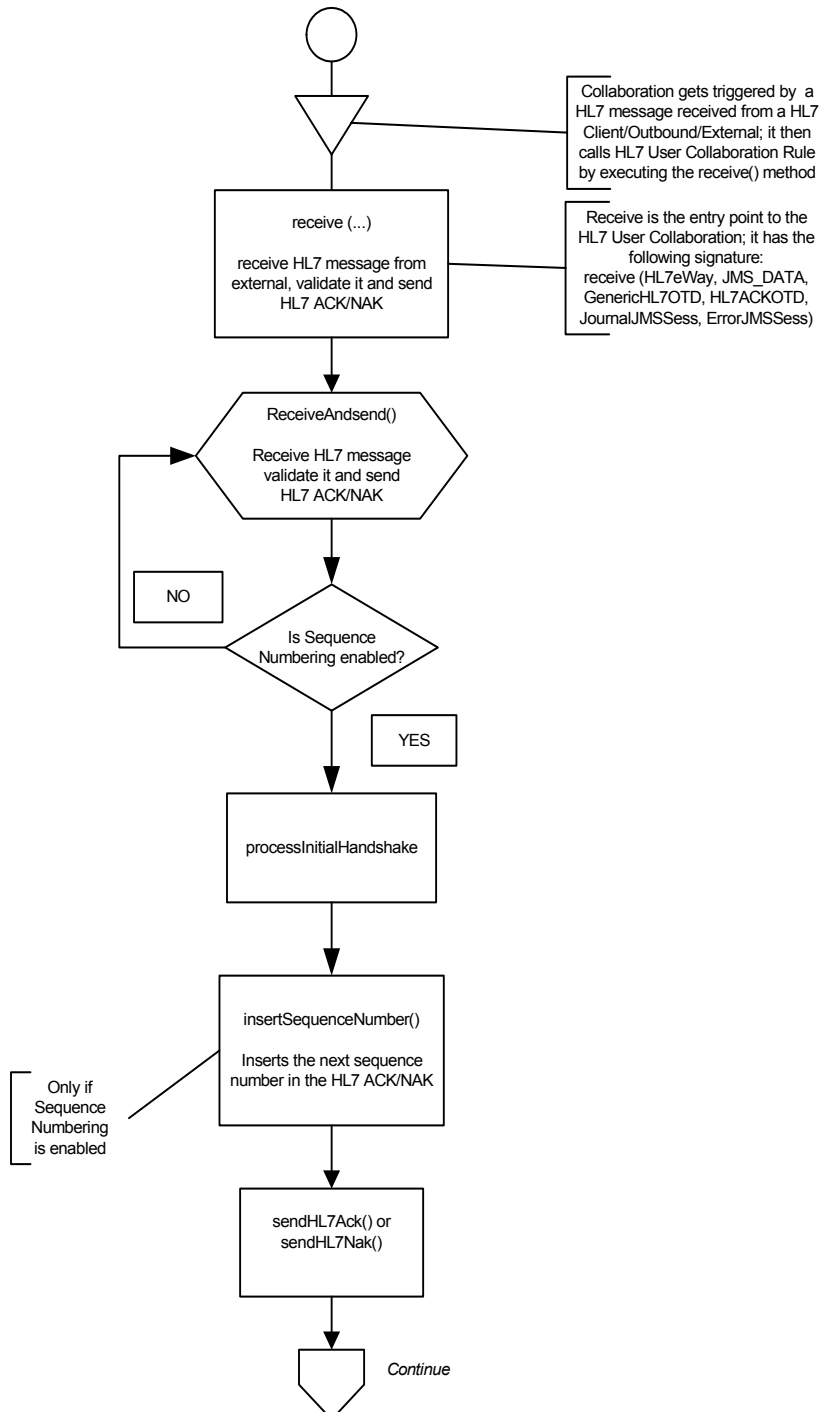
```
receive(com.stc.connector.appconn.tcpip.hl7.HL7ServerApplication input,  
com.stc.connectors.jms.JMS otdJMS_DATA,  
com.stc.SeeBeyond.OTD_Library.HL7.Generic.HL7_GENERIC_EVT.GENERIC_EVT  
otdHL7_GENERIC_EVT_1,  
com.stc.SeeBeyond.OTD_Library.HL7.Generic.HL7_ACK.ACK  
otdHL7_ACK_1,com.stc.connectors.jms.JMS otdJMS_JOURNAL,  
com.stc.connectors.jms.JMS otdJMS_ERROR) throws Throwable.
```

Once the message is received, the Collaboration determines whether the message needs to be validated. The HL7 message is then validated making sure that the message structure is correct. Various fields in the MSH segment of the message are also validated, such as Version ID and Sending Facility. If these fields do not match the configuration, a NAK is returned.

If sequence numbering is enabled the Collaboration checks to see if the messages sequence number is valid. If the sequence number is not valid, the eWay sends a NAK.

The validated HL7 message moves on to **processInitialHandshake()** and the sequence numbers are synchronized. The sequence number within the message is checked against the expected sequence number. If the numbers match, the Collaboration sends an ACK, if not it sends a NAK. The ACK or NAK includes information from various fields of the incoming MSA segment. The ACKs level of acknowledgement is set to **A** (acknowledgement is sent when the message is successfully received), or **C** (acknowledgement is sent after the message is successfully processed). See Figure 17.

Figure 17 The Inbound Collaboration - Part 1



Inbound Collaboration - Part 2

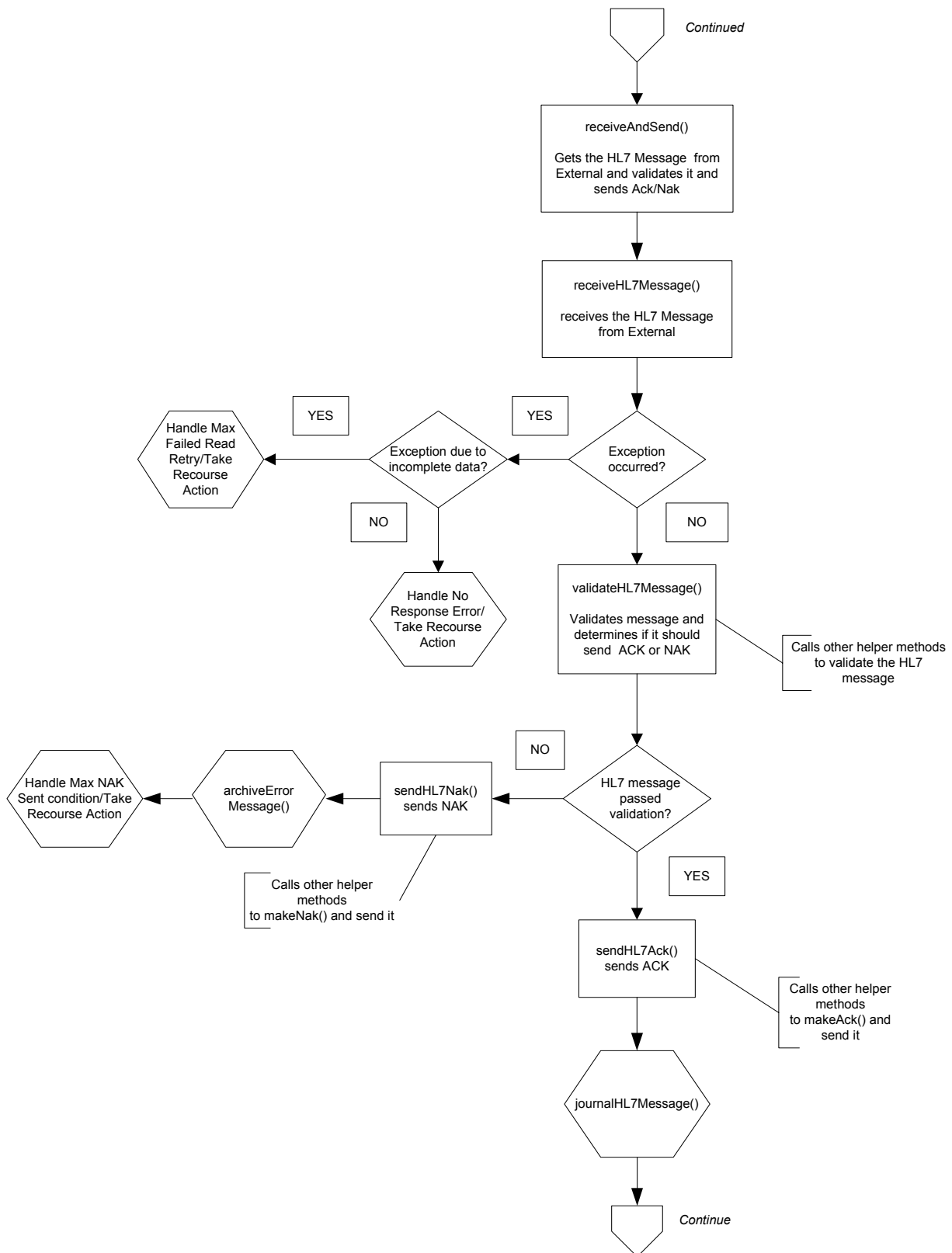
The Collaboration receives the HL7 message from the external using **receiveHL7message()**. If an exception occurs due to incomplete data, and the eWay fails to read the data within the configured number of retries, the associated recourse action is taken (see [Action on Max Failed Read Retry](#) on page 95). If the exception is due to no response, the associated recourse action is taken (see [Action on No Response](#) on page 70).

If no exception occurs, **validateHL7Message()** is called, which validates the message to determine whether to ACK or NAK the message. Other helper methods are also called to validate the HL7 message.

If the HL7 message does not pass validation, the Collaboration calls **makeNak()** and **sendHL7Nak()** to create and send the NAK to the external. The HL7 message, with the NAK, is archived to the Error Queue. If the number of consecutive NAKs sent surpasses the maximum number of retries, the associated recourse action is taken (see [Action on Max Nak Sent](#) on page 72).

If the HL7 message passes validation, the Collaboration calls **makeAck()** and **sendHL7Ack()** to create and send the ACK to the external. See [Figure 18 on page 109](#).

Figure 18 The Inbound Collaboration - Part 2

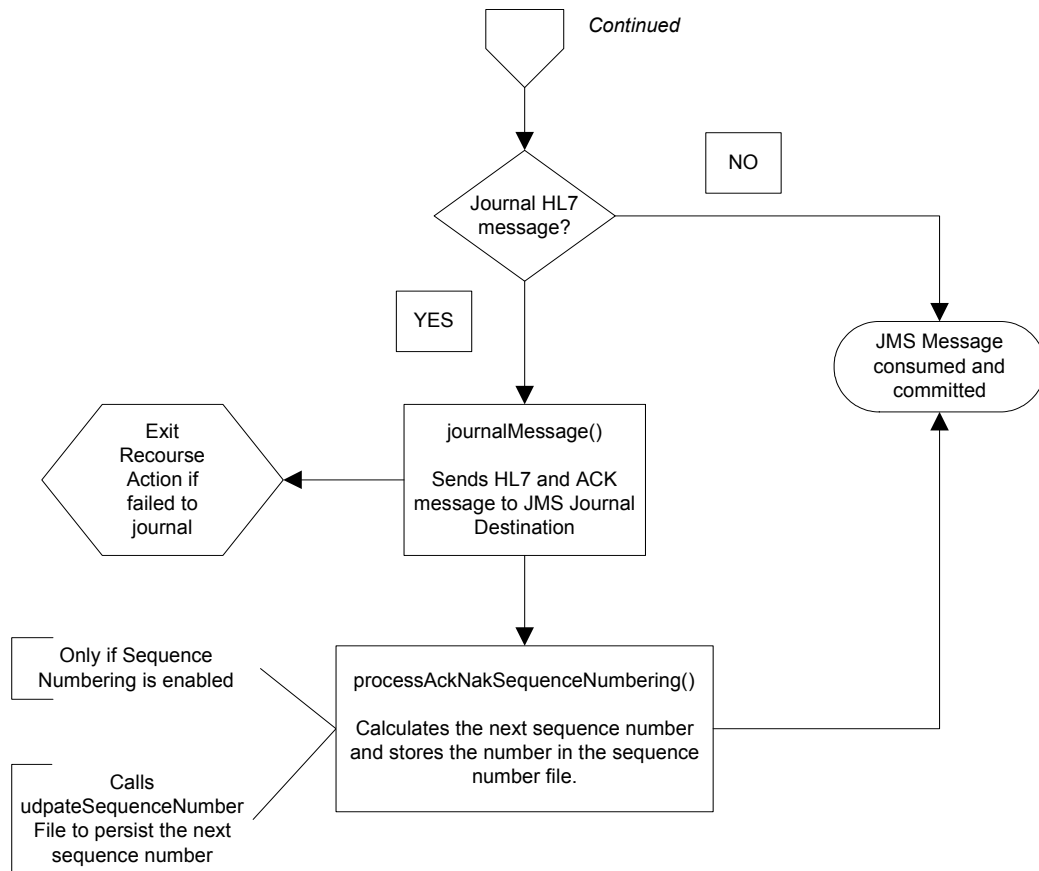


Inbound Collaboration - Part 3

After the ACK is sent, the HL7 message and the ACK are journaled to the JMS Queue Journal destination. If the message fails to journal the associated recourse action is taken.

If Sequence Numbering is enabled, the **processAckNakSequenceNumbering** method calculates the next sequence number and stores the number in the sequence number file by calling the **updateSequenceNumberFile** method to persist the next sequence number (see Figure 19).

Figure 19 The Inbound Collaboration - Part 3



5.1.2 Outbound Collaboration Overview

The Outbound HL7 Collaboration, **jcdHL7Outbound**, contains OTDs for the HL7 Resource Adapter, JMS Data, HL7 ACK, JMS Journal, and JMS Error, as well as the Generic HL7 Event. The Collaboration works with its own internal code and the Properties Configuration files. The outbound Collaboration assumes that it is reading valid HL7 messages, so the data flow that feeds this Collaboration must ensure this.

Outbound Collaboration - Part 1

The Collaboration is triggered by a JMS HL7 message. The Collaboration then calls the HL7 User Collaboration Rule by executing the **receive** method. Receive is the entry

point to the HL7 User Collaboration, with the following signature: receive (input, otdHL7eWay_1, otdJMS_JOURNAL, otdJMS_ERROR, otdHL7_ACK_1, otdHL7_GENERIC_EVT_1).

The incoming HL7 message is then validated, making sure that the message structure is correct. Various fields of the message are also validated, such as Sending Facility, version ID, and MSH.

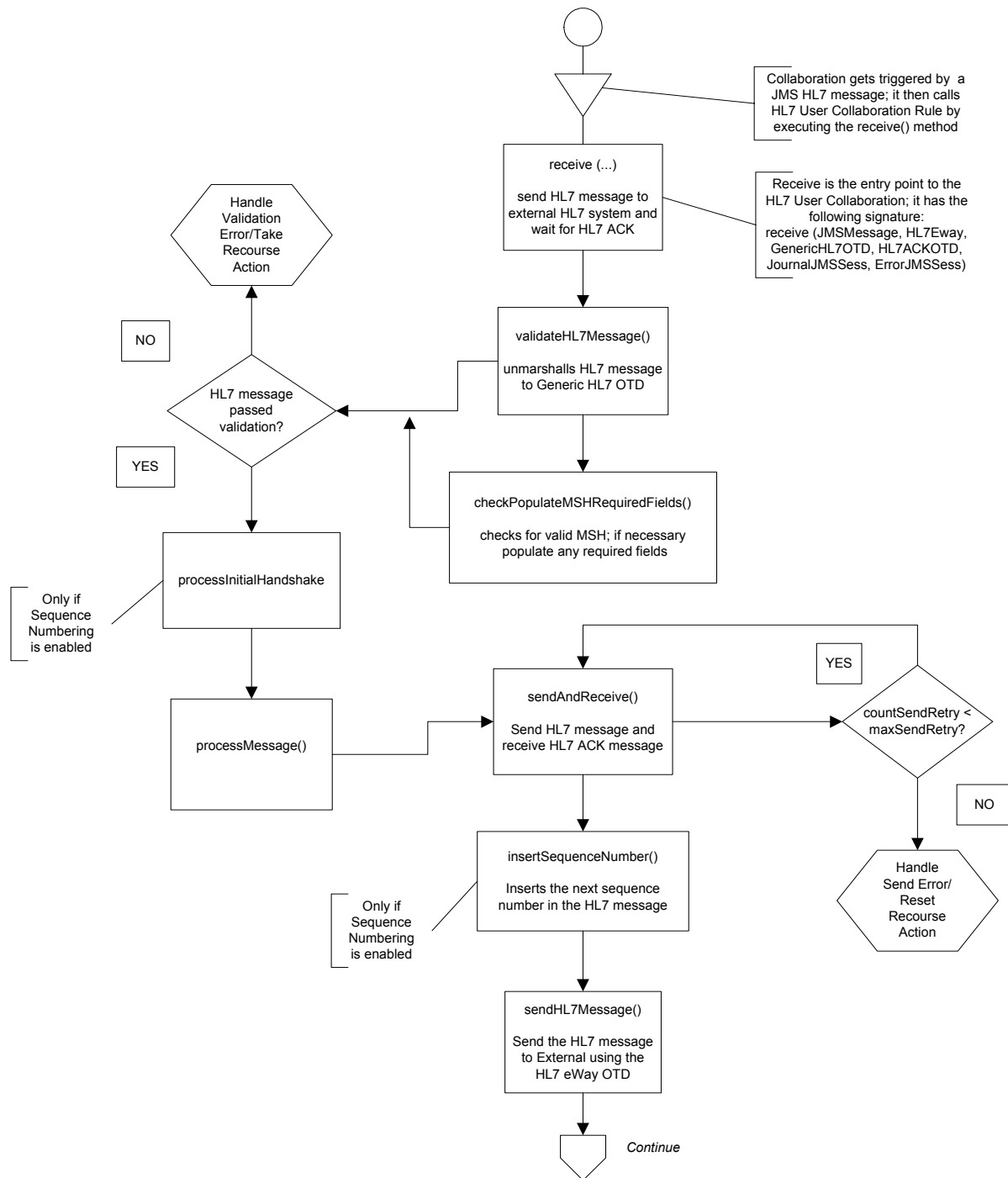
If the message does not pass validation, an error occurs and the associated recourse action is applied. If the HL7 message passes validation, the message moves on to **processInitialHandshake()** to receive a sequence number (if sequences numbering is enabled). The Collaboration takes the sequence number from the Sequence Numbering file and determines the next number to use. This number is then inserted into the HL7 message.

Next, the message moves on to **processMessage()**, which calls the helper method, **sendAndReceive()**. The **sendAndReceive** method sends the HL7 message, waits for an HL7 ACK message, and processes the ACK or NAK. The validation also checks the message structure to see if the message is unmarshaled. If a valid ACK is not received, it continues to send the HL7 message up to the configured number of retries, at which time an error occurs and the associated recourse action is taken. If a valid ACK is received, the message moves on to **insertSequenceNumber()**.

If sequence numbering is enabled, the **insertSequenceNumber** method inserts the sequence number and call **sendHL7Message()**.

The **sendHL7Message** method sends the HL7 message to the external using the HL7 eWay OTD (see [Figure 20 on page 112](#)).

Figure 20 Outbound Collaboration - Part 1



Outbound Collaboration - Part 2

The Collaboration receives the HL7 ACK or NAK from the external using **receiveHL7AckNak()**. If an exception occurs due to incomplete data, and the eWay fails to read the data within the configured number of retries, the associated recourse action is taken. If the exception is due to no response, the associated recourse action is taken. If no exception occurs, the ACK or NAK message moves on to **isAckMessage()**, which validates the message to determine whether the message is an ACK or a NAK.

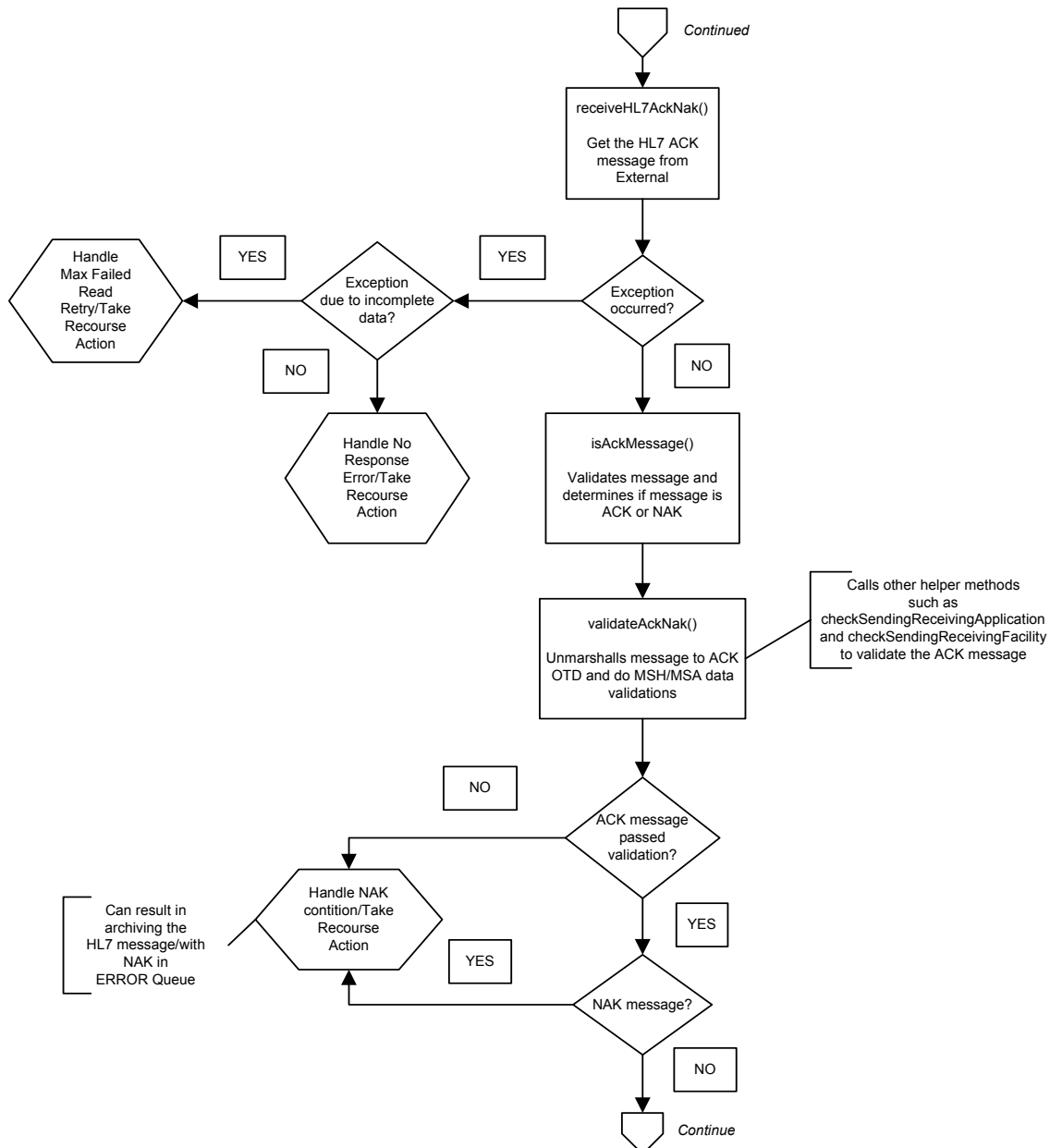
Next the `validateAckNak` method unmarshalls the message to the ACK OTD and does validation on the MSH/MSA data. In addition, it also calls other helper methods such as `checkSendingReceivingApplication()` and `checkSendingReceivingFacility()` to validate the ACK message.

If the message does not pass validation it is handled as a NAK, the associated recourse action is taken, and the message is archived in the Error Queue.

If the message is a NAK, the associated recourse action is taken, and the message is archived in the Error Queue, along with the NAK message, as a JMS property.

If the message is an ACK and passes validation, the message is sent on to the `journalMessage` method (see Figure 21).

Figure 21 Outbound Collaboration - Part 2

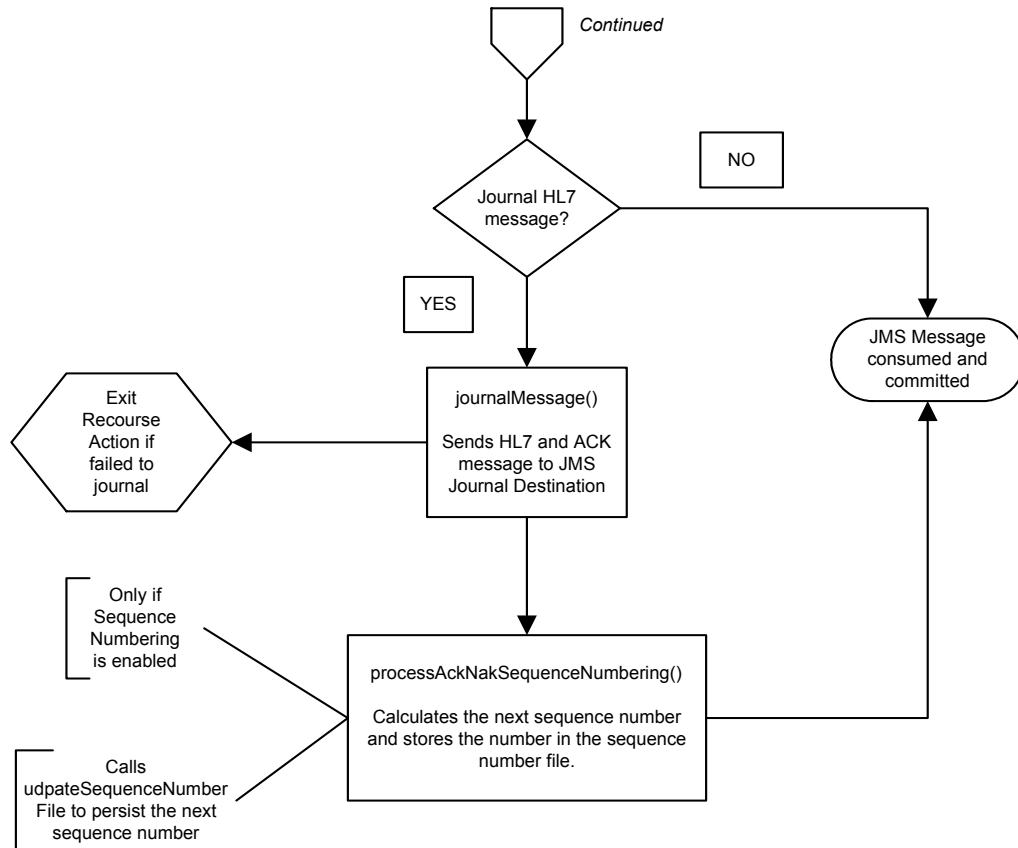


Outbound Collaboration - Part 3

If the ACK message validates, the HL7 message and ACK message are sent to the JMS Journal Destination. If the message fails to journal, the associated recourse action is taken.

If Sequence Numbering is enabled, the **processAckNakSequenceNumbering** method calculates the next sequence number and stores the number in the sequence number file, calling **updateSequenceNumberFile** to persist the next sequence number (see Figure 22).

Figure 22 Outbound Collaboration - Part 3



5.1.3 HL7 Outbound Test Collaboration

In addition to the Inbound and Outbound HL7 Collaborations, an Outbound test Collaboration, **jcdHL7OutboundTestDriver**, is provided to test the HL7 Outbound and HL7 Outbound Delayed ACK samples.

The **jcdHL7OutboundTestDriver** Collaboration simply picks up HL7 messages from the File eWay and sends the message to the JMS queue. This is used by the sample Projects to test the number of HL7 messages processed per minute.

5.2 Creating a Copy of a Project

It is recommended that you retain the prjHL7Inbound and prjHL7Outbound Projects as they are and create copies of the Projects to use as a basis for your new Projects. To create a copy of a Project, the original Project is first exported to a file, the name of the original Project is changed, and the new Project is imported into the Repository.

Exporting a Project

To export a Project or Environment using Enterprise Designer do the following:

- 1 From the Enterprise Explorer tree, right-click the Project you wish to export. Select **Export** from the shortcut menu. The **Export Manager** dialog box appears.
- 2 The selected Project appears in the **Selected Projects** pane of the Export Manager; if not, select the Project and click the arrow (>) button to add that Project to the **Selected Projects** pane.
- 3 Click the Browse button and select an appropriate directory to save the exported Project.
- 4 Click Export. The Project is zipped and saved to the chosen directory.

Note: *The Enterprise Designer will not import identically named Projects to the same root. Once your Project has been exported, rename the original Project (for example, prjHL7InboundSave) before importing the same Project.*

Importing a Project

To import a Project or Environment using Enterprise Designer do the following:

- 1 Save any current changes before importing a new Project. Any unsaved changes may be lost during import.
- 2 From the Enterprise Explorer tree, right-click the Repository and select **Import** from the shortcut menu. The Import Manager dialog box appears.
- 3 Click **Browse** and select the Project that you want to import.
- 4 Click **Import**. The new Project is imported to the repository.

5.3 Customizing Predefined Collaborations

The predefined Collaborations are designed to be extended and modified, however, for HL7 compliant systems this is not necessary. If you need to modify an HL7 Collaboration, it is strongly suggested that these “template” Collaborations be used as the basis for any new Collaborations. Therefore, it is important to maintain the original predefined **jcdHL7Inbound** and **jcdHL7Outbound** Collaborations in their initial form for future use.

Creating copies of the HL7 Collaborations

Production Projects contain several Collaborations. To create multiple copies of the HL7 Collaborations for your Project, copy and paste the original Collaborations into your project as follows:

- 1 Right-click the Collaboration you want to copy and select **Copy** from the shortcut menu. For example, from the **prjHL7Inbound** Project, right-click and copy **jcdHL7Inbound**.
- 2 Right-click the Project to which you are copying the Collaboration, and select **Paste** from the shortcut menu. A numbered copy of the Collaboration appears in you Project. For example, **jcdHL7Inbound_1**. Repeat this step for additional copies of the Collaboration.
- 3 To add the new Collaborations to your Connectivity Maps, drag and drop each new Collaboration from the Project Explorer tree, to your Connectivity Map canvas. The new Collaborations can now be associate (bind) it with the Project's components.

5.3.1 Adding an OTD to an Existing Collaboration

In some cases, a specific HL7 message or messages may need to be added to the Collaboration. To add an HL7 OTD to an existing Collaboration do the following:

- 1 From the Enterprise Explorer tree, right-click the Collaboration to which you want to add the HL7 OTD. From the shortcut menu, select Properties. The Collaboration Definition (Java) Properties dialog box appears (for this example, the jcdHL7Outbound Collaboration is displayed in Figure 23).

Figure 23 Collaboration Properties

Collaboration Definition (Java) Properties

Collaboration Name:

Operation Configuration:

Operation Name:

Input Message:

☐ Output Message:

☐ Callable as an external SOAP Web Service

Object Type Definitions:

OTD	Instance Name
SeeBeyond.eWays.HL7.HL7eWay	otdHL7eWay_1
SeeBeyond.eGate.JMS	otdJMS_JOURNAL
SeeBeyond.eGate.JMS	otdJMS_ERROR
SeeBeyond.OTD Library.HL7.Generic.HL7_GENERIC_EVT	otdHL7_GENERIC_EVT_1
SeeBeyond.OTD Library.HL7.Generic.HL7_ACK	otdHL7_ACK_1

- 2 Select **Choose an Existing Operation** as the value for the **Operation Configuration** field.
- 3 Click **Add**. From the **Select** dialog box, browse to and select the appropriate HL7 OTD. The OTD is added to the Properties's OTDs.
- 4 Once the Collaboration has been modified and additional Java code or Business Rules have been added, it must be dragged and dropped again from the Project Explorer tree to the associated Service, and all associations (bindings) must be re-mapped in the Connectivity Map.

For more information on how to add a Java Collaboration to a Service see the *Sun SeeBeyond eGate™ Integrator User's Guide* and the *Sun SeeBeyond eGate™ Integrator Tutorial*.

Working With HL7 OTDs

This chapter provides information on how to use the Sun SeeBeyond Enterprise Designer's OTD Editor to select, modify, and test HL7 OTDs.

What's in this Chapter

- [Viewing an OTD using the OTD Editor](#) on page 119
- [Modifying an OTD Using the OTD Editor](#) on page 120
- [Using the OTD Tester](#) on page 134

6.1 The OTD Editor

The OTD Editor displays the structure of a selected Object Type Definition (OTD) and allows you to verify its operation with a built-in tester. You can also use the editor to create and modify OTDs. For an overview of Object Type Definitions, OTD structure, and the OTD Editor, see the *Sun SeeBeyond eGate Integrator User's Guide*. The eGate Integrator User's Guide defines all available OTD properties and provides a description of all of the OTD Editor's features.

The following chapter provides information specific to using Library OTDs with the OTD Editor. These OTDs are templates that correspond to message types used by industry-specific data exchange systems and open-source standards. The templates are pre-defined and can be used as-is, or modified using the OTD Editor.

6.1.1 Generic HL7 OTDs

The HL7 eWay requires special abstracted OTDs that represent all of the versions and types of HL7 messages. Two have been included with the eWay:

- HL7_GENERIC_EVT: Parses any valid HL7 message.
- HL7_ACK: Parses any valid HL7 ACK or NAK.

The purpose of these OTDs (messages) is to allow the MSH fields to implement the HL7 messaging protocol. It is assumed that other processes will transform incoming and outgoing messages to and from other formats.

The HL7 Generic OTDs are found on the Project Explorer tree under **Sun SeeBeyond > OTD Library > HL7 > Generic**.

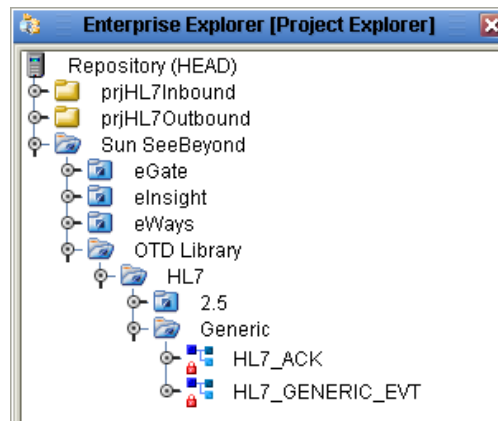
6.2 Viewing an OTD using the OTD Editor

The OTD Editor displays the structure of a selected Object Type Definition (OTD) and allows you to verify its operation with a built-in tester. You can also use the editor to create and modify User-Defined OTDs.

To view the HL7 Generic OTD or an HL7 Library OTD (if you are using the HL7 eWay in conjunction with the Sun SeeBeyond HL7 OTD Library) using the OTD Editor, do the following:

- 1 From the Enterprise Designer's Project Explorer tree, expand the **Sun SeeBeyond OTD Library** directory, the **HL7** directory, and either the **HL7_GENERIC_EVT** folder or the folder for the appropriate HL7 Library version (see Figure 24). Only the version or versions you install will be displayed in your build.

Figure 24 Selecting an OTD from the Project Explorer Tree

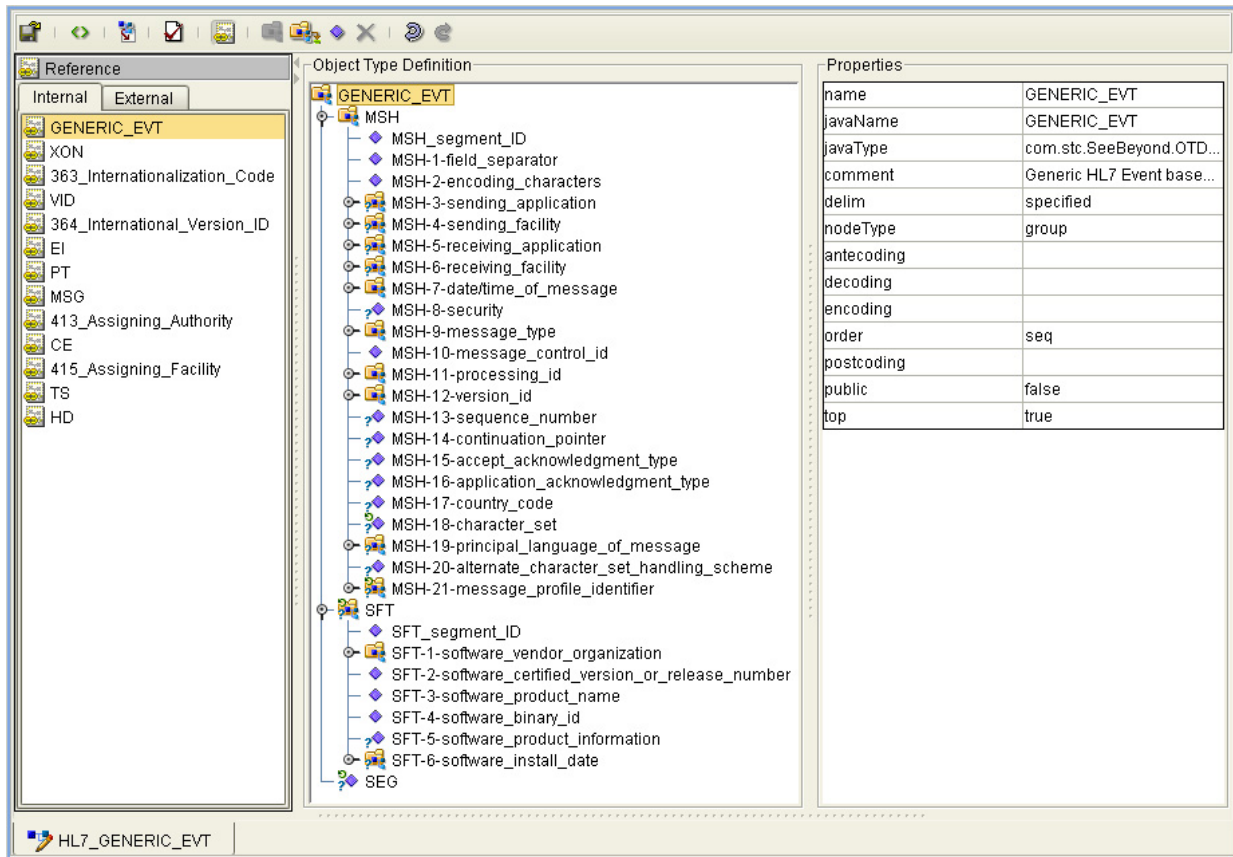


- 2 The OTDs available under OTD Library in the Project Editor are protected (read-only). You can view the OTD in **read-only** mode by double-clicking the OTD in its current location.

Copying an OTD to your Project

- 3 To view the OTD in an editable mode, you must copy and paste the OTD to your Project. Right-click the OTD and select **Copy** from the shortcut menu, then right-click your Project and select Paste from the shortcut menu. The OTD is added to your Project on the Project Explorer tree.
- 4 To view the copied OTD, double-click the copied OTD. The editable OTD appears in the OTD Editor. Notice that for the Library OTDs, the OTD segments are still write protected. The OTDs properties can only be modified from the Root node at this point. The copied HL7_GENERIC_EVT OTD is fully editable (see [Figure 25 on page 120](#)).

Figure 25 The OTD Editor - HL7_GENERIC_EVT



- From the editor's Object Type Definition pane, select any of the OTDs nodes or sub-nodes to see the nodes properties displayed in the editor's Properties pane.

For an overview of Object Type Definitions, OTD structure, and the OTD Editor, see the *Sun SeeBeyond eGate Integrator User's Guide*.

6.3 Modifying an OTD Using the OTD Editor

OTD Check Out and Check In

The Generic HL7 OTDs are located in the Project Explorer's **Sun SeeBeyond** folder. These OTDs are protected and cannot be modified. This assures that the original OTDs are always available in their original form. To modify an OTD, you must first copy and paste the OTD from the Sun SeeBeyond > OTD Library folder to your Project.

Version Control is available for any OTD you save to your Projects. To check-in or check-out an OTD, right-click the OTD from the Project Explorer tree, and select **Version Control > Check Out** or **Check In** from the shortcut menu. When an OTD has been checked-in, the OTD file icon appears in the Project Explorer tree as "locked" (The icon includes a red padlock).

6.3.1 Editing an OTD's Root Properties

An HL7 OTD copied to your Project can only be edited from the Root node. Each of the OTD's segments are write protected. The OTD segments are visible from the Reference pane of the OTD Editor. This Reference pane contains internal and external templates for the OTD file. To directions on editing specific segments of an HL7 Library OTD, see the *Sun SeeBeyond HL7 OTD Library User's Guide*. For more information regarding OTD properties, see ["OTD Properties" on page 128](#).

Root Node Properties

The set of properties associated with Root nodes is shown in Figure 26.

Figure 26 Root Node Properties (delim nodeType)

Properties	
name	GENERIC_EVT
javaName	GENERIC_EVT
javaType	com.stc.SeeBeyond.OTD...
comment	Generic HL7 Event based...
delim	specified
nodeType	array
showDelim	\r
antecoding	
decoding	
encoding	
order	seq
postcoding	
public	false
top	true

Table 34 Node Properties

Node Property Descriptions	
name	Node display name. This can be a virtually-arbitrary string.
javaName	Property accessor basename. This is normally derived from the display name, modified to suit the restrictions on Java identifiers, and supplied automatically by eGate.
javaType	Java type; automatically assigned, not editable.
comment	Free-form text (no run-time effect).
delim	Specified delimiter. See Specifying Delimiters on page 124.
nodeType	Governs the marshal/unmarshal format. See Specifying the Node Type on page 132.

Node Property Descriptions (Continued)	
antecoding	Specifies the input data coding (see Specifying Data Encoding on page 219). If this property is not specified, the value specified for the decoding property will be used for the input data. This property is displayed only when the top property is set to true .
decoding	Specifies the unmarshal coding (see Specifying Data Encoding on page 219). (It is recommended to use UTF-8 for DBCS data, since the hex value of some ASCII delimiter may coincide with a hex value contained within a double-byte character.) This property is displayed only when the top property is set to true .
encoding	Specifies the marshal coding (see Specifying Data Encoding on page 219). This property is displayed only when the top property is set to true.
order	Specifies the ordering of the Root node's children: <ul style="list-style-type: none"> ▪ seq: specifies that the child nodes must appear in sequence. ▪ any: specifies that the child nodes can appear in any order. ▪ mix:
postcoding	Specifies the output data coding (see Specifying Data Encoding on page 219). If this property is not specified, the value specified for the encoding property will be used for the output data. This property is displayed only when the top property is set to true .
public	Reserved for future development
top	Flag on Root node: support marshal/unmarshal (T/F).

Properties edited from the Root node are applied inclusively to the OTD. For example, a level three delimiter, changed from the Root node, applies to all level three node delimiters. (The properties for specific segments can be exclusively edited, but to do this you must copy and paste the specific OTD that the segment refers to into your Project. For more information on editing specific segments, see the *Sun SeeBeyond HL7 OTD Library User's Guide*.

To edit an OTD's Root node properties, do the following:

- 1 Copy and paste the OTD to your Project. The OTD is added to your Project in the Project Explorer tree.
- 2 Double-click the OTD to open your Project in the OTD Editor.
- 3 From the editor's Object Type Definition pane, select the **Root node** of the OTD. The Root properties are displayed in the editor's Properties pane.
- 4 From the Properties pane, click on any property field to edit the property.

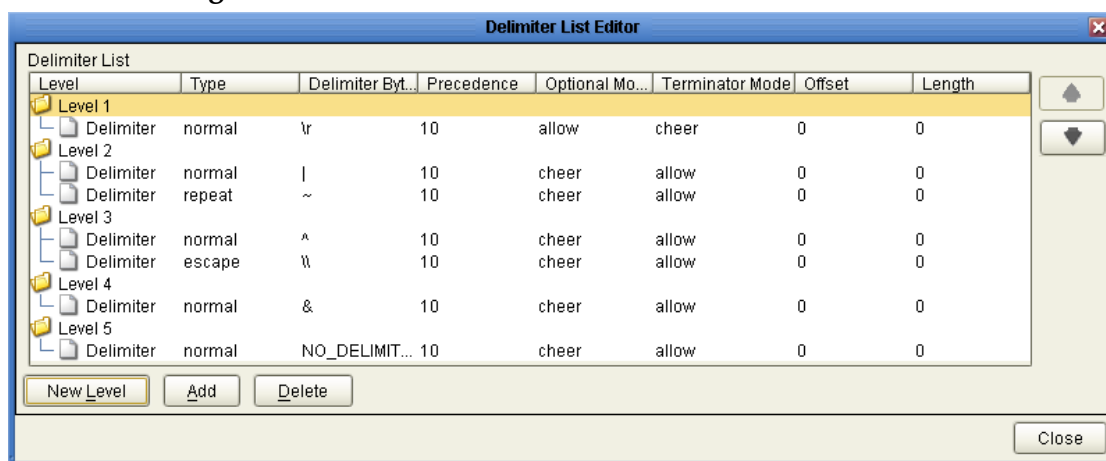
6.3.2 Editing the OTD Delimiters

Delimiters for all node levels are set (and modified) from the Root node. Be aware that the default level 1 delimiter character is a non-ASCII character. Once it has been changed it cannot be typed back in as a character (but can be pasted). For information about editing a specific segment of the OTD, see the *Sun SeeBeyond HL7 OTD Library User's Guide*.

To edit the Delimiters from the Root node do the following:

- 1 From the OTD Editor, select the Root node in the Object Type Definition pane (for this example ADT_A02). From the Properties pane, double-click the **delim** properties field. An ellipsis (...) button appears in the field. Click the ellipsis button. The **Delimiter List Editor** appears (see [Figure 27 on page 123](#)).

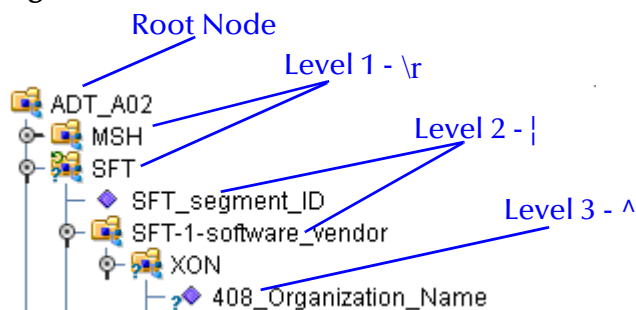
Figure 27 HL7 OTD Editor - Delimiter List Editor



- 2 Double-clicking any field in the OTD Editor's Properties field, for any level, makes the field editable or displays a list of options. Double-click the Delimiter Bytes field for level 3 (see Figure 27). Change the current delimiter character to a pound sign (#), **Tab** to the next field, and click **OK**.

The delimiter for all level 3 nodes in the OTD is now a pound sign (#), unless it is specified differently for a specific segment. Figure 28 displays an example of various levels in the Object Type Definition tree, from the Root node.

Figure 28 Node Levels From the Root Node



6.3.3 Changing HL7 Standard Encoding Characters

All HL7 OTDs have a predefined list of delimiters, per the definition in the HL7 Standard. If you elect to change the delimiter encoding characters in your HL7 messages, you must change the delimiter in the OTD from the Root node using the OTD Editor, to match the delimiters used in the HL7 message.

The delimiter encoding character field is a fixed-length field of four encoding characters plus the field separator. The fifth (extra) character is necessary for the segment field separator.

If you wish to validate against the encoding characters, modify the pre-built Collaboration rules as follows:

```
// first unmarshal the HL7 OTD payload

// then get the encoding character field:
String encodingChars = otdHL7_GENERIC_EVT_1.getMsh().getMsh2EncodingCharacters();

if (!encodingChars.equals("<customer_encoding_characters")) {
    validated = false;
    ErrorMessage = "Validation Failure: Receiving Facility";
    log( LOG_LEVEL_ERROR, "Validate HL7 Message failed: Encoding character field" );
}
```

6.3.4 Specifying Delimiters

A node defines a set of delimiters to be used in the external data representation for itself and its descendents in the hierarchical data structure. If a node defines a delimiter list, this negates any effect of any ancestor's delimiter list on itself and its descendents. The delimiter list is typically specified on the Root node.

For example, if you want to parse the following data:

a^b|c^d|e

you might define an OTD as follows:

- demo-otd
 - ♦ element1
 - ♦ field1
 - ♦ field2
 - ♦ element2
 - ♦ field3
 - ♦ field4
 - ♦ field5

The delimiter list for this OTD will be specified on the *demo-otd* element, so that it applies to the entire OTD, and will have two levels:

- Level 1
 - ♦ Delimiter |
- Level 2
 - ♦ Delimiter ^

Level 1's delimiter applies to the two elements and field5, and level 2's delimiter applies to fields 1 through 4.

Delimiter lists can be much more complex than this very simple example. For instance, you can create multiple delimiters of different types at any given level, and you can specify a delimiter list on any node within the OTD—not only the Root node as shown in the example. See [Modifying an OTD Using the OTD Editor](#) on page 120 for a description of the procedure for creating a Delimiter List.

Delimiter Properties

Delimiters are defined using the Delimiter List Editor (see Figure 27).

The Delimiter properties and values are displayed in Table 35.

Table 35

Delimiter Properties and Value Options		
Property	Option	Description
Level		Child level beneath defining node.
Type	escape	Escape sequence.
	repeat	Array delimiter/separator.
	normal	Terminator.
Delimiter Bytes		Delimiter (single or multiple characters).
Precedence		See Precedence on page 127.
Optional Mode	never	Do not allow on input, do not emit on output (empty field between delimiters implies zero length data field).
	allow	Skip empty field if present; if absent, do not delimit on output.
	cheer	Skip empty field if present; if absent, do delimit on output.
	force	Require empty, delimited field on input; always delimit on output.
Terminator Mode	never	Do not allow on input, do not emit on output (pure separator).
	allow	Allow on input, do not emit on output.
	cheer	Allow on input, always emit on output.
	force	Require on input, always emit on output (pure terminator).

Type Property - Escape Option

An *escape* delimiter is simply a sequence that will be recognized *and ignored* during parsing. Its purpose is to allow the use of escape sequences to embed byte sequences in data that would otherwise be seen as delimiter occurrences.

For example, if there is a normal delimiter “+” at a given level, and we define an escape delimiter “\+”, then **aaa+b\+c+ddd** will parse as three fields: **aaa**, **b\+c**, and **ddd**. If the escape delimiter were not defined, the sequence would then parse as four fields: **aaa**, **b**, **c**, and **ddd**.

If there is *only* an escape delimiter on a given level, however, it presents a *no delimiter defined* situation for **delim** and **array** nodes.

Delimiter Bytes

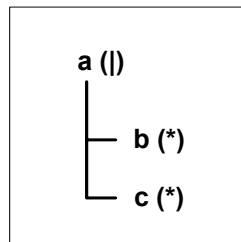
There is essentially no limitation on what characters you can use as delimiters; however, you obviously want to avoid characters that can be confused with data or interfere with escape sequences. The backslash (\) is normally used as an escape character (the HL7 protocol uses a double backslash as part of an escape sequence that provides special text formatting instructions).

Note: You should avoid using a colon (:) as a delimiter character, since it is used as a literal in system-generated time strings. This can interfere with recovery procedures, for example following a Domain shutdown.

Terminator Mode Property

Consider the tree structure shown in [Figure 29 on page 126](#), where the node **a** has a pipe (|) as its delimiter, the sub-node **b** has a tilde (~) as its delimiter, and sub-node **c** has an asterisk (*) as its delimiter.

Figure 29 Terminal Type Property Example



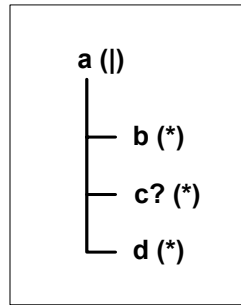
Option	Input	Output
never	c	c
allow	c or c*	c
cheer	c or c*	c*
force	c*	c*

Optional Mode Property

Consider the tree structures shown in [Figure 30 on page 127](#) and [Figure 31 on page 127](#), where the node **a** has a pipe (|) as its delimiter, and the sub-nodes **b**, **c**, and **d** all have asterisks (*) as their delimiters.

- **Example 1:** Sub-node **c** is *optional*. (Sub-node **c** and sub-node **d** must have different values for the *match* parameter.)

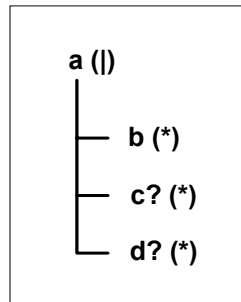
Figure 30 Optional Property (Example 1)



Option	Input	Output
never	b*d 	b*d
allow	b**d 	b*d
cheer	b**d 	b**d
force	b**d 	b**d

- **Example 2:** Both sub-node **c** and sub-node **d** are *optional*.

Figure 31 Optional Property (Example 2)



Option	Input	Output
never	b 	b
allow	b , b*, or b** 	b
cheer	b , b*, or b** 	b**
force	b** 	b**

Precedence

Precedence (see [Figure 28 on page 123](#)) indicates the priority of a certain delimiter, relative to the other delimiters. By default, all delimiters are at precedence 10, which means they are all considered the same; fixed fields are hard-coded at precedence 10. Delimiters on parent nodes are not considered when parsing the child fields; only the child's delimiter (or if it is a fixed field, its length).

Changing the precedence of a delimiter will cause them to be applied to the input data-stream in different ways. For example:

- *Root node*
 - ♦ element (type delim, delimiter = "^", repeat)
 - ♦ field1 (type fixed, length = 5)
 - ♦ field2 (type fixed, length = 8, optional)

Although this will parse 'abcde12345678^zyxvuABCDEFGH', it will *not* parse the text 'abcde^zyxvuABCDEFGH' even though the second fixed field is optional. The reason is that the element's delimiter is ignored within the fixed field because they have the same precedence. If you want the element's delimiter to be examined within the fixed field data, you must change its precedence, for example:

- *Root node*
 - ♦ element (type delim, delimiter = "^", repeat, **precedence = 11**)
 - ♦ field1 (type fixed, length = 5)
 - ♦ field2 (type fixed, length = 8, optional)

This will successfully parse the text 'abcde^zyxvuABCDEFGH'.

6.3.5 OTD Properties

The Object Type Definition pane (center pane) of the OTD Editor displays the nodes, elements, and fields of the OTD. When any of these are selected, the item's properties are displayed in the Properties pane.

Node Properties

When an HL7 OTD is opened in the OTD Editor, the properties of the Root node are displayed in the Properties pane. The configurable node properties are displayed in Table 36.

Table 36 Node Properties

Node Property Descriptions	
name	Node display name. This can be a virtually-arbitrary string.
javaName	Property accessor basename. This is normally derived from the display name, modified to suit the restrictions on Java identifiers, and supplied automatically by eGate.
javaType	Java type; automatically assigned, not editable.
comment	Free-form text (no run-time effect).
delim	Specified delimiter. See Specifying Delimiters on page 124.
nodeType	Governs the marshal/unmarshal format. See Specifying the Node Type on page 132.

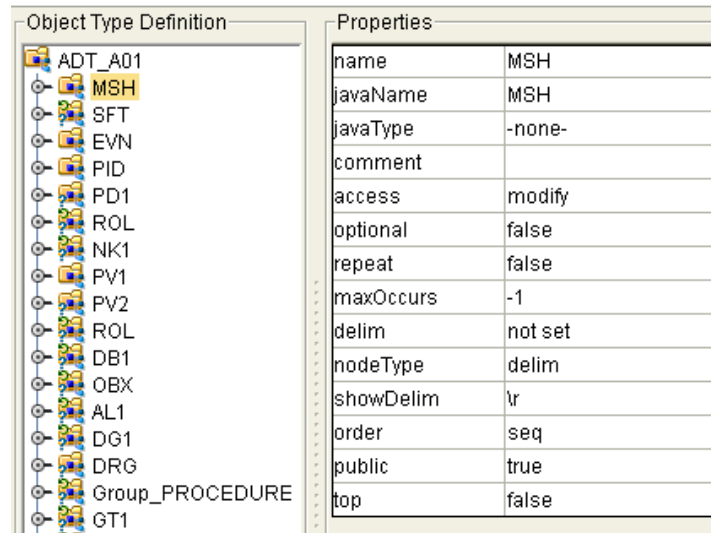
Node Property Descriptions (Continued)	
antecoding	Specifies the input data coding (see Specifying Data Encoding on page 219). If this property is not specified, the value specified for the decoding property will be used for the input data. This property is displayed only when the top property is set to true .
decoding	Specifies the unmarshal coding (see Specifying Data Encoding on page 219). (It is recommended to use UTF-8 for DBCS data, since the hex value of some ASCII delimiter may coincide with a hex value contained within a double-byte character.) This property is displayed only when the top property is set to true .
encoding	Specifies the marshal coding (see Specifying Data Encoding on page 219). This property is displayed only when the top property is set to true.
order	Specifies the ordering of the Root node's children: <ul style="list-style-type: none"> ▪ seq: specifies that the child nodes must appear in sequence. ▪ any: specifies that the child nodes can appear in any order. ▪ mix:
postcoding	Specifies the output data coding (see Specifying Data Encoding on page 219). If this property is not specified, the value specified for the encoding property will be used for the output data. This property is displayed only when the top property is set to true .
public	Reserved for future development
showDelim	If nodeType is delimited.
top	Flag on Root node: support marshal/unmarshal (T/F).

Important: Do not modify the *javaName* property.

Element Properties

The set of properties associated with the element level is shown in Figure 32.

Figure 32 OTD Editor - OTD Element Properties



The configurable element properties are displayed in Table 37

Table 37

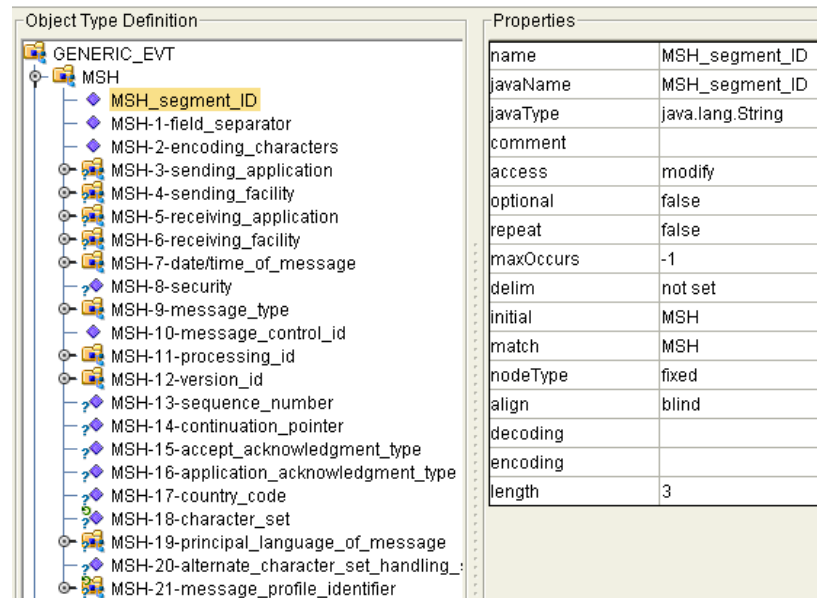
Element Property Descriptions	
name	Element display name.
javaName	Property accessor basename.
javaType	Java type; automatically assigned, not editable.
comment	Free-form text (no run-time effect).
access	Access specification.
optional	Flag: Can the element be absent? (T/F) Not applicable to Root, or child of a choice Node.
repeat	Flag: Can the node appear multiple times? (T/F) Not applicable to Root, or child of a choice Node.
maxOccurs	Specifies the maximum number of occurrences of the node if the node is repeating. Property has no effect if node is non-repeating, but may show error during validation if set to value >1.
delim	Delimiter specification (see Specifying Delimiters on page 124).
nodeType	Governs the marshal/unmarshal format.
showDelim	If nodeType is delimited,
Public	For future use, not currently active.
Top	Specifies whether or not marshal/unmarshal is supported (true or false). The default value is true .

Important: Do not modify the *javaName* property.

Field Properties

The set of properties associated with the field level is shown in Figure 33.

Figure 33 OTD Editor - OTD Field Properties



The configurable field properties are displayed in Table 38

Table 38

Field Property Descriptions	
name	Field display name.
javaName	Property accessor basename.
javaType	Java type: can be either java.lang.String or byte array (byte[]).
comment	Free-form text (no run-time effect).
access	Access specification.
optional	Specifies whether or not the field can be absent from an instance. Clicking the <i>Value</i> field toggles between true and false . Not applicable if the field is the child of a <i>choice element</i> node.
repeat	Specifies whether or not the node can appear multiple times. Clicking the <i>Value</i> field toggles between true and false . Not applicable if the field is the child of a <i>choice element</i> node.
maxOccurs	Specifies the maximum number of occurrences of the node if the node is repeating. Property has no effect if node is non-repeating, but may show error during validation if set to value >1.

Field Property Descriptions (Continued)	
delim	Delimiter specification (see Specifying Delimiters on page 124).
initial	Initial field value, set when the parent node is created or reset. When provided, it is assigned to the node before the node is populated with any data.
match	If nodeType is delimited, performs exact match to the data.
nodeType	Governs the marshal/unmarshal format.
align	Specifies the byte alignment criteria for the match property.
decoding	Displayed only if nodeType is fixed. Specifies the unmarshal coding. (It is recommended to use UTF-8 for DBCS data, since the hex value of some ASCII delimiter may coincide with a hex value contained within a doublebyte character.)
encoding	Displayed only if nodeType is fixed. Specifies the marshal coding.
length	Displayed only if nodeType is <i>fixed</i> . Specifies the length of the field; the default value is 0 .

Important: Do not modify the *javaName* property.

6.3.6 Specifying the Node Type

Click on the nodeType properties field to activate the field for editing. Click the arrow button to display the selection menu. Descriptions of the property options are listed in Table 39.

Table 39

Node Type Property Options				
Option	Description	Element	Field	Internal
array	Array is a delimited structure. If repeated, occurrences are separated by the <i>repeat</i> delimiter. The last occurrence may be terminated by a <i>normal</i> delimiter.	Yes	Yes	simple or group
delim	Delim (<i>delimited</i>) structure. If repeated, occurrences are separated by a <i>normal</i> delimiter.	Yes	Yes	simple or group
fixed	Fixed indicates a fixed length, which is specified by non-negative integer (or zero to indicate end of parent node data).	Yes	Yes	simple or group
group	Group provides organizational grouping for purposes such as repetition. Applies only to elements.	Yes	No	group

Node Type Property Options (Continued)				
Option	Description	Element	Field	Internal
trans	Trans (<i>transient</i>) appears only in an internal tree as a scratch pad field. It does not appear in external data representation, and can only have <i>trans</i> nodeTypes as children.	Yes	Yes	choice, simple, or group

Note: If you move an OTD node, you must reset the *nodeType* for that node.

Node Management

The OTD Editor allows you to:

- **Add** nodes and elements to an OTD.
- **Delete** nodes and elements from an OTD.

When a node is *deleted*, both the node and its associated *children* (data elements) are deleted.

- **Prune** nodes in an OTD.

When a node is *pruned*, only its associated *children* (data elements) are deleted, while the node itself is preserved. Pruning can only be performed on nodes.

These commands are accessed from the node context menu.

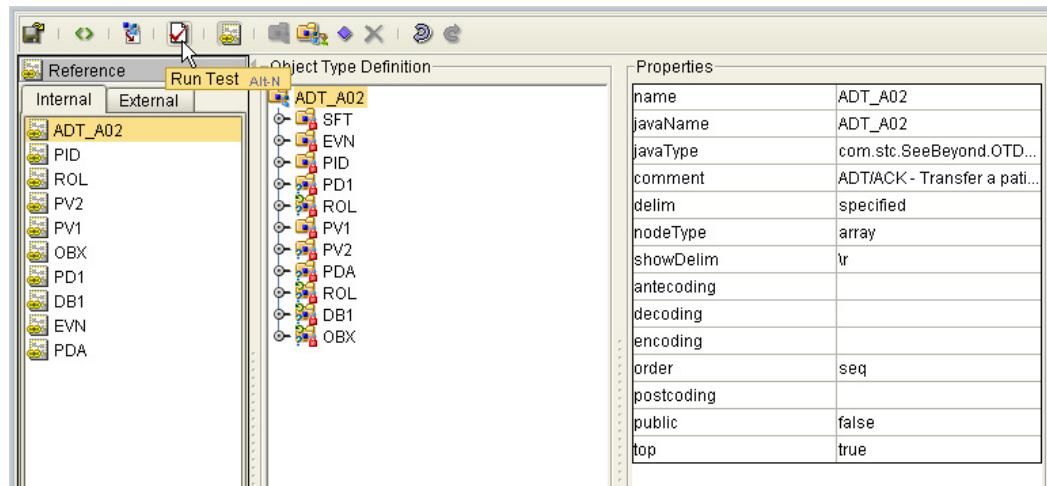
6.4 Using the OTD Tester

The OTD Tester allows you to simulate the operation of a Collaboration containing a specific OTD, thereby checking the correctness of the OTD during the design phase. You can enter input data values, perform the unmarshal and marshal operations, and also manipulate the OTD tree structure as a Collaboration might do by using the Add Instance and Delete Instance buttons. By using these latter features, you can prepare an output data file that can then be used as an input data file for testing purposes.

To use the OTD Tester

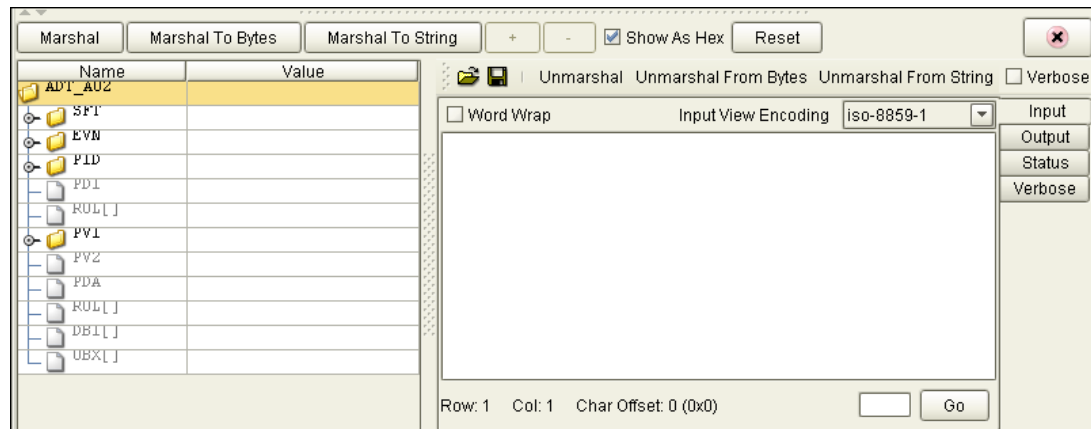
- 1 Open an OTD.
- 2 From the OTD Editor toolbar, click the **Run Test** icon (see Figure 34).

Figure 34 OTD Tester



This saves the currently displayed OTD to the Repository and displays the OTD Tester. The OTD Tester appears at the bottom of the OTD editor. Note that the data display panel offers four different data display modes. (see Figure 35). The Input mode is selected by default.

Figure 35 Test Panel Data Display



- 3 You can provide the input test data either by clicking the **Open a file** icon (the open folder) and selecting a data file, or by copying and pasting the data to the OTD Tester data panel.

Testing Data Unmarshal Functionality

- 4 Click the desired **unmarshal** command to unmarshal the data from the Input panel to the OTD tree.
- 5 To verify the unmarshal process, check the values of each element for correctness.

Figure 36 Object Elements and Values

Name	Value
MSH	
MSH-segment_ID	"MSH"
MSH-1-field_separator	" "
MSH-2-encoding_characters	"^~\& "
MSH-3-sending_application	
MSH-4-sending_facility	
MSH-5-receiving_application	
MSH-6-receiving_facility	
MSH-7-date/time_of_message	
TS	
353_Time	"200201291848"
354_Degree_of_Precision	null
MSH-8-security	""
MSH-9-message_type	
MSH-10-message_control_id	"CHPFADIT"
MSH-11-processing_id	
MSH-12-version_id	
VID	
362_Version_ID	"2.5"
363_Internationalization_Code	
364_International_Version_ID	
MSH-13-sequence_number	""
MSH-14-continuation_pointer	""

- 6 Save your input test data to a file for re-use by selecting the Input panel and clicking the Save icon.
- 7 If there are errors in your input data, the Status panel is automatically invoked, displaying the appropriate error messages. Confirmation of correct operation is also reported.

Interpreting Failed Parse Messages Using the Verbose Option

For selected OTDs, the Verbose option provides a trace of parsing actions during the unmarshal process to aid in debugging the OTD structure. Selecting the Verbose check box causes parsing information to appear on the Verbose panel. The format and content of the data display are OTD-specific.

Testing Data Unmarshal Functionality

- 1 From the OTD Tester, enter or change data values for each node in the Value column of the node table. Use the Add/Delete Instance (+/-) buttons to add or remove instances where appropriate.
- 2 Specify the appropriate output encoding by selecting the character encoding in the Input View Encoding drop-down menu.
- 3 Click the desired marshal button to marshal (serialize) the data. The output is displayed in the Output panel

For detailed information on the OTD Tester and its features, see the *Sun SeeBeyond eGate Integrator User's Guide*.

Working with the TCP/IP HL7 eWay Projects

This chapter describes how to use the TCP/IP HL7 eWay template Projects that are installed automatically as part of the TCP/IP HL7 eWay.

What's in This Chapter

- [TCP/IP HL7 Project Overview](#) on page 136
- [Completing a Project](#) on page 138
- [Creating an Environment](#) on page 139
- [Configuring the Properties](#) on page 141
- [Creating the Deployment Profile](#) on page 142
- [Running the Samples](#) on page 145

7.1 TCP/IP HL7 Project Overview

The TCP/IP HL7 eWay Projects provide predefined eWay components designed to be extended and modified for your specific Project requirements. This chapter provides directions for completing the TCP/IP HL7 sample Projects to test the eWay on your system and demonstrate how an HL7 eWay is implemented.

7.1.1 Project Components

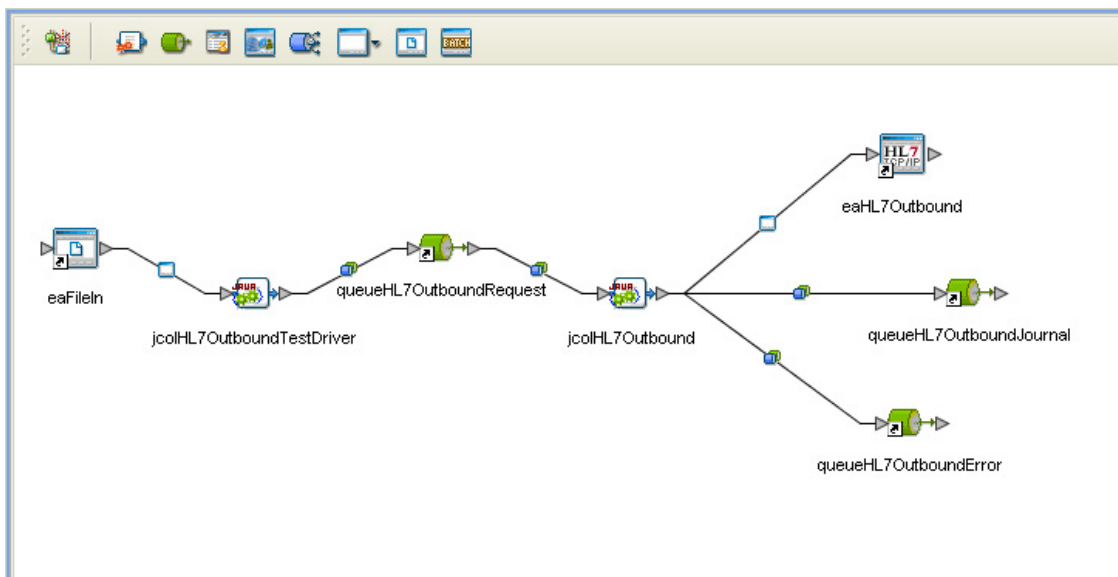
Projects are created using tools contained within the Enterprise Designer and are deployed to specific Logical Hosts in specific Environments by means of Deployment Profiles. Components developed for use in one Project can be used in another, and a Project can internally reference another Project. The components found in a typical Project include:

- **Services:** A service provides a framework for a process or a Collaboration.
- **External Applications:** External applications are logical representations of external software applications that are being integrated by the eGate system. These are linked to a Service by means of an eWay.
- **Schedulers:** A Scheduler allows a service to be performed at a prescribed interval.

- **Component Connections:** When you link two components on a Connectivity Map, the Enterprise Designer places either an eWay or JMS Client connection icon on the link, depending upon the type of components you are linking. When an External Application and a Collaboration are linked, the link contains an **eWay**. When a Service and a Message Destination (queue or topic) are linked, the link contains a **JMS Client Connection**.
- **Message Destinations:** A Message Destination is a container for stored data, and can follow either the topic or queue JMS model.
 - ♦ **Topic:** A topic is a message destination that conforms to the publish-and-subscribe messaging model (one to many).
 - ♦ **Queue:** A queue is a message destination that conforms to the point-to-point messaging model (one to one).

These components are graphically represented in a Project's Connectivity Map: for example, the cmHL7Outbound Connectivity Map as displayed in Figure 37. This Connectivity Map does not contain a Scheduler or any Topics.

Figure 37 Connectivity Map - cmHL7Outbound



7.1.2 TCP/IP HL7 eWay Sample Projects

The TCP/IP HL7 eWay includes two sample Projects, **prjHL7Inbound** and **prjHL7Outbound**. Each of these Projects demonstrate multiple operations (Collaborations).

prjHL7Inbound Project

The prjHL7Inbound Project includes the following Collaboration:

- **HL7Inbound Collaboration:** A standard inbound HL7 messaging operation that receives HL7 messages from an external system, sends an acknowledgement of the message, provides sequence numbering, writes the HL7 message to a JMS data queue, and also writes the HL7 message and ACK to a JMS Journal queue. For more

information see [Standard Inbound Message Mode Data Flow and Architecture](#) on page 27.

- **HL7ForwardMSG Inbound Collaboration:** Inbound Forward Message mode is use with the Outbound Delayed ACK. Its purpose is to get a message from an outbound Forwarder and return and acknowledgement. For more information see [Inbound Receiver Message Mode](#) on page 28.

prjHL7Outbound

The prjHL7Outbound Project includes the following samples:

- **HL7Outbound Collaboration:** A standard outbound HL7 messaging operation that receives the HL7 message from the JMS data queue, provides sequence numbering, sends the HL7 message to an external system, receives an acknowledgement from the external system, and writes the HL7 message and the ACK to a JMS Journal queue. For more information see [Outbound Standard Messaging Mode](#) on page 31.
- **HL7Forward Outbound Collaboration:** Outbound Forward Message is use with the Delayed ACK. Its purpose is to get a message from a JMS queue and send it to an external system. No validation is preformed. For more information see [Outbound Forwarder Role](#) on page 33.
- **HL7OutboundDelayedAck Collaboration:** Delayed Acknowledgement is similar to HL7Outbound, but the initial acknowledgement is received from the receiving system. After the Sender receives the first ACK, it waits for a second ACK that indicates that the message was received by the external HL7 system. For more information see [Outbound Delayed ACK Role](#) on page 32.

7.2 Completing a Project

To complete a TCP/IP HL7 eWay sample Project do the following:

- Import the sample Projects (see [Importing the TCP/IP HL7 eWay Sample Projects](#) on page 17).
- Create the Environments (see [Creating an Environment](#) on page 139).
- Configure the eWays for your specific requirements (see [Configuring the Properties](#) on page 141).
- Create the Deployment Profiles (see [Creating the Deployment Profile](#) on page 142).
- Create and start the domain (see [Creating and Starting the Domain](#) on page 144).
- Build and deploy your Projects (see [Building and Deploying the Project](#) on page 145).
- Run the Projects (see [Running the Samples](#) on page 145).

7.3 Creating an Environment

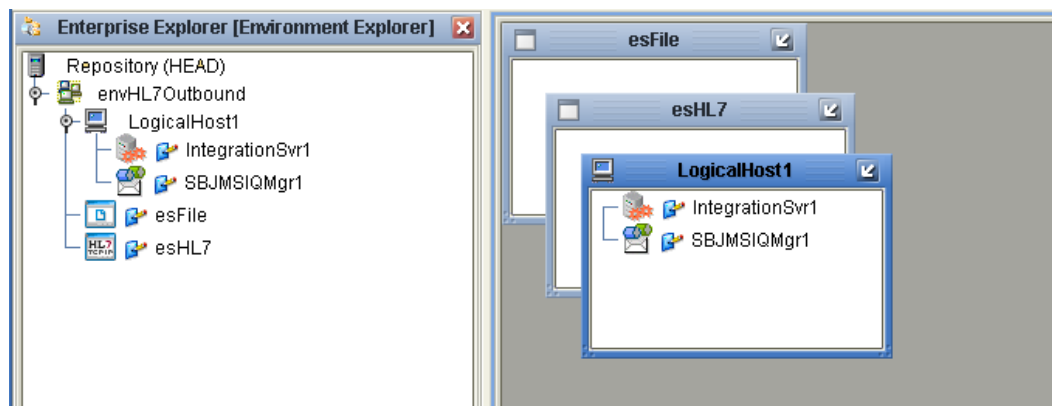
Environments include the external systems, Logical Hosts, integration servers and message servers used by a Project, and contain the configuration information for these components. Environments are created using the Enterprise Designer's Environment Explorer and Environment Editor.

Create the Environment for the prjHL7Outbound Sample

To create the Environment for the **prjHL7Outbound** Outbound sample, do the following:

- 1 From the 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 **envHL7Outbound**.
- 4 Right-click **envHL7Outbound** and select **New > File External System**. Name the External System **esFile**. Click **OK**. **esFile** is added to the Environment Editor.
- 5 Right-click **envHL7Outbound** and select **New > HL7 External System**. Name the External System **esHL7**. Click **OK**. **esHL7** is added to the Environment Editor.
- 6 Right-click **envHL7Outbound** and select **New > Logical Host**. **LogicalHost1** is added to the Environment Editor.
- 7 Right-click **LogicalHost1** and select **New > Sun SeeBeyond Integration Server**. A new Integration Server (**IntegrationSvr1**) is added to the Environment Explorer tree under **LogicalHost1**.
- 8 Right-click **LogicalHost1** and select **New > Sun SeeBeyond JMS IQManager**. A new JMS IQ Manager (**SBJmsIQMgr1**) is added to the Environment Explorer tree under **LogicalHost1**. The Environment Explorer and Environment Editor appear as displayed in Figure 38.

Figure 38 Environment Editor



- 9 Save your current changes.

7.3.1 Creating Environments for the HL7 Inbound and Outbound Samples

To create Environments for the remaining TCP/IP HL7 samples, follow the example given above to create the following:

HL7Inbound Inbound Sample

From the Environment Editor, create the **envHL7Inbound** Environment with the following components:

- HL7 External System named **esHL7**
- New Logical Host
 - ♦ Sun SeeBeyond Integration Server
 - ♦ Sun SeeBeyond JMS IQ Manager

HL7Inbound ForwardMSG Sample

From the Environment Editor, create the **envHL7InboundForwardMSG** Environment with the following components:

- New HL7 External System named **esHL7**
- New Logical Host
 - ♦ New Sun SeeBeyond Integration Server
 - ♦ New Sun SeeBeyond JMS IQ Manager

HL7Outbound OutboundDelayedACK Sample

From the Environment Editor, create the **envHL7OutboundDelayedACK** Environment with the following components:

- New HL7 External System named **esHL7**
- New File External System named **esFile**
- New Logical Host
 - ♦ New Sun SeeBeyond Integration Server
 - ♦ New Sun SeeBeyond JMS IQ Manager

HL7Outbound Forward Sample

From the Environment Editor, create the **envHL7OutboundForward** Environment with the following components:

- New HL7 External System named **esHL7**
- New Logical Host
 - ♦ New Sun SeeBeyond Integration Server
 - ♦ New Sun SeeBeyond JMS IQ Manager

7.4 Configuring the Properties

To run the inbound and outbound Projects as samples, the following properties must be specified for your specific system. For additional information on the eWay properties, see [Creating and Configuring the TCP/IP HL7 eWay](#) on page 41.

HL7Inbound HL7 eWay Sample

The **HL7Inbound** sample contains one eWay, an Inbound TCP/IP HL7 eWay. This Project will work using the default settings.

HL7ForwardMSG (Inbound) HL7 eWay Sample

The **HL7ForwardMSG** inbound sample contains one eWay, a TCP/IP HL7 eWay. This Project will work using the default settings.

HL7Outbound HL7 eWay Sample

The **HL7Outbound** sample contains two eWays, a File eWay, and a TCP/IP HL7 eWay. The **File** eWay will work with the default setting, except for the following that must be set for your specific system:

- **Connectivity Map Properties -> Parameter Settings -> Input file name**
- **Environment Properties -> Inbound File eWay -> Parameter Settings -> Directory**

The **HL7Outbound HL7 eWay** will work with the default setting, except for the following that must be set for your specific system:

- **Environment Properties -> HL7 Outbound Settings -> TCPIP Outbound Settings -> Host**
- **Environment Properties -> HL7 Outbound Settings -> TCPIP Outbound Settings -> Port**

HL7Forward (Outbound) HL7 eWay Sample

The **HL7Forward** outbound sample contains one eWay, an Outbound TCP/IP HL7 eWay. The Outbound TCP/IP HL7 eWay has both the Connectivity Map and Environment properties.

The **HL7Forward** Outbound HL7 eWay will work with the default setting, except for the following that must be set for your specific system:

- **Environment Properties -> HL7 Outbound Settings -> TCPIP Outbound Settings -> Host**
- **Environment Properties -> HL7 Outbound Settings -> TCPIP Outbound Settings -> Port**

HL7OutboundDelayedAck HL7 eWay Sample

The **HL7OutboundDelayedAck** sample contains two eWays, an Inbound File eWay, and an Outbound TCP/IP HL7 eWay.

The **File** eWay will work with the default setting, except for the following that must be set for your specific system:

- **Connectivity Map Properties -> Parameter Settings -> Input file name**
- **Environment Properties -> Inbound File eWay -> Parameter Settings -> Directory**

The TCP/IP HL7 eWay will work with the default setting, except for the following that must be set for your specific system:

- **Environment Properties -> HL7 Outbound Settings -> TCPIP Outbound Settings -> Host**
- **Environment Properties -> HL7 Outbound Settings -> TCPIP Outbound Settings -> Port**

7.4.1 Configuring the Integration Server

You must set your Sun SeeBeyond Integration Server Password property before deploying your Project.

- 1 From the Environment Explorer, right-click **IntegrationSvr1** under your **Logical Host**, and select **Properties** from the shortcut menu. The Integration Server Properties Editor appears.
- 2 Click the **Password** property field under **Sun SeeBeyond Integration Server Configuration**. An ellipsis appears in the property field.
- 3 Click the ellipsis. The **Password Settings** dialog box appears. Enter **STC** as the **Specific Value** and as the **Confirm Password**, and click **OK**.
- 4 Click **OK** to accept the new property and close the Properties Editor.

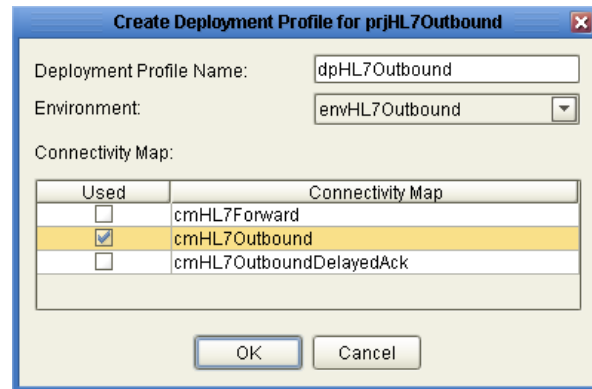
For more information on deploying a Project see the *Sun SeeBeyond Java™ Composite Application Platform Suite Deployment Guide*.

7.5 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 **prjHL7Outbound** Project and select **New > Deployment Profile**. The **Create Deployment Profile for prjHL7Outbound** dialog box appears.
- 2 Enter the Deployment Profile name (for this sample **dpHL7Outbound**). Make sure that the selected Environment is **envHL7Outbound** and the selected Connectivity Map is **cmHL7Outbound** (see [Figure 39 on page 143](#)). Click **OK**.

Figure 39 Create HL7Outbound Deployment Profile



- Click the **Automap** icon as displayed in Figure 40. The Project's components are automatically mapped to their system window as seen in Figure 41.

Figure 40 Deployment Profile - Automap

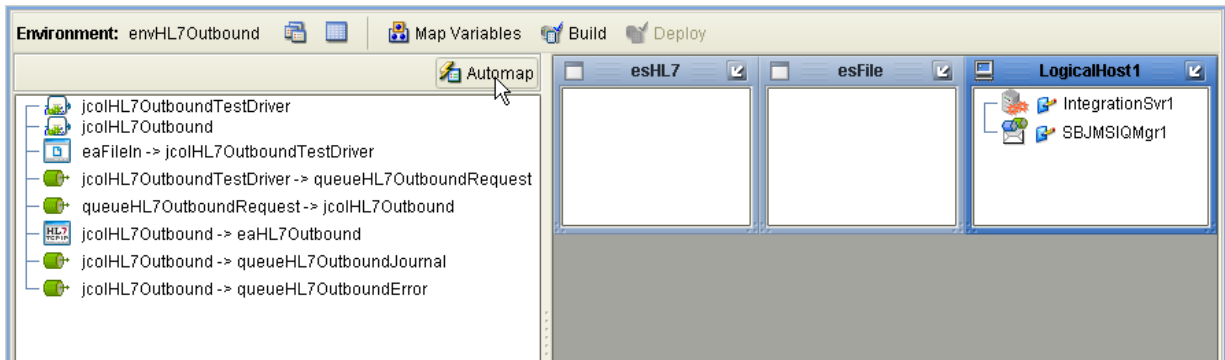
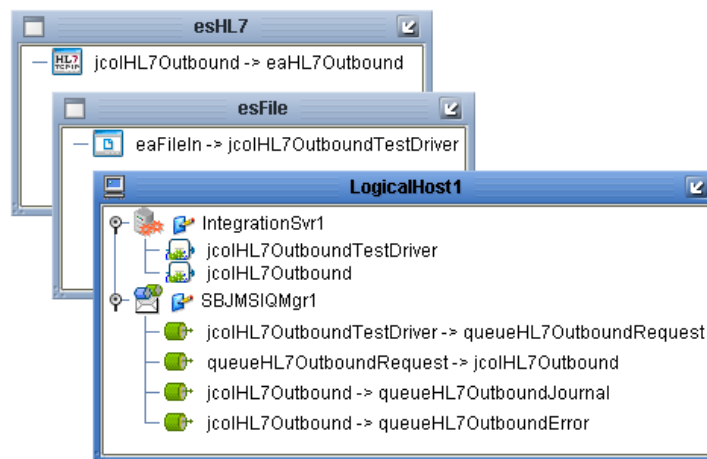


Figure 41 Deployment Profile



- Save your changes to the Repository.

Creating Deployment Profiles for the HL7 Inbound and Outbound Samples

Create Deployment Profiles for the remaining TCP/IP HL7 samples. Follow the procedures in the prior example to create the following:

HL7Inbound Inbound Sample

To create the **dpHL7Inbound** Deployment Profile, use the following:

- Environment: **envHL7Inbound**
- Connectivity Map: **cmHL7Inbound**

HL7Inbound ForwardMSG Sample

To create the **dpHL7InboundForwardMSG** Deployment Profile, use the following:

- Environment: **envHL7InboundForwardMSG**
- Connectivity Map: **cmHL7InboundForwardMSG**

HL7Outbound OutboundDelayedACK Sample

To create the **dpOutboundDelayedACK** Deployment Profile, use the following:

- Environment: **envOutboundDelayedACK**
- Connectivity Map: **cmOutboundDelayedACK**

HL7Outbound Forward Sample

To create the **dpHL7OutboundForward** Deployment Profile, use the following:

- Environment: **envHL7OutboundForward**
- Connectivity Map: **cmHL7OutboundForward**

7.6 Creating and Starting the Domain

To deploy your Project you must first create a domain. A domain is an instance of a Logical Host.

Create and Start the Domain

- 1 Navigate to your **<JavaCAPS51>\logicalhost** directory (where **<JavaCAPS51>** is the location of your Sun SeeBeyond 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 Building and Deploying the Project

The Build process compiles and validates the Project's Java files and creates the Project EAR file. Do the following for each of your Deployment Profiles.

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*.

7.8 Running the Samples

The inbound and outbound TCP/IP HL7 sample Projects demonstrate several different operations (see [“TCP/IP HL7 eWay Sample Projects” on page 137](#)).

To run a sample operation (Project), deploy only the sample operation (Project) that you would like to test, leaving the other TCP/IP HL7 Projects undeployed.

Projects are deployed and undeployed from the **Enterprise Manager**. For information on deploying and undeploying your Projects using the Sun SeeBeyond Enterprise Manager, see the *Sun SeeBeyond eGate™ Integrator System Administration Guide*.

To run your deployed sample Project do the following:

- 1 From your configured input directory, paste the sample input file to trigger the eWay.
- 2 From your output directory, verify the output data.

Index

A

- acknowledgment level 39
- acknowledgment processing
 - inbound eWay 29
 - outbound 33
- Action 70
- alerter.fatal("error msg", "HL7") 36
- alerting 24
- alerts 39
- Automap 143

B

- basic features 9
- Block Checksum 29
- Block Size 29

C

- Canned HL7 NAK 29, 30, 34
- canned HL7 NAK 30
 - outbound 34
- Carriage Return 29
- client
 - connected as 37
- Collaboration
 - adding an OTD 116
 - inbound
 - overview 106
 - outbound
 - overview 110
 - test
 - overview 114
- Collaboration Properties 116
- Collaborations
 - customizing 115
 - overview 105
- configurable properties
 - connection types 37
 - connected as a TCP/IP server 37
 - envelope type
 - HLLP 38
 - MLLP 38
- connection type 37

- conventions, text 11

D

- data direction
 - outbound 31
- delimiter
 - level 123
 - specifying 123, 124
- delimiter properties 125
 - Optional Mode 126
 - Precedence 127
 - Terminator Mode 126
 - Type 125
- delimiters 123
 - changing 123
- Deployment Profile 142
 - Autmap 143
 - Automap 143
- direction 37

E

- End of Data 29
- Environment 139
 - creating 139
- error queues 23, 39
- eWay
 - components 136
 - location 42
- eWay properties, overview 9
- eWays
 - creating 42
- exporting
 - a Project 115
- External Application
 - creating 41
 - selecting 41

F

- failed message handling 35
- fatal alert 36

G

- general operation, eWay 9
- generic HL7 OTDs 23, 118

H

- heap size
 - adjusting heap memory size 18

- increasing 18
- HL7 sequence numbering 35
- HLLP 37, 38

I

- importing
 - a Project 115
- inbound eWay data flow 27
- inbound eWay functionality 27
- installation 14

J

- Java Integration Suite
 - functionality 23
- JMS error queues 23
- JMS journal queues 23
- journal queues 23
- journaling 39

L

- logging 24
- lower layer protocol 37

M

- message verification
 - outbound 33
- MLLP 37
- monitoring 24, 39

N

- non-blocking I/O 35

O

- operating systems
 - requirements 13
 - supported 9, 13
- Options Setup
 - dialog box 18
- OTD
 - adding an OTD to a Collaboration 116
 - check-in 120
 - check-out 120
 - Delimiters 123
 - Editor 120
 - modifying 120
 - tester 134
- OTD Editor 119, 120

- opening 119
 - viewing an OTD 119
- OTD properties 128
 - delimiters 124
 - element properties 130
 - field properties 124, 131
 - node properties 128
 - Node Type 132
- OTD Tester 134
 - test data 135
- OTDs
 - adding to a Collaboration 116
 - editing 118
 - generic 118
- outbound eWay
 - data flow 31
 - functionality 30
 - standard messaging mode 31
- outbound eWay Connectivity Map properties 73
- outbound from eGate 31
- OutOfMemoryError
 - increase heap size 18

P

- platforms
 - requirements 13
 - supported 13
- Project
 - completing 138
 - components 136
 - creating a copy 115
 - overview 136
- properties
 - Acknowledgment Level 59, 83
 - Action on Max Failed Read Retry 71, 95
 - Action on Max Nak Received 72, 96
 - Action on Max Nak Sent 72, 96
 - Action on Max No Response 70, 94
 - Action on Nak Received 71, 95
 - Action on No Response 70, 94
 - Alternate Character Set Handling Scheme 65, 89
 - Always Create New Connection 80
 - At Fixed Rate 56, 58
 - Auto Reconnect Upon Matching Failure 80
 - Character Set 65, 89
 - Close Notification 53
 - Communication Control 68, 92
 - configuring 41
 - Conformance Statement ID 65, 89
 - Connection Pool Size 53
 - Connection Type 47, 75
 - Connectivity Map properties 41
 - modifying 42

- Country Code 65, 88
- Dedicated Session Mode 46
- Delay 55, 57
- editor 43
- eGate Sends App Acks 59, 83
- Enable 66, 90
- Enable Journaling 69, 93
- Encoding Characters 63, 87
- End Block Character 61, 85
- End Data Character 61, 85
- Environment Explorer properties 41
- Field Separator 63, 87
- Forward External Acks to eGate 60, 84
- General Inbound Settings 45
- General Outbound Settings 74
- HL7 Acknowledgment 59, 83
- HL7 MSH Segment 63, 87
- HL7 Recourse Action 70, 94
- HL7 SFT Segment 66, 90
- HLLP Checksum Enabled 61, 85
- Host 97, 101
- IdleTimeout 54
- Inbound eWay Connectivity Map 43
- inbound eWay Connectivity Map 45
- Keep Alive 48, 76
- LLP Type 61, 85
- Lower Layer Protocol 61, 85
- Max Binding Retry 51, 82
- Max Canned NAK Send Retry 69, 93
- Max Connection Retry 81
- Max Data Size 45, 74
- Max Empty Read Retry 68, 92
- Max NAK Receive Retry 69, 93
- Max NAK Send Retry 69, 93
- Max No Response 68, 92
- modifying 42
- Period 55, 57
- Principal Language of Message 65, 89
- Processing ID 64, 88
- Receive Buffer Size 48, 76
- Receiving Application 64, 88
- Receiving Facility 64, 88
- Retry Binding Interval 51, 82
- Retry Connection Interval 81
- Schedule Type 55, 57
- Scheduler 55, 57
- Scope Of Connection 53
- Scope Of State 46, 74
- Security 64, 88
- Send Buffer Size 48, 77
- Sending Application 63, 87
- Sending Facility 63, 87
- Sequence Number Enabled 62, 86
- Sequence Number Protocol 62, 86, 100, 104

- Server Socket Factory Implementation Class
 - Name 47
- ServerPort 97, 101
- ServerSO Timeout 47
- ServerSoTimeout 75
- Socket Factory Implementation Class Name 79
- Software Binary ID 67, 91
- Software Certified Version or Release Number 66, 90
- Software Install Date 67, 91
- Software Product Information 67, 91
- Software Product Name 67, 91
- Software Vendor Organization 90
- SoLinger 49, 77
- SoLinger Timeout 49, 78
- SoTimeout 50
- Start Block Character 61, 85
- TCPIP Inbound Settings 47
- TCPIP Inbound Settings - Client Connection
 - Establishment 52
- TCPIP Inbound Settings - Connection
 - Management 53
- TCPIP Inbound Settings - Server Port Binding 51
- TCPIP Outbound Settings 75, 101
- TCPIP Outbound Settings - Connection
 - Establishment 80
- TCPIP Outbound Settings - Server Port Binding 82
- TCPIP Server Inbound Schedules - Listener
 - Schedule 55
- TCPIP Server Inbound Schedules - Service
 - Schedule 57
- TcpNoDelay 50, 79
- Time To Wait Before Attempting Connection 80
- Time to Wait Before Attempting Connection 52
- Time To Wait For A Response 68, 92
- Timeout For Delayed Ack 60, 84
- Validate MSH 65, 89
- Version ID 64, 88
- Properties Sheet 43
- properties sheet
 - properties
 - properties sheet 43

R

- recourse action
 - outbound 34
- Recourse Actions 36
- recourse actions 36
 - Exit 36
 - calling fatal alert 36
- inbound 30
- properties

- inbound 70
- outbound 94
- Resend 36
- Skip Message 36
- Resource Adapter 22
- roles
 - outbound delayed ACK 31, 32
 - outbound delayed ACK role 32
 - outbound forwarder role 33

S

- sequence numbering 35
- sequence numbering protocol 35
- server
 - connected as 37
- SFT segment
 - support 39
- SKIPPED Events 35
- Start of Block 29
- supported operating systems 13

T

- TCP/IP eWay
 - properties
 - Connectivity Map properties 41
 - Environment Explorer properties 41
- TCP/IP HL7 eWay
 - architecture 22
 - features 9
 - installation 14
 - modes 24
 - delayed ACK mode 25
 - standard mode 24
 - properties 41, 43
 - roles 24
 - supported operating systems 9
 - upload requirements 13
- tester
 - OTD 134
- text conventions 11

V

- verification
 - outbound 33